i.MX 7ULP Applications Processor Reference Manual

Document Number: IMX7ULPRM
Rev. 0, 06/2019

Contents

Section number **Title** **Page**

Chapter 7
AHB Local Memory Controller

Chapter 8
Bit Manipulation Engine (BME)

Chapter 9
Messaging Unit (MU)

Chapter 15
Direct Memory Access Multiplexer (DMAMUX)

Chapter 16
AIPS-Lite

Chapter 17
Cross Bar Switch

Chapter 18
Memory Architecture

Chapter 19
System and Peripheral Memory Maps

Chapter 20
Quad Serial Peripheral Interface Controller (QuadSPI)

Chapter 21
FlexBus

Chapter 22
Multi Mode DDR Controller (MMDC)

Chapter 23
Ultra Secured Digital Host Controller (uSDHC)

Chapter 24
Clocking

Chapter 25
Peripheral Clock Control

Chapter 26
System Clock Generator (SCG)

Chapter 27
Power Management

Chapter 28
Power Management Controller (PMC)

Chapter 29
Asynchronous Wakeup Interrupt Controller (AWIC)

Chapter 30
Multicore System Mode Controller (MSMC)

Chapter 31
System Integration Module (SIM)

Chapter 32
Low-Leakage Wake-Up Unit (LLWU)

Chapter 33
Wake-Up Unit (WKPU)

Chapter 34
On-Chip One-Time-Programmable Controller (OCOTP_CTRL)

Chapter 35
Reset and Boot

Chapter 36
Signal Multiplexing

Chapter 37
Input/Output Multiplexing Controller (IOMUXC)

Chapter 38
Trigger MUX (TRGMUX)

Chapter 39
Port Control (PCTL)

Chapter 40
Rapid General-Purpose Input and Output with 2 Ports (RGPIO2P)

Chapter 41
Debug

Chapter 42
Secured JTAG Controller (SJC)

Chapter 43
JTAG Controller (JTAGC)

Chapter 44
Integrated Interchip Sound (I2S) / Synchronous Audio Interface (SAI)

Chapter 45
MIPI-DSI Controller

Chapter 46
2D Graphics Processing Unit (GPU2D)

Chapter 47
3D Graphics Processing Unit (GPU3D)

Chapter 48
Enhanced LCD Interface (LCDIF)

Chapter 49
Video Interface Unit (VIU)

Chapter 50
Flexible Input/Output (FlexIO)

Chapter 51
Low Power Universal Asynchronous Receiver/Transmitter (LPUART)

Chapter 52
Low Power Serial Peripheral Interface (LPSPI)

Chapter 53
Low Power Inter-Integrated Circuit (LPI2C)

Chapter 54
Universal System Bus On-The-Go (USB-OTG)

Chapter 55
Universal System Bus Physical Layer (USB-PHY)

Chapter 56
USB DCD

Chapter 57
Cyclic Redundancy Check (CRC)

Chapter 58
Low Power Timer (LPTMR)

Chapter 59
Watchdog Timer (WDOG)

Chapter 60
Time Stamp Timer (TSTMR)

Chapter 61
External Watchdog Monitor (EWM)

Chapter 62
Low Power Timer/Pulse Width Modulation Module (TPM)

Chapter 63
Low Power Periodic Interrupt Timer (LPIT)

Chapter 64
Secure Non-Volatile Storage (SNVS)

Chapter 65
12-bit Analog to Digital Converter (ADC)

Chapter 66
Digital to Analog Converter (DAC)

Chapter 67
Analog Comparator (ACMP)

Chapter 1
About this manual

1.1 Audience

This reference manual is intended for system software and hardware developers and applications programmers who want to develop products with this device. It assumes that the reader understands operating systems, microprocessor system design, and basic principles of software and hardware. The manual describes the features and functionality of the i.MX 7ULP device.

1.2 Organization

This manual has two main sets of chapters.

- Chip-specific chapters: These chapters contain information that applies to all components on the chip.
- Module chapters: These chapters contain details of functional block level information with memory map and register configuration, used in this device

Both sets of chapters are organized into functional groupings that detail particular areas of functionality.

- Examples of these groupings are clocking, timers, and communication interfaces.
- Each grouping first includes the full chip-specific chapter followed by the chapters that provide a technical description of individual modules categorized under that group. For example, for clocking information, there is complete chip-specific Clocking chapter followed by the clock modules: PCC and SCG

1.3 Module descriptions

Each module chapter has three main parts:

- The first section, *Chip-specific [module name] information*, contains:

- A table listing the related references for chip-specific chapters like memory map, clock distribution, power management, and signal multiplexing, to facilitate the user to have a quick look at any of these chapters. See Figure 1-1
- Followed by this table, we have textual information about the module and then a configuration table which contains the general chip-specific information such as the number of module instances on the chip and connections between the module and other modules, configurable features, signal names, etc. See Figure 1-1
- Followed by this are any other specific chip-specific sections that detail the functionality of the device which may be different from what is mentioned in the complete block guide. These sections may also contain additional information needed to understanding of the module in this device. See Figure 1-2

NOTE

Read these sections *first* because its content is crucial to understanding the information in other sections of the chapter.

- The subsequent sections provide general information about the module, including its signals, registers, and functional description.

EXAMPLE

Chapter 14
Direct Memory Access Controller (DMA)

14.1 Chip-specific DMA information

Table 14-1. Reference links to related information

Topic	Related module	Reference
Full description	DMA	DMA
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

14.1.1 DMA

DMA is a second-generation module of the enhanced direct memory access (eDMA2) controller, which is capable of performing complex data transfers with minimal intervention from a host processor. There are two DMA instances in i.MX 7ULP. Each DMA supports 32 DMA channels. The transfer control descriptors for each of the 32 channels are located in system memory. The following table shows the configuration of the DMA.

Table 14-2. DMA configuration

Parameter	Description
Name	Enhanced Direct Memory Access v2 (eDMA2)
Instances	2
Configurable features	• DMA0-1: 32-channels • DMA0-1: 64-bit AHB
Interface speed	NA
External I/O pins	NA

Figure 1-1. Example: Related references and configuration table

Table 11-2. XRDC configuration

Parameter	Description
Name	Extended Resource Domain Control
Instances	1
Configurable features	Refer to Multicore Architecture for details
Interface speed	NA
External I/O pins	NA

11.1.2 XRDC configuration for MDAC, MRC, and PAC registers

Table 11-3. MDAC configuration

XRDC MDA	MDAC_REG_NUM	DEFAULT_DID	MDAC_CPU	MDAC_INST_NUM
CM4CODE	2	0	1	6'd00
CM4SYS	2	0	1	6'd01
CM4DMA	1	0	0	6'd02
CA7	2	1	1	6'd03
LCDIF	1	1	0	6'd04
GPU3D	1	1	0	6'd05
CA7DMA	1	1	0	6'd06
AXBS2NIC1	1	1	0	6'd07
CAAM	1	1	0	6'd08
USB0/1	1	1	0	6'd09
VIU	1	1	0	6'd10
SDHC0	1	1	0	6'd11
SDHC1	1	1	0	6'd12
GPU2D	1	1	0	6'd13

Table 11-4. MRC configuration

MRC#	SLAVE_NUM	Slave Memory	REGION_NUM	SEMA4_NUM	SEMA4_INST
MRC0	1	M4 TCMs	4	16	SEMA4_0

Table continues on the next page...

Figure 1-2. Example: chapter additional chip-specific information

1.3.1 Example: chip-specific information that clarifies content in the same chapter

The example below shows chip-specific information that clarifies general module information presented later in the chapter. In this case, the chip-specific register reset values supersede the reset values that appear in the register diagram.

22.1.1 External Bus Interface (FlexBus)

The External Bus Interface (FlexBus) module provides external memory expansion and provides connection to external peripherals with a parallel, memory-mapped interface. The FlexBus supports asynchronous and synchronous interface to external ROM, NOR flash, SRAM, PSRAM, programmble logic devices and other memory-mapped slave devices. The FlexBus supports 8-bit, 16-bit and 32-bit data bus width options and the multiplexing of address and data configuration. The following table shows the configuration of the FlexBus.

Table 22-2. Flexbus configuration

Parameter	Description
Name	External Bus Interface (FlexBus)
Instances	1
Configurable features	NA
Interface speed	Up to 100 MHz
External I/O pins	See the attached IOMUXC spreadsheet for pin details.

22.1.2 Transfer Acknowledge signal

The Transfer Acknowledge Signal ($\overline{\text{FB\_TA}}$) is hard-wired in the design of I.MX 7ULP, so this signal is not available. See the FB_CSPMCR register for the multiplexing of other signals.

22.1.3 Reset value for CSCRn register

The reset valus of CSCR0-CSCR5 for i.MX 7ULP are specified in the table below

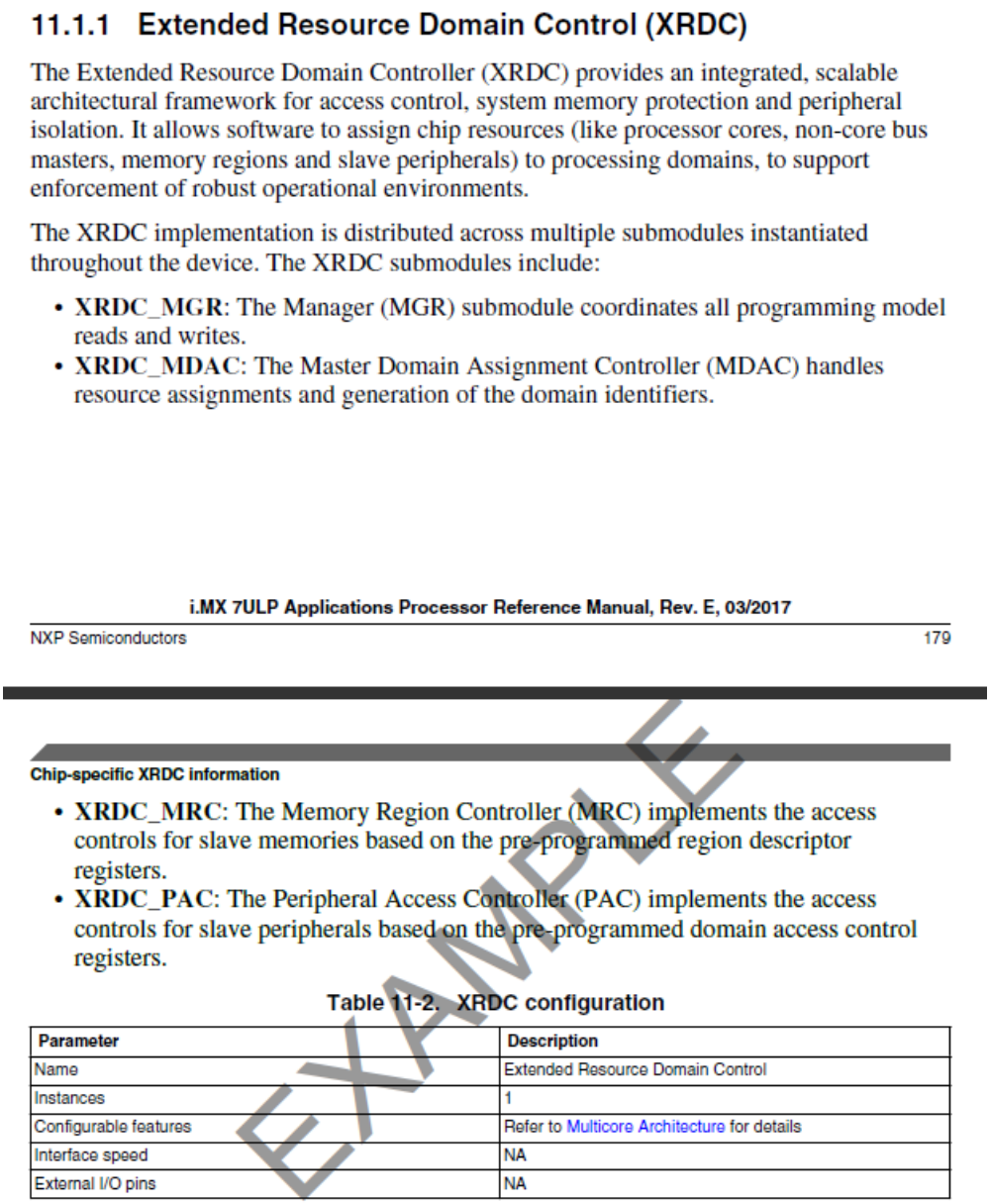
Table 22-3. CSCRn reset values

Register name	Reset value
CSCR0	0x003F_FC00
CSCR1-CSCR5	0x0000_0000

Figure 1-3. Example: chip-specific information that clarifies content in the same chapter

1.3.2 Example: chip-specific information that refers to a different chapter

The chip-specific information below refers to another chapter's chip-specific information. In this case, read both sets of chip-specific information before reading further in the chapter. In the following figure, the XRDC chapter refers to Multicore Architecture chapter in the configuration table



11.1.1 Extended Resource Domain Control (XRDC)

The Extended Resource Domain Controller (XRDC) provides an integrated, scalable architectural framework for access control, system memory protection and peripheral isolation. It allows software to assign chip resources (like processor cores, non-core bus masters, memory regions and slave peripherals) to processing domains, to support enforcement of robust operational environments.

The XRDC implementation is distributed across multiple submodules instantiated throughout the device. The XRDC submodules include:

- **XRDC_MGR**: The Manager (MGR) submodule coordinates all programming model reads and writes.
- **XRDC_MDAC**: The Master Domain Assignment Controller (MDAC) handles resource assignments and generation of the domain identifiers.

i.MX 7ULP Applications Processor Reference Manual, Rev. E, 03/2017

NXP Semiconductors 179

Chip-specific XRDC information

- **XRDC_MRC**: The Memory Region Controller (MRC) implements the access controls for slave memories based on the pre-programmed region descriptor registers.
- **XRDC_PAC**: The Peripheral Access Controller (PAC) implements the access controls for slave peripherals based on the pre-programmed domain access control registers.

Table 11-2. XRDC configuration

Parameter	Description
Name	Extended Resource Domain Control
Instances	1
Configurable features	Refer to Multicore Architecture for details
Interface speed	NA
External I/O pins	NA

Figure 1-4. Example: chip-specific information that refers to a different chapter

1.4 Register descriptions

Module chapters present register information in:

- Memory maps containing:
  - Offset from the module's base address or the absolute address
  - The name and acronym/abbreviation of each register
  - The width of each register (in bits)

    - Each register's reset value
    - Access of each register
- Register figures
- Field-description tables
- Associated text

NOTE

Some modules may contain multiple memory maps for different instances if there are actual differences in the register or bitfields within the instances. For example, PCC module has four memory maps (one for each instance) and similarly SCG module has two memory maps.

The register figures show the field structure using the conventions in the following figure.

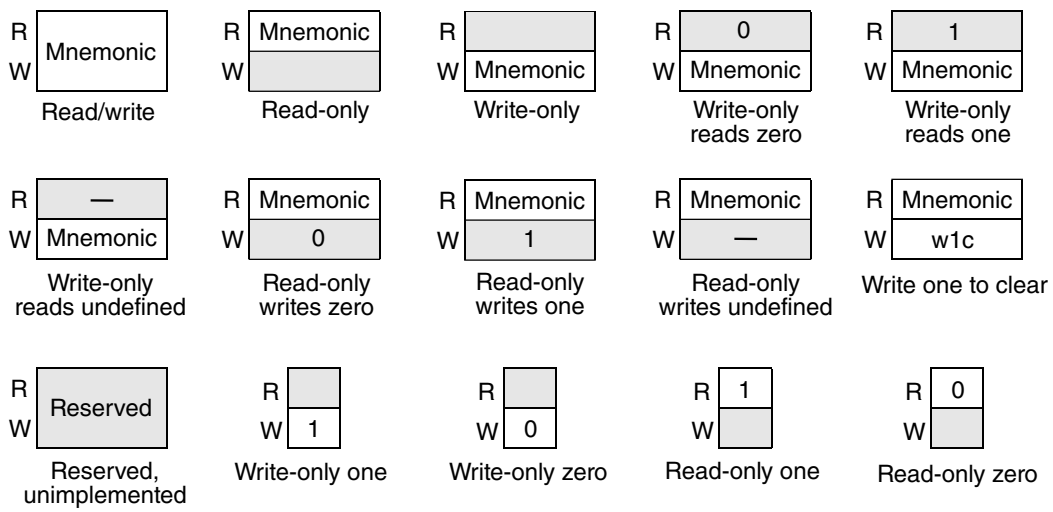


Figure 1-5. Register figure conventions

1.5 Conventions

1.5.1 Notes, Cautions, and Warnings

Note, Caution, and Warning notices appear throughout this manual. Each notice type alerts readers to a specific kind of information.

NOTE

Notes convey information that may be tangential to a topic or that may not apply to all readers.

CAUTION

Caution notices call out information that readers should know before taking further action. Cautions frequently point to trouble spots that may damage the chip or board.

WARNING

Warning notices inform readers about actions that could result in unwanted consequences, especially those that may cause bodily injury.

1.5.2 Numbering systems

The following suffixes identify different numbering systems:

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is written 101b. In some cases, binary numbers are shown with the prefix *0b*.
d	Decimal number. Decimal numbers are followed by this suffix only when the possibility of confusion exists. In general, decimal numbers are shown without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is written 3Ch. In some cases, hexadecimal numbers are shown with the prefix *0x*.

1.5.3 Typographic notation

The following typographic notation is used throughout this document:

Example	Description
placeholder, x	Items in italics are placeholders for information that you provide. Italicized text is also used for the titles of publications and for emphasis. Plain lowercase letters are also used as placeholders for single letters and numbers.
`code`	Fixed-width type indicates text that must be typed exactly as shown. It is used for instruction mnemonics, directives, symbols, subcommands, parameters, and operators. Fixed-width type is also used for example code. Instruction mnemonics and directives in text and tables are shown in all caps; for example, BSR.
SR[SCM]	A mnemonic in brackets represents a named field in a register. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets and separated by a colon represent either: • A subset of a register's named field For example, REVNO[6:4] refers to bits 6–4 that are part of the COREREV field that occupies bits 6–0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7–0 of the XAD bus.

1.5.4 Special terms

The following terms have special meanings:

Term	Meaning
asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
reserved	Refers to a memory space, register, field, or programming setting. Device operation is not guaranteed when reserved locations are written to any value. • Do not modify the default value of a reserved programming setting, such as the reset value of a reserved register field. • Consider undefined locations in memory to be reserved.
w1c	Write 1 to clear: Refers to a register bitfield that must be written as 1 to be "cleared."

Chapter 2
Introduction

2.1 Overview

i.MX 7ULP is an asymmetric microprocessor consisting of two separate processing domains. The domains are divided into an application domain and a real time domain. The application domain is built around a small Arm® Cortex® A7 core (hereafter referred as CA7 or A7) optimized for 500 to 800 MHz Linux and Android-based applications, and the real time domain is built around a Arm® Cortex® M4 core (hereafter referred as CM4 or M4) that is optimized for the lowest possible active and standby power. The design enables clean separation between two processing domains, where each has separate power, clocking and peripheral islands, but the bus fabric of each domain is tightly integrated for efficient communication. The part is streamlined to minimize pin count, enabling small packages and simple system integration. This microprocessor is intended for applications where efficiency and simple system integration is important.

2.2 Key features

Following is the block diagram of i.MX 7ULP showing key modules and features:

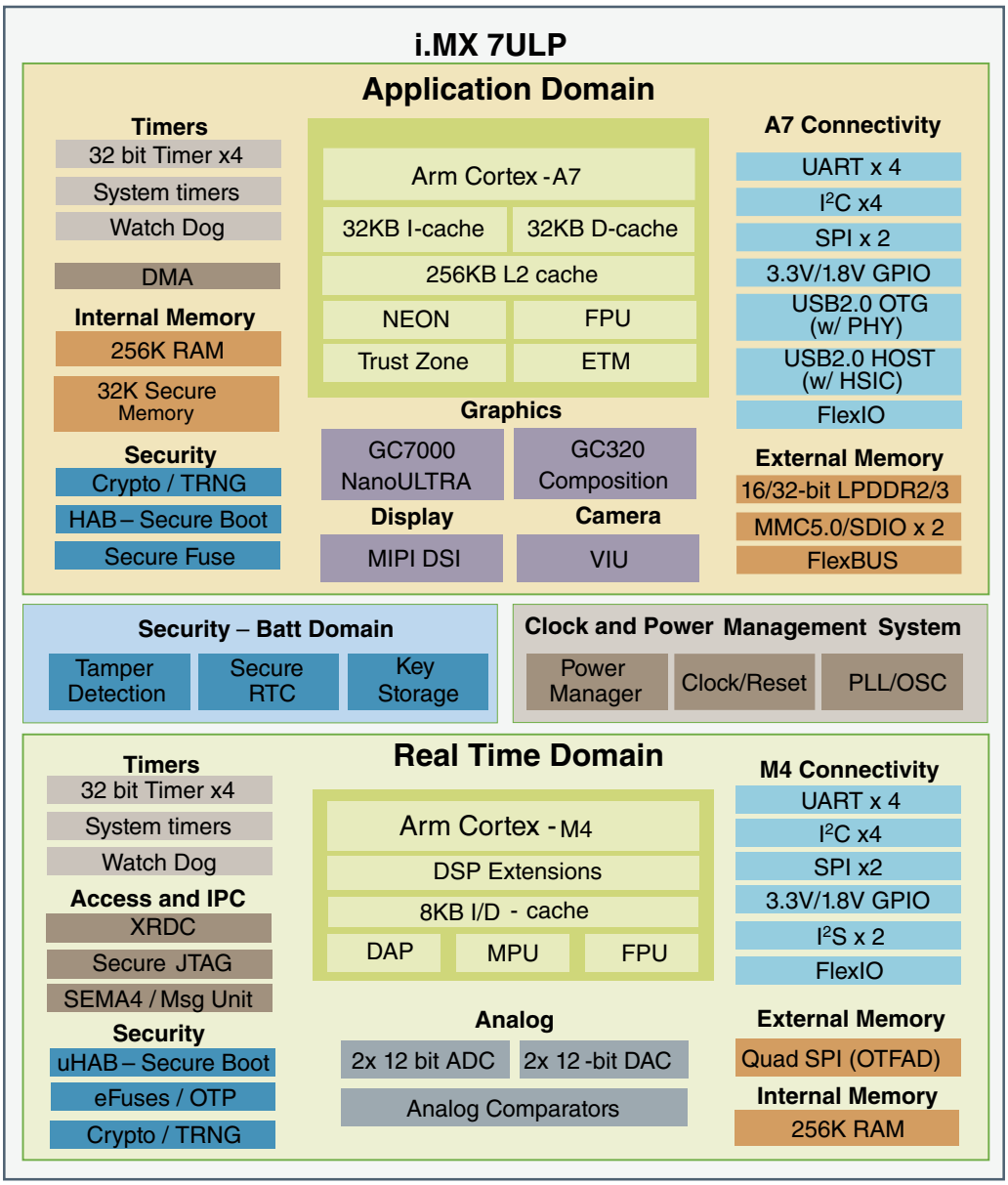


Figure 2-1. i.MX 7ULP Block Diagram

Application domain	
Arm® Cortex® A7	• Normal operating frequency of 500 MHz and overdrive up to 800 MHz. See the i.MX 7ULP Data Sheet for Normal Operating Frequency. • 32 KB instruction cache and 32 KB data cache • 256 KB of L2 cache • NEON SIMD engine • Floating Point Unit (FPU) • 32-channel DMA
Memory architecture and memory controllers	• 256 KB of RAM • 96 KB of boot ROM • DRAM interface supporting x16/x32 LPDDR2/3 running up to 380 MHz. See the i.MX 7ULP Data Sheet for DRAM interface supporting frequency. • Two eMMC/SD host interfaces supporting eMMC5.0 and SD3.0

Table continues on the next page...

Application domain	
Graphic, display and camera interface	• GC7000 Nano Ultra GPU supporting OpenGL ES2.0/1.1, Desktop OpenGL 2.1, OpenVG1.1, and built-in graphics functions for special effects. • Dual lane MIPI DSI interface operating up to 800 Mbps • Parallel camera sensor interface
Communication and timer peripherals	• One High-Speed USB-OTG port with integrated PHY and 1 High-Speed USB port with HSIC interface • HSIC and ULPI interface • Two SD host interface supporting SDIO3.0 • Four UART ports • Two SPI ports • Four I2C ports • Multiple timer peripherals suited for task scheduling and time keeping
Security[1]	• High assurance boot • Cryptographic acceleration supporting symmetric and asymmetric ciphering algorithms and SHA-256 hashing algorithm. • Random number generator • 32 KB secure memory

1. See the i.MX 7ULP Security Reference Manual, for details on security information

Real time domain	
Arm® Cortex® M4	• Normal operating frequency of 100 MHz and overdrive 200 MHz. See the i.MX 7ULP Data Sheet for Normal Operating Frequency. • FPU • Memory Protection Unit (MPU)
Memory architecture and memory controllers	• 256 KB RAM, zero wait state accessed by M4 • 64 KB boot ROM • Serial flash interface to quadSPI and octalSPI flash devices • 8 KB cache
Analog	• Two 12-bit ADC • Two DAC • Two analog comparators
Communication and timer peripherals	• Four UART ports • Two SPI ports • Four I2C ports • Two I2S ports • Multiple timer peripherals suited for task scheduling and time keeping
Security[1]	• High assurance boot • Cryptographic acceleration supporting SHA-256 hashing algorithm • Cryptographic acceleration supporting AES ciphering algorithm • Secure Non-Volatile Storage including cryptographic key, time counter, monotonic counter, and general purpose security information. • True random number generator • On-The-Fly decryption of the encrypted code stored in external flash device • One-Time Programmable electrical fuse used for security keys and configuring other security related functions

1. See the i.MX 7ULP Security Reference Manual, for details on security information

Following are the low-power features:

- On chip power management including regulators, drivers and switches for flexible power supplies, efficient power consumption, and short wake-up time

- Multiple power domains and ultra low power modes allow flexible power saving
- Voltage and frequency scaling in dynamic operating modes
- Software-controlled clock gating for cores and peripherals

2.3 Module description

2.3.1 Memory architecture and memory controllers

Table 2-1. Memory architecture

Module	Description
SRAM0-1	An AXI RAM Controller (RAMC), which acts as an interface between the AXI bus and RAM, is used on each SRAM connected to the NIC crossbar.
TCM RAM	Tightly Coupled Memory (TCM) RAM is tightly integrated to the M4 processor. M4 accesses this memory with zero wait-state. There is a backdoor port that allows M4 DMA and other bus masters in the device to access this memory.
MMDC	The Multi Mode DDR Controller (MMDC) is a configurable DDR controller that provides interface to LPDDR2 or LPDDR3 memory. The MMDC consists of a core and PHY. The core is responsible for communication with the system through AXI interface, DDR commands generation, DDR command optimizations, and read/write data path. The PHY performs timing adjustment using special calibration mechanisms to ensure data capture margin at the supported clock rate.
FlexBus	The External Bus Interface (FlexBus) module provides external memory expansion and provides connection to external peripherals with a parallel, memory-mapped interface. The FlexBus supports asynchronous and synchronous interface to external ROM, NOR flash, SRAM, PSRAM, programmble logic devices and other memory-mapped slave devices.
QSPI	The Quad Serial Peripheral Interface (QSPI) module provides an interface to various types of serial flash memory. The QSPI interface allows only one serial flash connection. It supports 1-bit, 4-bit and 8-bit SPI bus width.
uSDHC0-1	The ultra Secured Digital Host Controller (uSDHC) provides the interface between the host system and SD, SDIO or eMMC cards. The uSDHC acts as a bridge, passing host bus transactions to the cards by sending commands and performing data accesses to/from the cards or devices. It handles SD, SDIO and eMMC protocol at transmission level.

2.3.2 DMA and Bus Fabrics

Table 2-2. DMA and Bus Fabrics

Module	Description
DMA0-1	Direct Memory Access (DMA) is capable of performing complex data transfers with minimal intervention from a host processor. Each DMA module supports 32 DMA channels. The transfer control descriptors for each of the 32 channels are located in system memory.

Table continues on the next page...

Table 2-2. DMA and Bus Fabrics (continued)

Module	Description
DMAMUX0-1	The Direct Memory Access Multiplexer (DMAMUX) module routes DMA sources, called slots, to any of the supported DMA channels.
NIC0-1	The AMBA Network Interconnect Crossbar (NIC) is a highly configurable and high performance AMBA-compliant network infrastructure which arbitrates between multiple AXI or AHB masters to grant access to internal or external memories or other slave devices. It supports connectivity between several slave and master ports for parallel processing. It uses a hybrid round-robin arbitration scheme and contains frequency converters, data width converters, bus protocol converter, and AXI channel buffers.
AXBS-Lite	AHB-Lite Cross Bar Switch (AXBS-Lite) is a crossbar switch module that arbitrates between multiple AHB masters to grant access to downstream AHB slave devices. The crossbar enables multiple masters to connect to slaves and parallel processing.
AIPS-Lite	AHB-Lite to IP Bus Interface (AIPS-Lite) module interfaces an AHB-Lite bus to the peripheral bus. Each AIPS module occupies a total of 64 MB of address space and is equally divided into 128 4-KB slots.
AHB-PBridge	The AHB-Peripheral Bridge (AHB-PBridge) module interfaces an AHB bus to the peripheral bus. There are 2 AHB-PBridge instances in this device. Each AHB-PBridge module supports 128 IP slots. Each slot occupies 64 KB of address space.
PortSplitter	The PortSplitter gasket splits an AHB master port to multiple AHB slave ports based on address decoding.

2.3.3 Multicore peripherals and resource domain control submodules

Table 2-3. Multicore peripherals and resource domain control submodules

Module	Description
BME	The Bit Manipulation Engine (BME) provides hardware support for atomic read-modify-write memory operations to the peripheral address space. This architectural capability is also known as "decorated storage" as it defines a mechanism for providing additional semantics for load and store operations to memory-mapped peripherals beyond just the reading and writing of data values to the addressed memory locations.
MU_A-B	Messaging Unit (MU) is a shared peripheral with a 32-bit IP bus interface and interrupt request signals to each host processor. The MU exposes a set of registers to each processor which facilitate inter-processor communication via 32-bit words, interrupts and flags. Interrupts may be independently masked by each processor to allow polled-mode operation.
SEMA42	The Hardware Semaphore (SEMA42) module provides the hardware support needed in multicore systems for implementing semaphores and provides a simple mechanism to achieve "lock/unlock" operations via a single write access.
XRDC	The Extended Resource Domain Controller (XRDC) provides an integrated, scalable architectural framework for access control, system memory protection and peripheral isolation. It allows software to assign chip resources (like processor cores, non-core bus masters, memory regions and slave peripherals) to processing domains, to support enforcement of robust operational environments. The XRDC

Table continues on the next page...

Table 2-3. Multicore peripherals and resource domain control submodules (continued)

Module	Description
	implementation is distributed across multiple submodules instantiated throughout the device.
MGR	The XRDC Manager (MGR) submodule coordinates all programming model reads and writes.
MDAC	The XRDC Master Domain Assignment Controller (MDAC) submodule handles resource assignments and generation of the domain identifiers.
MRC	The XRDC Memory Region Controller (MRC) submodule implements the access controls for slave memories based on the pre-programmed region descriptor registers.
PAC	The XRDC Peripheral Access Controller (PAC) implements the access controls for slave peripherals based on the pre-programmed domain access control registers.

2.3.4 Connectivity and Communications

Table 2-4. Connectivity and communications modules

Module	Description
FlexIO0-1	The Flexible Input/Output (FlexIO) module is capable of supporting a wide range of protocols including, but not limited to: UART, I2C, SPI, I2S, camera interface, display interface, PWM waveform generation, and so on.
LPI2C0-7	The Low Power Inter-Integrated Circuit (LPI2C) module implements an efficient interface to an I2C bus as a master. The LPI2C can continue operating while the processor is in Stop mode provided an appropriate peripheral clock is available. This module is designed for low CPU overhead with DMA offloading of FIFO register accesses.
LPUART0-7	The Low Power Universal Asynchronous Receiver/Transmitter (LPUART) module provides asynchronous, serial communication capability with external devices. LPUART supports non-return-to-zero (NRZ) encoding format and IrDA-compatible infrared (low-speed) SIR format. The LPUART can continue operating while the processor is in Stop mode if an appropriate peripheral clock is available. This module is designed for low CPU overhead with DMA off loading of FIFO register accesses.
LPSPI0-3	The Low Power Serial Peripheral Interface (LPSPI) module implements an efficient interface to an SPI bus as a master and/or a slave. The LPSPI can continue operating while the processor is in Stop mode if an appropriate peripheral clock is available. This module is designed for low CPU overhead with DMA off loading of FIFO register accesses.
USBOTG1-2	The Universal System Bus On-The-Go (USBOTG) module is a USB 2.0-compliant implemention. The registers and data structures of this USB controller are based on the Enhanced Host Controller Interface Specification for Universal Serial Bus (EHCI). This module can act as a host, a device or an On-The-Go negotiable host/device on the USB bus.
USBPHY	The Universal System Bus Physical Layer (USBPHY) macrocell implements USB physical layer connecting to USB host/device systems at low-speed, full-speed, and high-speed. USB-PHY provides a standard UTMI interface for connection to the USB-OTG controller.

Table continues on the next page...

Table 2-4. Connectivity and communications modules (continued)

Module	Description
USBDCD	The USB Device Charging Detection (USBDCD) module works with the USB transceiver to detect whether the USB device is attached to a Charging Port, either a Dedicated Charging Port (DCP) or a Charging Downstream Port (CDP). System software coordinates the detection activities of the module and controls an off-chip integrated circuit that performs the battery charging.
HSIC-PHY	USB High-Speed Inter Chip Physical Layer (HSIC-PHY) is a complete digital IP designed to implement USB 2.0 HSIC connectivity interface.
CRC	The Cyclic Redundancy Check (CRC) module is a hardware CRC generator circuit using 16/32-bit shilft register. The CRC module supports error detection for all single-, double-, odd-, and most multi-bits errors, programmable initial seed value, and optional feature to transpose input data and CRC result via transpose register.

2.3.5 Timers

Table 2-5. Timer modules

Module	Description
LPTMR0-1	The Low Power Timer (LPTMR) module is a 16-bit timer which operates as real-time interrupt or pulse accumulator. This LPTMR module can remain functional when the chip is in low-power modes, provided the reference clock to this timer is active.
LPTPM0-7	The Timer/Pulse Width Modulation Module (TPM) is a multichannel timer module that supports input capture, output compare, and the generation of PWM signals. The counter, compare and capture registers are clocked by an asynchronous clock that can remain enabled in low-power modes.
LPIT0-1	Low Power Periodic Interrupt Timer (LPIT) is a multichannel timer module that can generate independent pretrigger and trigger outputs. These timer channels can operate individually or can be chained together. The pretrigger and trigger outputs can be used to trigger other modules on the device. The LPIT can also operate in low power modes.
TSTMR	The Time Stamp Timer (TSTMR) module is a free-running incremental counter that starts running after system reset deassertion and can be read at any time by the software for determining the software ticks. The TSTMR is a 64-bit clock cycle counter. It runs off the 1 MHz clock and resets on every system reset. The counter only stops when the clock to the TSTMR is disabled.
WDOG0-2	The Watchdog Timer (WDOG) module is an independent timer that is available for system use. It provides a safety feature to ensure that software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If the WDOG module is not serviced (refreshed) within a certain period, it resets the MCU.
EWM	The External Watchdog Monitor (EWM) module is designed to monitor external circuits, as well as the software flow. This provides a back-up mechanism to the internal WDOG that can reset the system. The EWM differs from the internal WDOG in that it does not reset the system. The EWM, if allowed to timeout, provides an independent trigger pin that when asserted resets or places an external circuit into a safe mode.

2.3.6 Multimedia

Table 2-6. Multimedia

Module	Description
GPU3D	i.MX 7ULP integrates the GC7000 Nano Ultra 3D Graphic Processing Unit (GPU). The GPU3D supports OpenGL ES2.0/1.1, Desktop OpenGL 2.1, OpenVG1.1, GLSL/ES Shading languages.
GPU2D	GC320 is a Composition Processing Core (CPC) GPU. The GPU2D supports user interface rendering and performs functions like blending, filtering, rotation, overlay, resizing, transparency, and other dynamic effects.
MIPI-DSI	The Mobile Industry Processor Interface (MIPI) Display Serial Interface Controller (DSI Controller) is responsible for serializing display data from the GPU. Data can come from either the GPU or the processor/DMA controller.
DSI PHY	The MIPI Display Serial Interface Physical Layer (DSI PHY) is a two-lane interface that supports up to 800 Mbps of data on each lane. DSI PHY includes a PLL which generates a dedicated clock for DSI purposes.
LCDIF	The enhanced LCD Interface (LCDIF) is a general-purpose display controller used to drive a wide range of display devices varying in size and capabilities. The LCDIF is used as a bridge between the DSI controller and the NIC0 crossbar.
VIU	The Video Input Unit (VIU) provides a parallel interface for digital video. The VIU accepts various types of digital video input on its parallel interface, decodes it and optionally performs processes such as down-scaling, and horizontal up-scaling.
SAI0-1	The Synchronous Audio Interface (SAI) module implements full-duplex serial interfaces with frame synchronization such as I2S, AC97, and CODEC/DSP interfaces.

2.3.7 Analog

Table 2-7. Analog modules

Module	Description
ADC0-1	Analog-to-Digital Converter (ADC) is a 12-bit resolution, successive approximation analog to digital converter. The ADC module supports up to 16 single-ended external analog inputs. The ADC supports only 12-bit non-differential and 13-bit (MSB is a sign bit) differential formatting option. The ADC can achieve 1 µs conversion rate.
DAC0-1	Digital-to-Analog Converter (DAC) is the 12-bit resolution digital-to-analog converters with programmable reference generator output. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator or ADC. The DAC is capable of achieving 1 microsecond conversion rate for high-speed signals and 2 µs conversion rate for low-speed signals.
CMP0-1	The Comparator (CMP) module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage (rail to rail operation).

2.3.8 Security

Table 2-8. Security

Module	Description
CAAM[1]	Cryptographic Acceleration and Assurance Module (CAAM) is a multi-function accelerator that supports the cryptographic functions common in many security protocols. This includes AES128, AES256, DES, 3DES, SHA1, SHA224, SHA256, and a random number generator with a true entropic seed. CAAM includes a DMA engine that is descriptor based to reduce processor-accelerator interaction. Hardware automatically clears key registers and specified Secure Memory areas when on-chip security monitor detects tampering. The Secure Memory provides secure storage of sensitive information in on-chip RAM.
LTC[1]	Low-power Trusted Cryptography (LTC) is an architecture that allows multiple cryptographic hardware accelerator engines to be instantiated and share common registers. This version of LTC supports 128-bit AES.
TRNG [1]	The True Random Number Generator (TRNG) module is used to generate high quality, cryptographically secure, random data. The TRNG module is capable of generating its own entropy using an integrated ring oscillator.
MMCAU[1]	Memory-Mapped Cryptographic Acceleration Unit (MMCAU) is an optimized security accelerator that supports the cryptographic functions common in many security protocols. This includes DES, 3DES, AES, MD5, SHA-1, SHA-256 algorithms via simple C calls to optimized security functions.
OTFAD[1]	The On-The-Fly AES Decryption (OTFAD) module provides an advanced hardware implementation that minimizes any incremental cycles of latency introduced by the decryption in the overall external memory access time. The OTFAD engine also includes complete hardware support for a standard AES key unwrap mechanism to decrypt a key BLOB data instruction containing the parameters needed for up to 4 unique AES contexts.
SNVS[2]	The Secure Non-Volatile Storage (SNVS) module is designed to safely hold security-related data such as cryptographic key, time counter, monotonic counter, and general-purpose security information. A part of the SNVS module belongs to the VBAT domain, which has its own dedicated power supply that is always on. This enables SNVS to keep this data valid and continue to increment the time counter when the power goes down in the rest of the device. The SNVS includes a Real- Time Clock (RTC) with roll-over protection and 32-bit alarm and a 64-bit monotonic counter with roll-over protection.

1. This chapter is available only in the i.MX 7ULP Security Reference Manual
2. The complete chapter including security features and its chip-specific information is available in the i.MX 7ULP Security Reference Manual

2.3.9 Debug interfaces

Table 2-9. Debug interfaces

Module	Description
DAP	Debug Access Port (DAP) provides debugger access to on-chip system resources via the SWJ-DP port. The DAP provides internal system access to A7 Debug Port, M4 Debug Port, System Bus, JTAG controller, and device Control and Status. The

Table continues on the next page...

Table 2-9. Debug interfaces (continued)

Module	Description
	DAP also enables system access to CoreSight debug subsystem through the APBIC port.
CTM	Cross Trigger Matrix (CTM) is a component of the Embedded Cross Trigger (ECT), which is key in the multi-core debug strategy. The CTM receives signals from various sources (i.e. cores and peripherals) and propagates or routes them to the different debug resources of the device. Those debug resources can include time stamping capability, real-time trace, triggers and debug interrupts.
ETF	The Embedded Trace FIFO (ETF) consists of a formatter, control, and the trace RAM. It is a configuration of the Trace Memory Controller (TMC). The ETF has a memory size of 4 KB. The ETF and associated memory should be connected in the system such that it will retain the information though a warm or cold reset of the system. This is to allow for debug information to be retained for debugging problems that may arise and cause a reset of the system.
ETR	The ETR is a trace sink that redirects the trace stream onto the AXI bus to external storage. It can utilize a single contiguous region or a scattered allocation of blocks for a circular buffer. Reading of the AXI based trace buffer can be done directly over AXI from a normal bus master. The ETR is a configuration option of the TMC as is the ETF.
FUNNEL	The Trace Funnel (FUNNEL) is used when there is more than one trace source. The Trace Funnel combines multiple trace stream onto a single ATB bus. The Trace Funnel includes an arbiter that determines the priority of the ATB inputs.
Replicator	The Trace Replicator (Replicator) enables two trace sinks (TPIU and TMC) to be wired together and receive ATB trace data from the same trace source. It takes incoming data from a single source and replicates it to two master ports.
TPIU	Trace Port Interface Unit (TPIU) acts as a bridge between on-chip trace data, ID distinguishable, and a TPA. It receives ATB trace data and sends it off chip via Arm's standard trace interface. The TPIU includes ATB interface, APB interface, Formatter, Asychronous FIFO, Register bank, Trace out serializer, and a Pattern generator.
SWO	Serial Wire Output (SWO) is a trace data drain that acts as bridge between the on-chip trace data to a data stream that is captured by the Trace Port Analyzer. It is a TPIU-like device that supports a limited subset of the full TPIU functionality for a simple debug solution.
TimeStamp Components	The timestamp components generate and distribute a consistent timestamp value for multiple processors and other blocks in a chip. The components used in i.MX 7ULP timestamp debug system are: one Timestamp Generator, one Timestamp Encoder, one Timestamp Replicator, one Timestamp Asynchronous Bridge (for A7), one Timestamp Synchronous Bridge (for M4), two Timestamp Decoders, and one Timestamp Interpolator (for A7).
JTAGC	Joint Test Action Group Controller (JTAGC) provides the means to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard.
SJC	The Secured JTAG Controller (SJC) is an authenticated debug module that implements a challenge/response mechanism using a standard cryptographic algorithm. This allows post production silicon debug without compromising security requirements. The SJC is connected in parallel with the JTAGC module, but it is only used for authenticated debug.

2.3.10 Clock modules

Table 2-10. Clock modules

Module	Description
SCG0-1	The System Clock Generation (SCG) module is responsible for clock generation and distribution across this device. Functions performed by the SCG include: clock reference selection, generation of clock used to derive processor, system, peripheral bus and external memory interface clocks; source selection for peripheral clocks; and, control of power saving clock gating mode.
PCC0-3	The Peripheral Clock Control (PCC) module is responsible for clock selection, optional division and clock gating mode for peripherals in their respected power domain.
RTC OSC[1]	The Real Time Clock Oscillator (RTC OSC) module provides the clock source for the Real-Time Clock module. The RTC OSC module, in conjunction with an external crystal, generates a 32.678 kHz reference clock for the RTC.
SYS OSC[1]	The System Oscillator (SYS OSC) module is a crystal oscillator. The SYS OSC, in conjunction with an external crystal or resonator, generates a reference clock for this device. It also supports optionally external input bypass clock from EXTAL signal directly.
Fixed-Freq PLL (PLL0)[2]	In addition to the main clock output, the Fixed-Frequency PLL also includes 4 Phase Fractional Dividers (PFDs) that can generate other clock frequencies. There is one instance of the Fixed-freq PLL (PLL0) which provides clocks for M4 core and buses and peripherals in the real-time domains.
Frac-N PLL (PLL1-3)[2]	The Fractional-N (Frac-N) PLL can generate an output clock of 528 MHz from a supported reference clock. In addition to the main clock output, this PLL also includes up to 4 Phase Fractional Dividers (PFDs) that can generate other clock frequencies. This PLL also supports tunable clock for audio applications.
FIRC[3]	The Fast Internal Reference Clock (FIRC) module is an internal oscillator that can generate a reference clock in the range from 48 MHz to 60 MHz. The FIRC output clock is used as a reference to the SCG module, and it is also used as a clock option to most on-chip modules.
SIRC[3]	The Slow Internal Reference Clock (SIRC) module is an internal oscillator that can generate a reference clock of 16 MHz. The SIRC output clock is used as a reference to the SCG module, and it is also used as a clock option to most on-chip modules.
IRC1K[3]	The Internal Reference Clock 1 kHz (IRC1K) module is an internal oscillator that can generate a reference clock of 1 kHz. The IRC1K clock is enabled in all modes of operation, including all low power modes.

1. See External clock sources
2. For details, see PLLs
3. See Internal clock sources

2.3.11 Power management

Table 2-11. Power management

Module	Description
Digital PMC	The Digital PMC module allows user software to control power modes of the chip and to optimize power consumption for the level of functionality needed. There are two instances of Digital PMC on this device, one for each main power domain.
Analog PMC	The Analog PMC consists of voltage/current references, core logic supply regulators, memory supply regulators, Back and Forward Biasing regulators, monitors and power switches, etc. There are two Analog PMC subsystems in i.MX 7ULP, one associated with the M4 power domain and the other with the A7 power domain.

2.3.12 System modules

Table 2-12. System control

Module	Description
MSMC	Multicore System Mode Controller (MSMC) is responsible for sequencing the system into and out of all low power Stop and Run modes. MSMC monitors events to trigger transitions between power modes, while controlling the power, clocks, and memories of the system to achieve the power consumption and functionality of that mode. **NOTE:** MSMC, CMC, and RMC are 3 separate but tightly coupled blocks. The memory map documented in MSMC chapter encompasses features related to all 3 blocks, being actually implemented into the CMC block alone. Therefore, there might be some references to CMC and RMC blocks throughout this document.
LLWU	The Low-Leakage Wake-Up Unit (LLWU) module allows user to select up to 32 external pin sources and up to 8 internal modules as a wake-up source from low-leakage power modes.
AWIC	The Asynchronous Wakeup Interrupt Controller (AWIC) module is capable of interrupt detection and wake up a processor when it is in low power mode.
IOMUXC0-1 and IOMUXC_DDR	The Input/Output Multiplexing Controller (IOMUXC) enables the chip to share one pad for multiple signals from different peripheral interfaces. This pad sharing mechanism is done by multiplexing the pad's input and output signals. The IOMUXC also controls the pads setting parameters and digital filter functions of the pad. In addition, the IOMUXC controls input multiplexing logic for input signals multiplexed at multiple locations.
WKPU	Wake Up (WKUP) module is capable of interrupt detection and wake a Cortex-A processor when it is in low power mode.
PCTLA-F	The Port Control (PCTL) module provides control for GPIO interrupt function. GPIO interrupt can be configured independently for each pin in the 32-bit port. There is one instance of the PCTL module for each port..
SIM	The System Integration Module (SIM) provides system control and chip configuration registers. The SIM includes the TSTMR module.

Table continues on the next page...

Table 2-12. System control (continued)

Module	Description
TRGMUX0-1	Peripheral Trigger Multiplexing (TRGMUX): The Trigger MUX module (TRGMUX) allows software to configure the trigger inputs for various peripherals.
OCOTP_CTRL[1]	The On-Chip One-Time-Programmable Controller (OCOTP_CTRL) module provides an interface for reading, programming and/or overriding identification and control information stored in on-chip fuse elements. The module supports electrically-programmable poly fuses. The OCOTP_CTRL also provides a set of volatile software-accessible signals which can be used for software control of hardware elements, not requiring non-volatility.
RGPIO2P0-1	The Rapid General-Purpose Input and Output with 2 Ports (RGPIO2P) is similar to the RGPIO module, except it has an AHB-lite port, in addition to the IPS port, for faster access. The RGPIO2P-0 in CM4 domain has two ports, AHB and IPS ports. The second RGPIO in CA7 domain only has a single port (IPS).

1. For complete chapter including security details, see the i.MX 7ULP Security Reference Manual

2.4 Ordering information

For details on orderable part numbers, see the ordering information table provided in the i.MX 7ULP Data Sheet.

Chapter 3
AIPS Interrupt Assignments

3.1 Interrupt assignments

3.1.1 M4 interrupt assignment

Table 3-1. M4 NVIC assignments

Vector	IRQ	Source Module	Source Description
0	—	Arm core	Initial Stack Pointer
1	—	Arm core	Initial Program Counter
2	—	Arm core	Non maskable interrupt (NMI)
3	—	Arm core	Hard Fault
4	—	Arm core	MemManage Fault
5	—	Arm core	Bus Fault
6	—	Arm core	Usage Fault
7	—	—	—
8	—	—	—
9	—	—	—
10	—	—	—
11	—	Arm core	Supervisor call (SVCall)
12	—	Arm core	Debug Monitor
13	—	Reserved	Reserved
14	—	Arm core	Pendable request for system service (PendableSrvReq)
15	—	Arm core	System tick timer (SysTick)
16	0	CTI0	Cross Trigger Interface
17	1	DMA0	DMA Channel 0, 4 Transfer Complete
18	2	DMA0	DMA Channel 1, 5 Transfer Complete

Table continues on the next page...

Table 3-1. M4 NVIC assignments (continued)

Vector	IRQ	Source Module	Source Description
19	3	DMA0	DMA Channel 2, 6 Transfer Complete
20	4	DMA0	DMA Channel 3, 7 Transfer Complete
21	5	DMA0	DMA Channel 8, 12 Transfer Complete
22	6	DMA0	DMA Channel 9, 13 Transfer Complete
23	7	DMA0	DMA Channel 10, 14 Transfer Complete
24	8	DMA0	DMA Channel 11, 15 Transfer Complete
25	9	DMA0	DMA Channel 16, 20 Transfer Complete
26	10	DMA0	DMA Channel 17, 21 Transfer Complete
27	11	DMA0	DMA Channel 18, 22 Transfer Complete
28	12	DMA0	DMA Channel 19, 23 Transfer Complete
29	13	DMA0	DMA Channel 24, 28 Transfer Complete
30	14	DMA0	DMA Channel 25, 29 Transfer Complete
31	15	DMA0	DMA Channel 26, 30 Transfer Complete
32	16	DMA0	DMA Channel 27, 31 Transfer Complete
33	17	DMA0	DMA Error Interrupt - All Channels
34	18	MCM0	MCM Interrupt
35	19	EWM	External Watchdog Monitor Interrupt
36	20	LLWU0	Low Leakage Wake Up
37	21	SIM	System Integation Module
38	22	MU_A	Messaging Unit - Side A
39	23	Reserved	Reserved
40	24	Software	Software Interrupt
41	25	Software	Software Interrupt
42	26	WDOG0	Watchdog Interrupt
43	27	SCG0	System Clock Generator 0
44	28	QSPI	Quad Serial Peripheral Interface
45	29	LTC[1]	Low Power Trusted Cryptography
46	30	XRDC	Extended Domain Resource Controller

Table continues on the next page...

Table 3-1. M4 NVIC assignments (continued)

Vector	IRQ	Source Module	Source Description
47	31	SNVS	Secure Non-Volatile Storage Consolidated Interrupt
48	32	TRNG0[1]	Random Number Generator
49	33	LPIT0	Low Power Periodic Interrupt Timer
50	34	PMC0	Power Management Control interrupts for M4 domain
51	35	CMC0	Core Mode Controller interrupts for M4 domain
52	36	LPTMR0	Low Power Timer
53	37	LPTMR1	Low Power Timer
54	38	TPM0	Timer PWM module
55	39	TPM1	Timer PWM module
56	40	TPM2	Timer PWM module
57	41	TPM3	Timer PWM module
58	42	FlexIO0	Flexible IO
59	43	LPI2C0	Low Power Inter-Integrated Circuit module
60	44	LPI2C1	Low Power Inter-Integrated Circuit module
61	45	LPI2C2	Low Power Inter-Integrated Circuit module
62	46	LPI2C3	Low Power Inter-Integrated Circuit module
63	47	SAI0	Serial Audio Interface
64	48	SAI1	Serial Audio Interface 1
65	49	LPSPI0	Low Power Serial Peripheral Interface
66	50	LPSPI1	Low Power Serial Peripheral Interface
67	51	LPUART0	Low Power UART
68	52	LPUART1	Low Power UART
69	53	LPUART2	Low Power UART
70	54	LPUART3	Low Power UART
72	56	PCTLA	Port A pin interrupt
73	57	PCTLB	Port B pin interrupt
74	58	ADC0	Analog to Digital Convertor
75	59	ADC1	Analog to Digital Convertor
76	60	CMP0	Comparator
77	61	CMP1	Comparator
78	62	DAC0	Digital to Analog Convertor
79	63	DAC1	Digital to Analog Convertor

Table continues on the next page...

Table 3-1. M4 NVIC assignments (continued)

Vector	IRQ	Source Module	Source Description
80	64	WDOG1	Watchdog Interrupt from A7 subsystem
81	65	USB0	USB 0 Interrupt from A7 subsystem
82	66	USB1	USB 1 Interrupt from A7 subsystem
83	67	—	
84	68	WDOG2	Watchdog Interrupt from A7 subsystem
85	69	USB PHY	USB PHY (used in conjunction with USBOTG1)
86	70	CMC1	A7 resets
87	71	-	A7 Subsystem
88	72	-	X Domain DMA Interrupt
89	73	-	X Domain Master Interrupt
90	74	-	A7 Subsystem
91	75	Graphics Processing Unit 3D	GPU3D
92	76	Graphics Processing Unit 2D	GPU2D
93	77	Reserved	—
94	78		
95	79		
96	80		
97	81		
98	82		
99	83		
100	84		
101	85		
102	86		
103	87		
104	88		
105	89		
106	90		
107	91		
108	92		
109	93		
110	94		
111	95		
112	96		
113	97		
114	98		
115	99		
116	100		

Table continues on the next page...

Table 3-1. M4 NVIC assignments (continued)

Vector	IRQ	Source Module	Source Description
117	101		
118	102		
119	103		
120	104		
121	105		
122	106		
123	107		
124	108		
125	109		
126	110		
127	111		
128	112		
129	113		
130	114		
131	115		
132	116		
133	117		
134	118		
135	119		
136	120		
137	121		
138	122		
139	123		
140	124		
141	125		
142	126		
143	127		

1. The complete chapter is available only in the i.MX 7ULP Security Reference Manual

3.1.2 A7 interrupt assignment

The following table shows interrupt sources mapped into A7 GIC.

Table 3-2. A7 interrupt assignments

IRQ	A7 GIC Interrupt Assignments	
0	DMA1-0,16	DMA1 channel 0, 16 transfer complete
1	DMA1-1,17	DMA1 channel 1, 17 transfer complete
2	DMA1-2,18	DMA1 channel 2, 18 transfer complete

Table continues on the next page...

Table 3-2. A7 interrupt assignments (continued)

IRQ	A7 GIC Interrupt Assignments	
3	DMA1-3,19	DMA1 channel 3, 19 transfer complete
4	DMA1-4,20	DMA1 channel 4, 20 transfer complete
5	DMA1-5,21	DMA1 channel 5, 21 transfer complete
6	DMA1-6,22	DMA1 channel 6, 22 transfer complete
7	DMA1-7,23	DMA1 channel 7, 23 transfer complete
8	DMA1-8,24	DMA1 channel 8, 24 transfer complete
9	DMA1-9,25	DMA1 channel 9, 25 transfer complete
10	DMA1-10,26	DMA1 channel 10, 26 transfer complete
11	DMA1-11,27	DMA1 channel 11, 27 transfer complete
12	DMA1-12,28	DMA1 channel 12, 28 transfer complete
13	DMA1-13,29	DMA1 channel 13, 29 transfer complete
14	DMA1-14,30	DMA1 channel 14, 30 transfer complete
15	DMA1-15,31	DMA1 channel 15, 31 transfer complete
16	DMA1_error	DMA1 error interrupt channels 0-31
17	A7 CTI	A7 Cross Trigger Interface
18	—	—
19	MU_B	MU B side
20	—	—
21	TPM4	Timer and PWM Module 4
22	TPM5	Timer and PWM Module 5
23	TPM6	Timer and PWM Module 6
24	TPM7	Timer and PWM Module 7
25	LPIT1	Lower Power Periodic Timer 1
26	FlexIO1	Flexible Input/Output
27	—	—
28	LPSPI2	Low Power SPI 2
29	LPSPI3	Low Power SPI 3
30	LPUART4	Low Power UART 4
31	LPUART5	Low Power UART 5
32	LPUART6	Low Power UART 6
33	LPUART7	Low Power UART 7
34	LPI2C4	Low Power I2C 4
35	LPI2C5	Low Power I2C 5
36	LPI2C6	Low Power I2C 6
37	LPI2C7	Low Power I2C 7
38	—	—
39	USB PHY	USB PHY (used in conjunction with USBOTG1 controller)
40	USBOTG1	USB On-The-Go 1
41	USBOTG2	USB On-The-Go 2

Table continues on the next page...

Table 3-2. A7 interrupt assignments (continued)

IRQ	A7 GIC Interrupt Assignments	
42	uSDHC0	SD Host Controller 0
43	uSDHC1	SD Host Controller 1
44	GPU-3D	3D Graphic Processing Unit (GC7000 core)
45	VIU	Video-In Unit
46	DSI	MIPI Display Serial Interface
47	LCDIF	LCD Interface
48	PCTLC	Port Interrupt Control Module C
49	PCTLD	Port Interrupt Control Module D
50	PCTLE	Port Interrupt Control Module E
51	PCTLF	Port Interrupt Control Module F
52	GPU-2D	2D Graphic Processing Unit (GC320 core)
53	—	—
54	CAAM	Cryptographic Acceleration and Assurance Module
55	WDOG1	Watchdog 1
56	WDOG2	Watchdog 2
57	PMC1	Power Management Control 1
58	SCG1	System Clock Generation 1
59	CMC1	Core Mode Controller interrupts from A7
60	Software	Software interrupt
61	—	—
62	—	—
63	—	—
64	QSPI	QuadSPI controller
65	SAI0	Synchronous Audio Interface 0
66	SAI1	Synchronous Audio Interface 1
67	PCTLA	Port Interrupt Control Module A
68	PCTLB	Port Interrupt Control Module B
69	Reserved	—
70	M4.XRDC	Extended Domain Resource Controller
71	LPSPI0	Low Power SPI 0
72	LPSPI1	Low Power SPI 1
73	LPI2C0	Low Power I2C 0
74	LPI2C1	Low Power I2C 1
75	LPI2C2	Low Power I2C 2
76	LPI2C3	Low Power I2C 3
77	LPIT0	Low Power PIT 0
78	M4 X-Domain-DMA-Int	Consolidated M4 DMA interrupts to the A7 domain
79	M4 X-Domain-Master-Int	Consolidated M4 IRQs to the A7 domain
80	LPUART0	Low Power UART 0

Table continues on the next page...

Table 3-2. A7 interrupt assignments (continued)

IRQ	A7 GIC Interrupt Assignments	
81	LPUART1	Low Power UART 1
82	LPUART2	Low Power UART 2
83	LPUART3	Low Power UART 3
84	—	—
85	—	—
86	—	—
87	—	—
88	SNVS	Secured Non-Volatile Storage
89	—	—
90	MU_B NMI	NMI interrupt from M4 side via Messaging Unit
91	TSTMR	Time Stamp Timer
92	—	—
93	—	—
94	—	—
95	—	—
96	—	—
97	—	—
98	—	—
99	—	—
100	—	—
101	—	—
102	—	—
103	—	—
104	—	—
105	—	—
106	—	—
107	—	—
108	—	—
109	—	—
110	—	—
111	—	—
112	—	—
113	—	—
114	—	—
115	—	—
116	—	—
117	—	—
118	—	—
119	—	—

Table continues on the next page...

Table 3-2. A7 interrupt assignments (continued)

IRQ	A7 GIC Interrupt Assignments	
120	—	—
121	—	—
122	—	—
123	—	—
124	—	—
125	—	—
126	—	—
127	—	—

Chapter 4
Multicore Architecture

4.1 Overview

i.MX 7ULP design allows flexibility for peripherals and memories partition in multicore environment; both cores will have access to all peripherals and memory regions unless a specific design calls for some isolated peripherals or dedicated memory. The resource domain control (XRDC) concept provides a framework for supporting isolation and safe sharing of chip resources in a multicore architecture. See Figure 4-1. Chip resources including cores, bus masters, peripherals and memory regions can be assigned to processing domains for the purpose of enforcing robust operational environments. Operating system resources are allocated to a domain and access to those resources are enforced, to prevent accidental collisions due to incorrect memory map references, to protect proprietary software resources from unauthorized sources, or to prevent manipulation of hardware shared resources from malicious attacks.

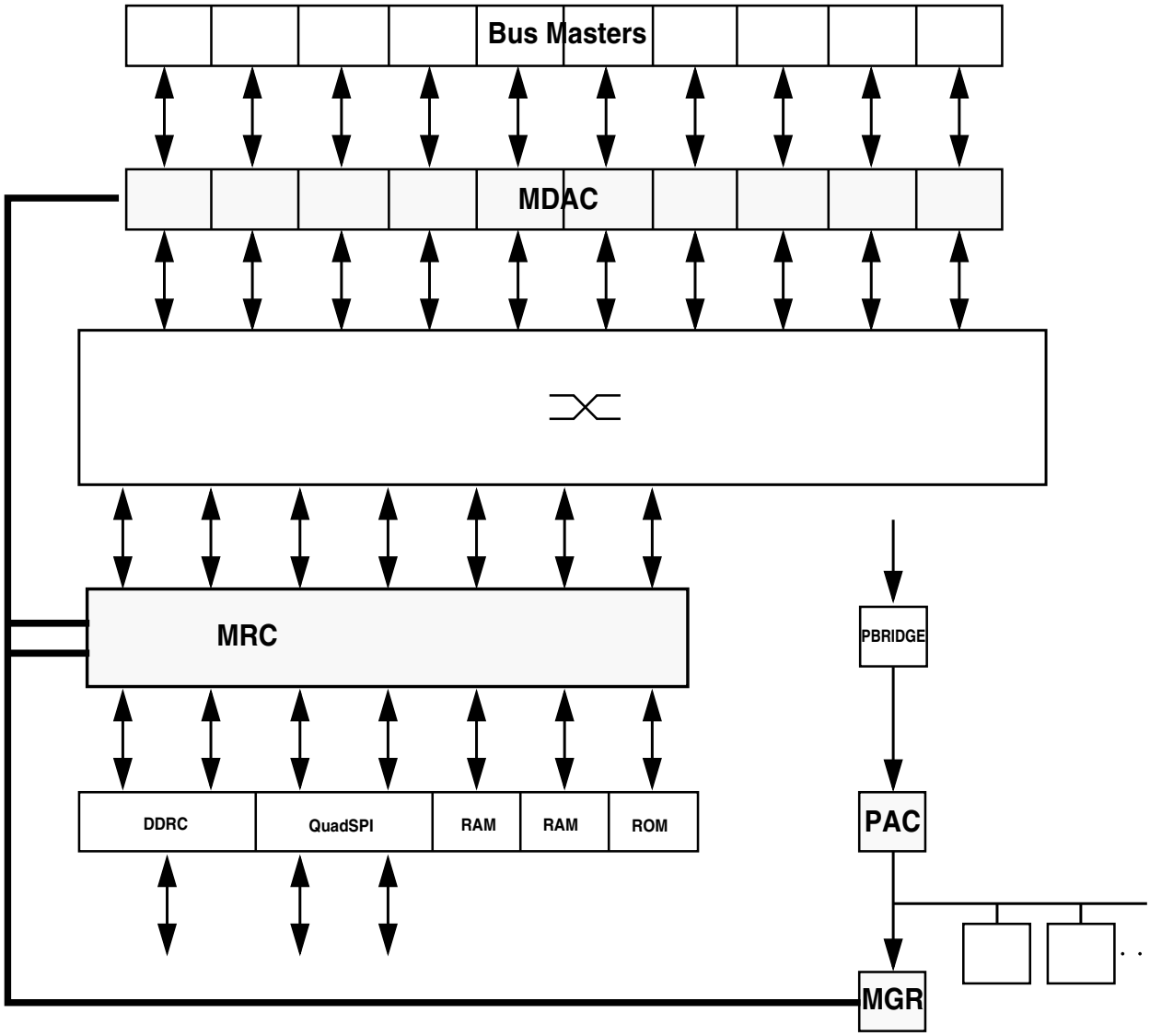


Figure 4-1. Simplified Extended RDC Block Diagram

4.2 Resource sharing and isolation

4.2.1 Extended Resource Domain Controller

The Extended Resource Domain Controller (XRDC) provides an integrated, scalable architectural framework for access control, system memory protection and peripheral isolation. It allows software to assign chip resources including processor cores, non-core bus masters, memory regions and slave peripherals to processing domains to support enforcement of robust operational environments.

- First, each bus mastering resource has an associated Master Domain Assignment Controller (MDAC) for assigning a domain identifier (DID) attribute to each bus transaction.
- Second, each peripheral and memory has an associated access control submodule that enforces access control policies based on the DID and privilege level of the transaction.
- Finally, all accesses throughout the device are monitored concurrently to determine the validity of each and every access. If a reference from a given domain has sufficient access rights, it is allowed to continue, otherwise the access is denied and a bus error is generated.

The access control scheme defined by the XRDC supports a 4-level model, combining the traditional privileged (also known as supervisor) and user modes with an additional signal defining the secure, nonsecure attributes of each memory reference. The result is a 4-level hierarchical access control mechanism, where:

SecurePriv(ileged) > SecureUser > NonsecurePriv(ileged) > NonsecureUser

with different access control policies based on read and write references. Combined with the user/privileged and secure/nonsecure attributes, a domainID is associated with every system bus transaction and forms the hardware basis for implementation of XRDC's access control mechanisms.

Access to shared memory regions and slave peripherals can be dynamically controlled with the optional inclusion of a hardware semaphore. If a hardware semaphore is enabled for a given address space or peripheral, then writes to the targeted address space are only allowed if the requesting domain owns the semaphore. This capability allows the access control policy for a given resource to be dynamically revised, based on hardware semaphore ownership.

The XRDC implementation is distributed across multiple submodules instantiated throughout the device. The XRDC submodules include:

- **XRDC_MGR**: The Manager (MGR) coordinates all programming model reads and writes.
- **XRDC_MDAC**: The Master Domain Assignment Controller (MDAC) handles resource assignments and generation of the domain identifiers based on pre-programmed Master Domain Assignment (MDA) registers.

- **XRDC_MRC**: The Memory Region Controller (MRC) implements the access controls for slave memories based on the pre-programmed Memory Region Descriptor (MRGD) registers.
- **XRDC_PAC**: The Peripheral Access Controller (PAC) implements the access controls for slave peripherals based on the pre-programmed Peripheral Domain Access Control (PDAC) registers.

4.2.2 Shared memory protection

Both processors and non-core bus masters share common memories. Because the memories are used for the exchange of data between the cores or between a processor and a bus master, unless additional protection mechanisms are provided, untrusted programs on one processor could read and write sensitive code or data placed in the shared memory region accessed by the other processor. This would fundamentally compromise the security of the device. To combat this, restricted memory access control is enforced by the XRDC.

The MRC provides domain-based, hardware access control for all system bus references targeted at non-peripheral memory spaces. Using pre-programmed region descriptors which define memory spaces and their associated access rights per domain identifier, the MRC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer. Memory references that have sufficient access control rights are allowed to complete, while references that are not mapped to any region descriptor or have insufficient rights are terminated with an access error response.

For each access, the MRC hardware performs a two-step evaluation process. First, the access is compared to all memory region descriptors to determine hit/miss status, and then, using the DID associated with the reference plus the address attributes {write, nonsecure, privileged}, the access right are examined.

4.2.3 Shared peripheral isolation

For shared peripherals, the PAC performs an access control evaluation similar to that of the memory region controller, but using dedicated memory region descriptors per slave peripheral slot. The presence of a one and only one memory region descriptor per slave peripheral address space slot in the PAC simplifies the required functionality compared to the MRC.

Unimplemented peripheral slots and unimplemented domain identifiers default to all-zero and provide no access rights.

Interrupts and DMA requests are also routed to the appropriate core and DMA engine associated with a core. In addition, the XRDC handles peripheral interface pin assignments and configuration of read/write access to control IO pins.

4.3 System control for multicore operation

System controls on i.MX 7ULP must follow procedures described by individual peripherals below for multi-core operation. The device follows the concept of master core (M4) and slave core (A7). There are some peripherals that are multicore aware allowing both processors to access them. These peripherals typically contain controls that need dynamic changes throughout the applications. If these controls affect other peripherals or pins then the XRDC subsystem will pass ownership information to these multicore aware peripherals.

On the other hand, there are some peripherals that are typically configured once in the application. These peripherals are not multi-core aware and are expected to be configured by the master core.

4.3.1 Power modes, transition, wake-up, etc.

The Multicore System Mode Control (MSMC) module is used to control the power mode settings for each processor's domain. MSMC consists of SMC0 and SMC1, which are associated with M4 and A7 domains, respectively. The MSMC module works in conjunction with PMCs, RMCs and LLWU to faciliate power mode transitions.

4.3.2 System clock management

The System Clock Generation (SCG) modules is used to configure clock sources and divider settings for their respective domain. The PCC modules contain clock configuration settings for individual peripheral clock source options and clock gating. There is one PCC instance designated for peripherals on the same IP bus. SCG and PCC modules are not multicore aware peripherals.

4.3.3 Pin allocation and port control

The IOMUXC peripheral will be used to configure pin settings including pin multiplexing assignments and pad characteristics. IOMUXC is a multi-core aware peripheral. IOMUXC will only allow the core has its ownership to manipulate configuration.

Similar to IOMUXC, the RGPIO2P peripheral uses XRDC ownership data to allow or disallow changes of state to a respective pin's direction or state attempted per core.

4.3.4 System Reset and Initial Boot

A Power-on Reset or System Reset event resets the entire chip except for certain components that are not affected by a specific System Reset source.

Coming out of a system reset, the M4 processor always boots first using the normal reset handling mechanism in the boot ROM. The M4 then releases the A7 sequentially depending on the boot mode setting. More details about reset and system boot are described in the Reset and Boot chapter.

Chapter 5
Arm Cortex A7 Platform (Cortex-A7)

5.1 Chip-specific Arm CA7 information

Table 5-1. Reference links to related information

Topic	Related module	Reference
Full description	Arm CA7	Arm CA7
System memory map		System memory map

5.2 WFE/SEV instruction support

WFE/SEV instructions are not supported in i.MX 7ULP, so any reference of these instructions throughout this document, must be ignored.

5.3 A7 processor

5.3.1 Overview

The application processing domain is built around a Cortex A7 processor optimized to run nominally at 500 MHz, supported by a large L2 cache and an LPDDR memory interface. A functional block diagram of the A7 complex is shown in the following figure.

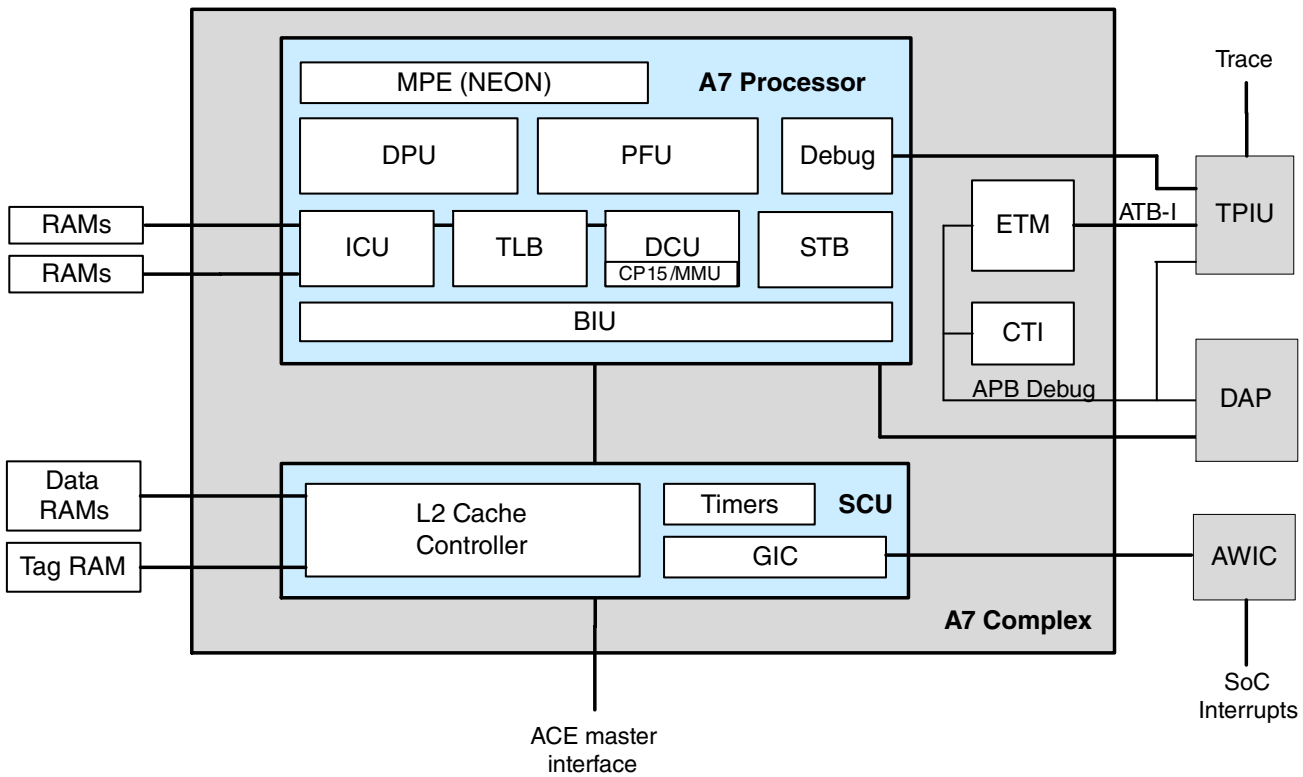


Figure 5-1. A7 complex functional diagram

5.3.2 Implementation information

The A7 complex consists of a Arm Cortex-A7 MPCore processor, which includes a Neon co-processor, FPU, and 32 KB instruction and 32 KB data L1 caches. The Snoop Control Unit (SCU) of the CA7 complex includes a unified 256 KB L2 cache, Generic Timer, and Generic Interrupt Controller (GIC). In addition, it also includes various components composing the Arm CoreSight debug/Trace system, such as ETM and CTI.

The Arm A7 Complex includes the following features:

- Arm Cortex-A7 processor running at 500 MHz in normal operation and up to 800 MHz overdrive mode
- Harvard Level 1 memory system with 32 KB Instruction Cache and 32 KB Data Cache with a Memory Management Unit (MMU).
- Level 2 memory system with 256 KB L2 unified cache.
- Media Processing Engine (MPE) with NEON technology.
- Generic Interrupt Controller (GIC) supports up to 128 external interrupt vectors.
- Generic Timer for scheduling events and trigger interrupts
- APB debug interface that supports integer processor clock ratios
- Embedded Trace Macrocell for instruction trace

- Cross Trigger Interface
- The A7 complex will not support State Retention Power Gating (SRPG).
- The A7 complex contain synchronizers for signals into and out of the complex where synchronization is required. All components in the A7 complex operate synchronously with the A7 core but may be at an integer clock ratio, including 1:1.

5.3.3 A7 Core

The A7 core has following features and functional descriptions:

- Full implementation of the Armv7-A architecture instruction set with the architecture extensions
- The Data Processing Unit (DPU) holds most of the program-visible state of the processor, such as general-purpose registers, status registers and control registers. It decodes and executes instructions, operating on data held in the registers in accordance with the Arm Architecture. Instructions are fed to the DPU from the Prefetch Unit (PFU). The DPU executes instructions that require data to be transferred to or from the memory system by interfacing to the Data Cache Unit (DCU), which manages all load and store operations.
- The Instruction Cache Unit (ICU) contains the Instruction Cache controller and its associated linefill buffer. The Cortex-A7 MPCore ICache is two-way set associative and uses Virtually Indexed Physically Tagged (VIPT) cache-lines holding up to 8 Arm or Thumb 32-bit instructions or up to 16 Thumb 16-bit instructions.
- The Prefetch Unit (PFU) obtains instructions from the instruction cache or from external memory and predicts the outcome of branches in the instruction stream, then passes the instructions to the DPU for processing. In any given cycle, up to a maximum of four instructions can be fetched and two can be passed to the DPU.
- The Data Cache Unit (DCU) consists of the Level 1 data cache controller, the load/store pipeline that interfaces with the DPU and main TLB, the system coprocessor controller (CP15), and interface to receive coherency requests from the Snoop Control Unit (SCU). The DCU also contains a combined local and global exclusive monitor.
- The system control coprocessor, CP15, provides configuration and control of the memory system and its associated functionality. That includes Memory Management Unit (MMU) configuration and management.

- The Store Buffer (STB) holds store operations when they have left the load/store pipeline and have been committed by the DPU. From the STB, a store can request access to the cache RAMs in the DCU, request the BIU to initiate linefills, or request the BIU to write the data out on the external write channel. External data writes are through the SCU.
- The Bus Interface Unit (BIU) contains the SCU interface and buffers to decouple the interface from the cache and STB. The BIU interface and the SCU always operate at the processor frequency.

5.3.4 Memory Management Unit (MMU)

The Cortex-A7 MPCore processor implements the Extended VMSAv7 MMU, which includes the Armv7-A Virtual Memory System Architecture (VMSA), the Security Extensions, the Large Physical Address Extensions (LPAE), and the Virtualization Extensions.

The Extended VMSAv7 MMU controls address translation, access permissions, and memory attributes determination and checking, for memory accesses.

The MMU controls table walk hardware that accesses translation tables in main memory. The MMU works with the L1 and L2 memory system to translate virtual addresses to physical addresses. The MMU enables fine-grained memory system control through a set of virtual-to-physical address mappings and memory attributes held in the Translation Look-aside Buffers (TLBs).

The MMU features in each processor of the multiprocessor device include the following:

- 10-entry fully-associative micro instruction TLB.
- 10-entry fully-associative micro data TLB.
- 2-way set-associative 256-entry unified main TLB.
- 2-way set-associative 64-entry walk cache.
- 2-way set-associative 64-entry IPA cache.
- The TLB entries include global and application specific identifiers to prevent context switch TLB flushes.
- Virtual Machine Identifier (VMID) to prevent TLB flushes on virtual machine switches by the hypervisor.

5.3.5 A7 Level 1 Memory

The L1 memory system consists of separate instruction and data caches of 32KB each.

The L1 memory system has a store buffer that has four 64-bit slots with data merging capability. It handles writes to Device, Strongly-ordered, Cacheable and Non-cacheable memory.

The L1 instruction memory system has the following features:

- Instruction side cache line length of 32-bytes.
- Virtually indexed and physically tagged instruction cache.
- Pseudo random cache replacement policy.
- 2-way set-associative instruction cache.
- Support for four sizes of memory page.
- Export of memory attributes for external memory systems.
- Support for Security Extensions.
- Can be disabled independently, using the system control coprocessor.
- On a cache miss, critical word first filling of the cache is performed

The L1 data memory system has the following features:

- Data side cache line length of 64-bytes.
- Physically indexed and physically tagged data cache.
- Pseudo random cache replacement policy.
- 4-way set-associative data cache.
- Two 32-byte linefill buffers and one 64-byte eviction buffer.
- A 4-entry, 64-bit merging store buffer.
- Can be disabled independently, using the system control coprocessor.
- On a cache miss, critical word first filling of the cache is performed.
- Virtually indexed and physically tagged instruction cache.

5.3.6 A7 Level 2 Memory

The L2 memory system includes 256KB unified cache. It provides:

- Data is only allocated to the L2 cache when evicted from the L1 memory system, not when first fetched from the system. The L1 cache can prefetch data from the system without data being evicted from the L2 cache.
- Instructions are allocated to the L2 cache when fetched from the system and can be invalidated from the L2 during maintenance operations.
- The L2 cache is 8-way set associative. The L2 cache tags are looked up in parallel with the SCU duplicate tags. If both the L2 tag and SCU duplicate tag hit, the L2 tag hit takes priority.

- Additionally, speculative requests to the system from the SCU are not made until the system checks the L2 cache.
- L2 RAMs are invalidated automatically at reset unless the L2 cache reset was disabled.

The L2 memory system interfaces with an AMBA AXI Coherency Extension (ACE) interconnect on a 128-bit wide bus.

5.3.7 Generic Interrupt Controller (GIC)

The integrated GIC collates and arbitrates from a large number of interrupt sources. It provides:

- 128 external interrupt vectors.
- Masking of interrupts.
- Prioritization of interrupts.
- Distribution of the interrupts to the target processors.
- Tracking the status of interrupts.
- Generation of interrupts by software.
- Support for Security Extensions.
- Support for Virtualization Extensions.

5.3.8 Generic Timer

The Generic Timer can schedule events and trigger interrupts based on an incrementing counter value. It provides:

- Generation of timer events as interrupt outputs.
- Generation of event streams.
- Support for Virtualization Extensions.

5.3.9 Debug

The processor debug unit assists in debugging software running on the processor. The debug unit enables users to:

- Stop program execution.
- Examine and alter process and coprocessor state.
- Examine and alter memory and input/output peripheral state.

• Restart the processor.

The processor supports six breakpoints, four watchpoints, and a standard Debug Communications Channel (DCC). Four of the breakpoints match only to virtual address and the other two match against either virtual address or context ID, or Virtual Machine Identifier (VMID). All the watchpoints can be linked to two breakpoints to enable a memory request to be trapped in a given process context.

The ETM module provides instruction trace. It generates trace data in the form of packets and transfer them through an Advanced Trace Bus (ATB) to the Trace Port Interface Unit (TPIU). In addition, A7 debug subsystem includes the CTI block quick takes cross-trigger event inputs from A7 complex or from its own configuration registers, and generates outputs on the appropriate channels.

Chapter 6
Arm Cortex M4 Platform (Cortex-M4)

6.1 Chip-specific Arm M4 information

Table 6-1. Reference links to related information

Topic	Related module	Reference
Full description	Arm CM4	Arm CM4
System memory map		System memory map

6.2 WFE/SEV instruction support

WFE/SEV instructions are not supported in i.MX 7ULP, so any reference of these instructions throughout this document, must be ignored.

6.3 Overview

This block details the Arm Cortex-M4 core. The Cortex-M4 implements the Armv7-ME instruction set architecture (ISA). It provides compatibility with Cortex-M3 and adds significant new capabilities with DSP and SIMD extensions. The basic multiply-accumulate instructions support operations up to 32 x 32 + 64. Cortex-M4 also includes a single-precision floating-point unit (FPU), which includes an extension register file of thirty-two 32-bit floating-point data registers. Cortex-M4 complex includes the FPU and two 32-bit system bus interfaces. The Cortex-M4 implementation includes two tightly-coupled local memories and two cache memories connected to these bus interfaces although the device implementation connects to the 64-bit system bus interconnect and supports a 32-byte cache line size.

- L1 2-way set-associative 8 KB Instruction/Data cache with 32B line size length

6.3.1 Cortex-M4 Block Diagram

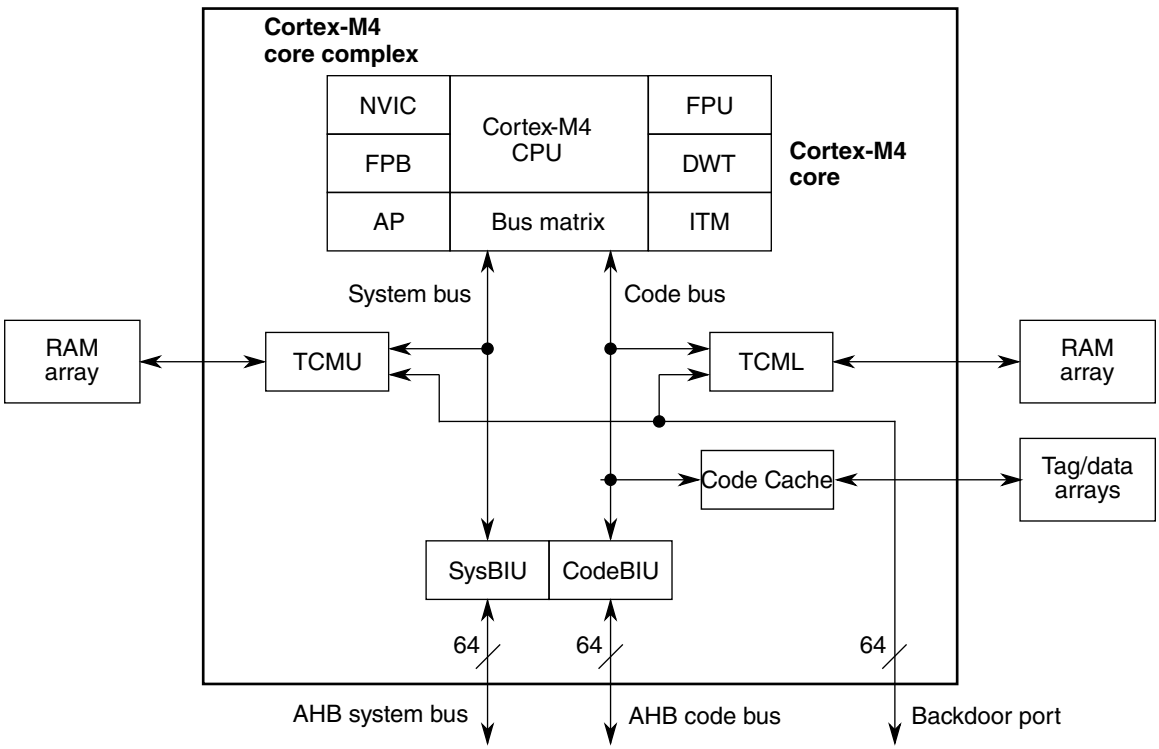


Figure 6-1. Cortex-M4 Block Diagram

Cortex-M4 core features a single issue, three stage pipeline microarchitecture. A high-level spatial pipeline block diagram of the CPU is shown below. The stages of the pipeline include:

- Fe - Instruction fetch stage where data is returned from instruction memory
- De - Instruction decode stage, generation of Load/Store Unit (LSU) address using forwarded register ports and immediate offset of LR register branch forwarding
- Ex - Instruction execute stage, single pipeline with multi-cycle stalls, LSU address/data pipelining to AHB interface, multiply/divide and ALU with branch result

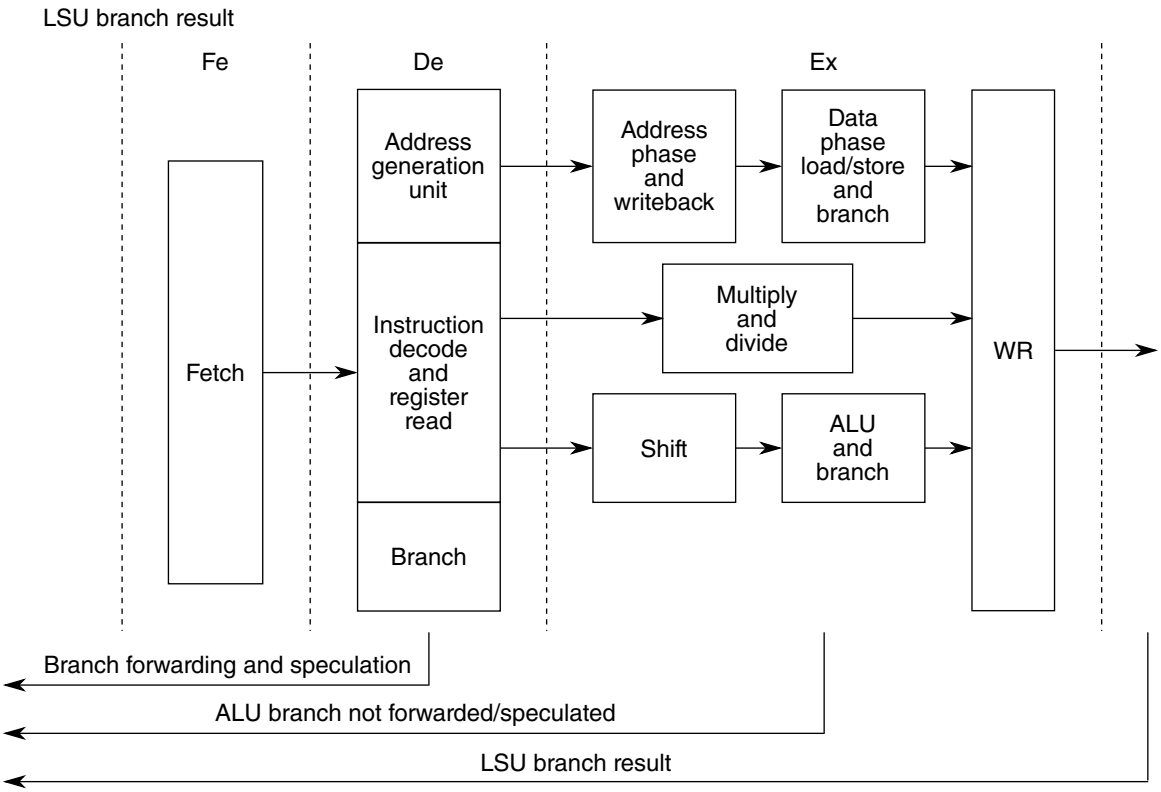


Figure 6-2. Cortex-M4 Pipeline Block Diagram

6.3.2 Integration Layer Block Diagram

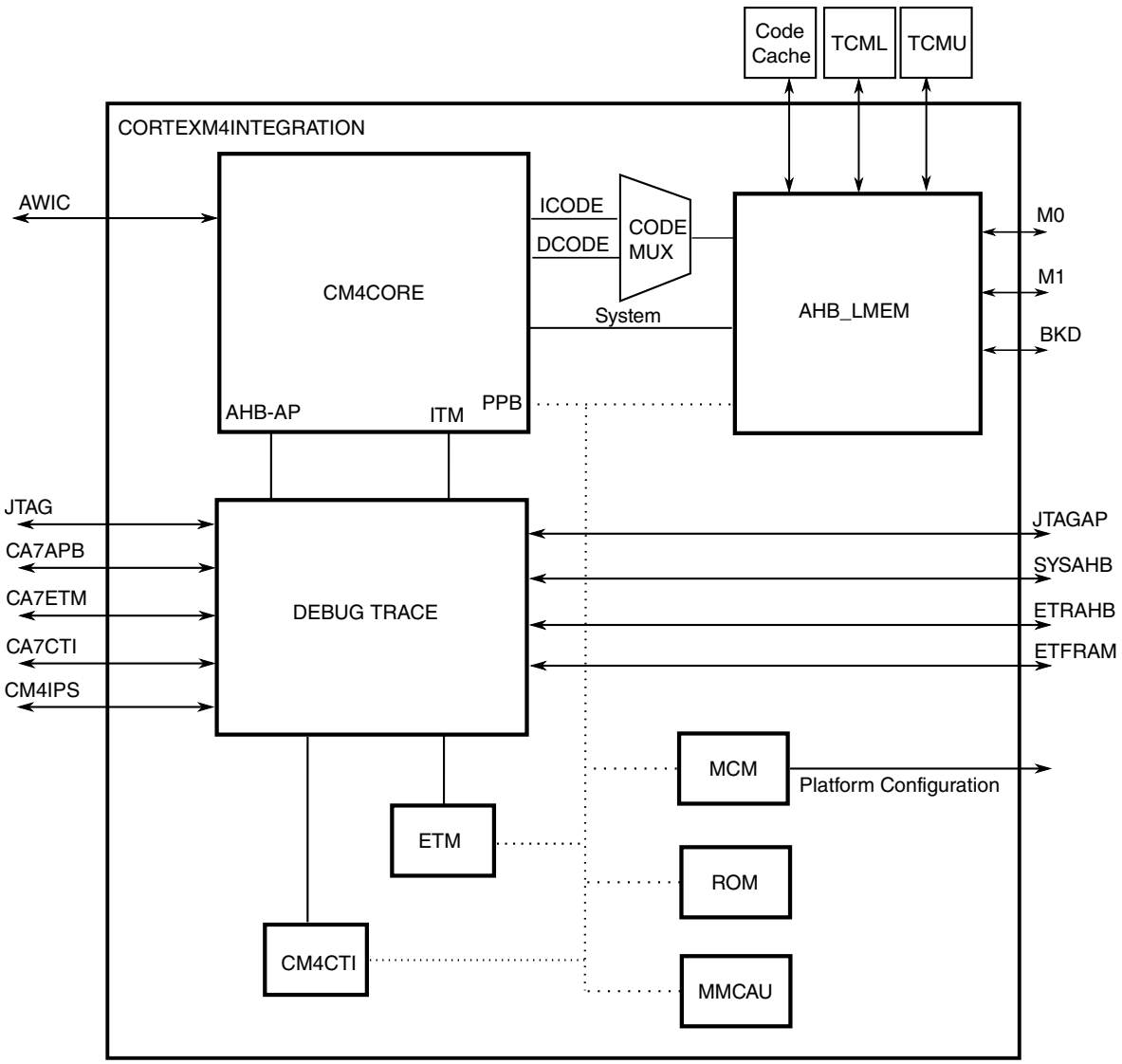


Figure 6-3. Cortex M4 Integration Layer Block Diagram

The Cortex M4 Integration Layer includes the following blocks:

- CM4CORE - Cortex M4 Core
- CODE MUX - Muxing logic for I- and D-Code Busses
- AHB_LMEM - AHB bus Local Memory (Include TCMC and Cache Controller)
- SWJDP - Serial-Wire/JTAG Debug Port
- MDM - Miscellaneous Debug Module
- TPIU - Trace Port Interface Unit
- ETM - Embedded Trace Macrocell
- FUNNEL - Coresight Funnel
- UPSZ - ATB Upsizer
- REP - ATB Replicator

- MCM - Miscellaneous Control Module
- ROM - ROM Table

6.4 Cortex-M4 Platform Features

6.4.1 Core Module Features

The following are the Cortex-M4 core module features:

- Cortex-M4 Core
  - 3-stage pipelined processor supporting Arm v7-M ISA with DSP extensions
  - Single precision FPU supporting the variant of the Armv7-M Floating-Point Extension (FPv4-SP)
  - MPU - Memory Protection Unit supporting up to 8 regions
  - Modified Harvard connections to AHB_LMEM and the crossbar switch
  - NVIC - Nested Vectored Interrupt Controller with 128 interrupt connections and 16 levels
  - SWJ-DP - Serial-Wire/JTAG Debug Port
  - TPIU - Trace Port Interface Unit
  - 4 KB ETF
- AHB LMEM
  - TCMC - Tightly Coupled Memory Controller with support for 256 KB RAM
  - One 8 KB of combined data/instruction cache to minimize the performance impact of memory access latencies.

6.5 Cortex-M4 Instruction Fetches on the System Bus

The Cortex-M4 processors implement multiple 32-bit bus interfaces that support a Harvard memory architecture. Specifically, the cores provide a modified Harvard connection with 2-cycle pipelined AMBA-AHB code and system buses. The modified Harvard memory architecture results since the bus interfaces are activated by address range and include both instruction fetches and operand data references on a given bus port. A traditional Harvard architecture separates instruction fetches and operand data references onto specific bus ports regardless of access address.

The code bus is typically used for instruction fetching and data accesses of PC-relative data, while the system bus is typically used for operand data references to the on- and off-chip memories and peripheral accesses. This bus structure fully supports concurrent

instruction fetch and data accesses, but the Cortex-M4 implementations can generate both types of references on each bus. Additionally, there is a separate 32-bit Private Peripheral Bus (PPB) connection to several important modules (for example, the Nested Vectored Interrupt Controller) accessible to only the core. By placing the various code and data sections in the appropriate locations within the memory map, overall system performance can be maximized.

To provide a “clean timing interface” on the core's system bus, instruction and vector fetch requests to this bus are registered. This increases fetch time by an additional cycle of latency because instructions fetched from the system bus take a minimum of two cycles. This also means that back-to-back instruction fetches from the system bus are not possible.

Instruction fetch requests to the code bus are not registered. It is recommended that performance critical code be located such that it fetches from the ICode bus interface as defined by addresses < 0x2000_0000 (the system bus interface includes the addresses ≥ 0x2000_0000 and < 0xE000_0000 and the Private Peripheral Bus is used for addresses ≥ 0xE000_0000).

NOTE

In the device, the memory map includes aliased address spaces that are mapped into the ICode region for code sections that reside in the system address space. As a simple example, the QSPI address space is located in the system region of the memory map, but a subset of this space is aliased so that it appears in the ICode region that instructions mapped into the QSPI space can be executed as maximum performance.

6.6 Major Platform Bus Interfaces

The platform supports the AMBA AHB and AXI bus protocol.

- HBSTRB is the only v6 extension that is supported
- HRESP[1] (SPLIT/RETRY) is not supported.
- HTRANS[1:0] = 2b01 (BUSY) is not supported
- HBURST - The platform and its memory controllers support the full range of AHB burst sizes

6.7 Clocks and Resets

The platform inputs several reset signals. The following table describes the use of each reset.

Table 6-2. Platform Reset Descriptions

Reset Name	Description
ipg_core_async_reset_b	asynchronous reset for CORTEX-M4
ipg_hard_async_reset_b	asynchronous system reset for platform modules
ipg_hard_async_po_reset_b	asynchronous power-on reset used in CM4 Core

The platform generates an internal reset, debug_reset_b, which is used to reset the following debug modules:

- AHB-AP
- APB-AP
- JTAG-AP
- SWO
- MDM-AP
- SWJ-DP
- TPIU
- CTM
- CTI
- ETF/ETR
- REPLICATORs
- FUNNEL
- TS-GEN
- TS-ENCODER
- TS-DECODER

The platform inputs several clocks. The following table describes each clock input:

Table 6-3. Platform Clock Descriptions

Clock Input Name	Description
sw_clk_tck	Single Wire/ JTAG Test Clock
trace_clk_in	TPIU/SWO Clock
tcmc_hclk	TCMC Clock
cm4_hclk	Gated CPU Clock. Platform output "cm4_gate_hclk" can be used as the enable signal.
cm4_fclk	Free-running CPU Clock

The platform clocks, cm4_fclk, cm4_hclk, tcmc_hclk are equal in frequency and phase.

6.8 Platform JTAG Requirements

The Miscellaneous Debug Module (MDM) contains the DAP (Debug Access Port) status, control and ID registers as well as the DAP mux. Access to these DAP registers is through the SWJ-DP. The read-only DAP status register is located at DAPADDR[31:0] = 32'h0400_0000, while the DAP control register is located at DAPADDR=32'h0400_0004. The IDR is located at DAPADDR = 32'h0400_00FC.

The platform's 32-bit *dap_status[31:1]* input vector is registered in the MDM. Bit 0 is reserved for platform itself.

The *mdm_ap_control[31:0]* platform output vector controls SoC-defined functions. The *mdm_ap_control[0]* is reserved for platform itself.

The IDR is a read-only register with a value of 32'h001C_0040.

6.9 Local Memory Controller (LMEM)

The Local Memory Controller provides the Arm Cortex-M4 processor with tightly-coupled processor-local memories and bus paths to all slave memory spaces. See the AHB Local Memory Controller chapter for details.

6.10 Miscellaneous Control Module (MCM)

The Miscellaneous Control Module (MCM) provides a myriad of miscellaneous control functions.

6.10.1 MCM features

The MCM includes the following features:

- Program-visible information on the platform configuration and revision

6.10.2 MCM Interrupts

The MCM generates the following interrupt requests:

- Normal interrupt

6.10.2.1 Normal interrupt

The MCM's normal interrupt is generated if any of the following is true:

- cache write buffer error
- fpu errors

6.11 LMEM memory map/register definition

For details on AHB LMEM registers, see the LMEM Register Descriptions

6.12 MCM Memory Map/Register Definition

MCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_0008	Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)	16	R	001Fh	6.12.1/78
E008_000A	Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)	16	R	000Fh	6.12.2/78
E008_000C	Crossbar Switch (AXBS) Control Register (MCM_PLACR)	32	R/W	0000_0000h	6.12.3/79
E008_0020	Fault address register (MCM_FADR)	32	R	Undefined	6.12.4/80
E008_0024	Fault attributes register (MCM_FATR)	32	R	Undefined	6.12.5/81
E008_0028	Fault data register (MCM_FDR)	32	R	Undefined	6.12.6/83

6.12.1 Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)

PLASC is a 16-bit read-only register identifying the presence/absence of bus slave connections to the device's crossbar switch.

Address: E008_0000h base + 8h offset = E008_0008h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								ASC							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

MCM_PLASC field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ASC	Each bit in the ASC field indicates whether there is a corresponding connection to the crossbar switch's slave input port. 0 A bus slave connection to AXBS input port *n* is absent 1 A bus slave connection to AXBS input port *n* is present

6.12.2 Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)

PLAMC is a 16-bit read-only register identifying the presence/absence of bus master connections to the device's crossbar switch.

Address: E008_0000h base + Ah offset = E008_000Ah

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								AMC							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

MCM_PLAMC field descriptions

Field	Description
15–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
AMC	Each bit in the AMC field indicates whether there is a corresponding connection to the AXBS master input port.

Table continues on the next page...

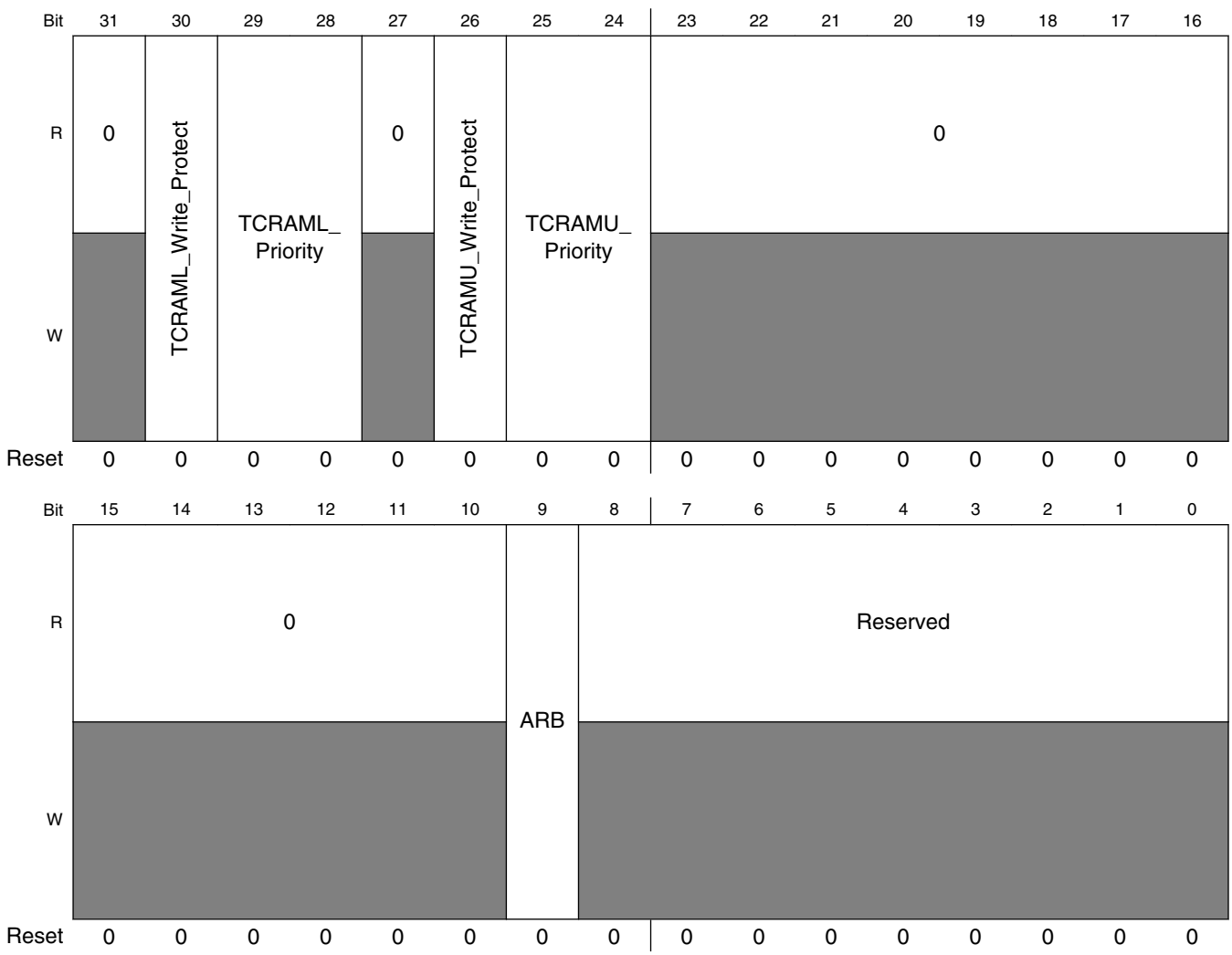
MCM_PLAMC field descriptions (continued)

Field	Description
	0 A bus master connection to AXBS input port *n* is absent 1 A bus master connection to AXBS input port *n* is present

6.12.3 Crossbar Switch (AXBS) Control Register (MCM_PLACR)

The PLACR register selects the arbitration policy for the crossbar masters.

Address: E008_0000h base + Ch offset = E008_000Ch



MCM_PLACR field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 TCRAML_Write_Protect	TCRAML_Write_Protect
29–28 TCRAML_Priority	TCRAML_Priority
27 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
26 TCRAMU_Write_Protect	TCRAMU_Write_Protect
25–24 TCRAMU_Priority	TCRAMU_Priority
23–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 ARB	Arbitration select 0 Fixed-priority arbitration for the crossbar masters 1 Round-robin arbitration for the crossbar masters
Reserved	This field is reserved.

6.12.4 Fault address register (MCM_FADR)

When a properly-enabled cache write buffer error interrupt event is detected, the faulting address is captured in the MCM_FADR register. The MCM logic supports capturing a single cache write buffer bus error event; if a subsequent error is detected before the captured error information has been read from the corresponding registers and the MCM_ISCR[CWBER] indicator cleared, the MCM_FATR[BEOVR] flag is set. However, no additional information is captured.

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. Attempted writes to this location are terminated with an error.

Address: E008_0000h base + 20h offset = E008_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDRESS																															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCM_FADR field descriptions

Field	Description
ADDRESS	Fault address

6.12.5 Fault attributes register (MCM_FATR)

When a properly-enabled cache write buffer error interrupt event is detected, the faulting attributes are captured in the MCM_FATR register.

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. Attempted writes to this location are terminated with an error.

Address: E008_0000h base + 24h offset = E008_0024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BEOVR	0														
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				BEMN				BEWT	0	BESZ		0		BEMD	BEDA
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCM_FATR field descriptions

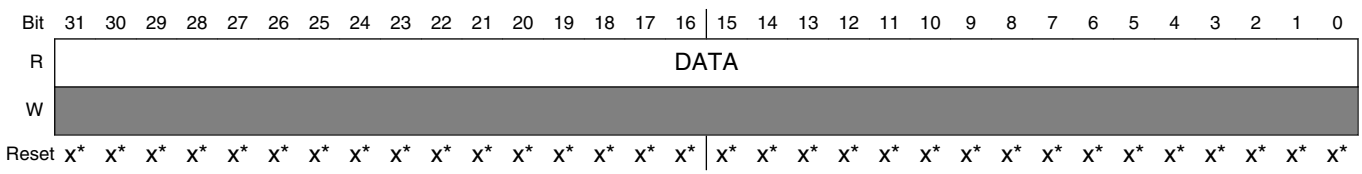
Field	Description
31 BEOVR	Bus error overrun Indicates if another cache write buffer bus error is detected before system software has retrieved all the error information from the original event, this overrun flag is set. The window of time is defined from the detection of the original cache write buffer error termination until the MCM_ISCR[CWBER] is written with a 1 to clear it and rearm the capture logic. This bit is set by the hardware and cleared whenever software writes a 1 to the CWBER bit. 0 No bus error overrun 1 Bus error overrun occurred. The FADR and FDR registers and the other FATR bits are not updated to reflect this new bus error.
30–12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 BEMN	Bus error master number Crossbar switch bus master number of the captured cache write buffer bus error. For this device, this value is always 0x1.
7 BEWT	Bus error write Indicates the type of system bus access when the error was detected. Since this logic is monitoring data transfers from the cache write buffer, this bit is always a logical one, signaling a write operation. 0 Read access 1 Write access
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 BESZ	Bus error size Indicates the size of the cache write buffer access when the error was detected. 00 8-bit access 01 16-bit access 10 32-bit access 11 Reserved
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 BEMD	Bus error privilege level Indicates the privilege level of the cache write buffer access when the error was detected. 0 User mode 1 Supervisor/privileged mode
0 BEDA	Bus error access type Indicates the type of cache write buffer access when the error was detected. This attribute is always a logical one signaling a data reference. 0 Instruction 1 Data

6.12.6 Fault data register (MCM_FDR)

When a properly-enabled cache write buffer error interrupt event is detected, the faulting data is captured in the MCM_FDR register.

The bits in this register are set by hardware and signaled by the assertion of MCM_ISCR[CWBER]. For byte and halfword writes, only the accessed byte lanes contain valid data; the contents of the other bytes are undefined. Attempted writes to this location are terminated with an error.

Address: E008_0000h base + 28h offset = E008_0028h



* Notes:
- x = Undefined at reset.

MCM_FDR field descriptions

Field	Description
DATA	Fault data

Chapter 7
AHB Local Memory Controller

7.1 Chip-specific AHB-LMEM information

Table 7-1. Reference links to related information

Topic	Related module	Reference
Full description	AHB-LMEM	AHB-LMEM
System memory map		System memory map

7.2 Introduction

The Local Memory Controller provides the processor with tightly-coupled processor-local memories and bus paths to all slave memory spaces.

7.2.1 Block Diagram

The processor has a modified 32-bit Harvard bus architecture. Using a 32-bit address space, low-order addresses (0x0000_0000 through 0x1FFF_FFFF) use the Processor Code (PC) bus, and high-order addresses (0x2000_0000 through 0xFFFF_FFFF) use the Processor System (PS) bus. As the bus names imply, normal operation has code accesses on the PC bus and data accesses on the PS bus.

This device has been augmented with tightly-coupled memories for the PC and PS buses. The memories include RAMs and caches. These local memories provide zero wait state access to RAM and cacheable address spaces.

The local memory controller includes the following memory controllers and their attached memories:

- SRAM lower (SRAM_L) controller via the PC bus

- SRAM upper (SRAM_U) controller via the PS bus
- Cache memory controller via the PC bus

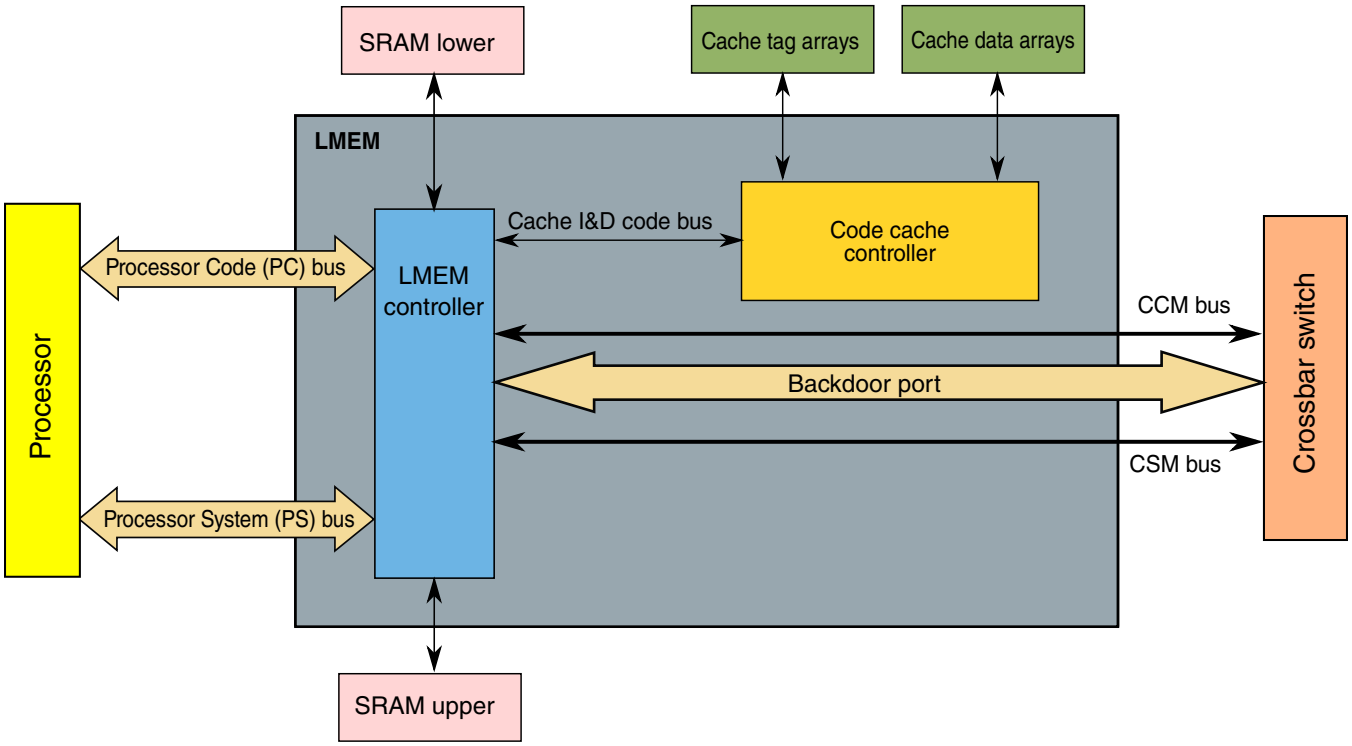


Figure 7-1. Local memory controller block diagram

NOTE

The SRAM and cache controllers reside within the LMEM, but the single-port synchronous RAM arrays used by these controllers are external.

The LMEM contains address decode logic for the PC and PS buses. This logic routes the core's accesses to the various system resources.

7.2.2 Cache features

A cache is a block of high-speed memory locations containing address information (commonly known as a tag) and the associated data. The purpose is to decrease the average time of a memory access. Caches operate on two principles of locality:

- Spatial locality — An access to one location is likely to be followed by accesses from adjacent locations (for example, sequential instruction execution or usage of a data structure).
- Temporal locality — An access to an area of memory is likely to be repeated within a short time period (for example, execution of a code loop).

To minimize the quantity of control information stored, the spatial locality property is used to group several locations together under the same tag. This logical block is commonly known as a cache line.

When data is loaded into a cache, access times for subsequent loads and stores are reduced, resulting in overall performance benefits. An access to information already in a cache is known as a cache hit, and other accesses are called cache misses.

Normally, caches are self-managing, with the updates occurring automatically. Whenever the processor wants to access a cacheable location, the cache is checked. If the access is a cache hit, the access occurs immediately. Otherwise, a location is allocated and the cache line is loaded from memory. Different cache topologies and access policies are possible. However, they must comply with the memory coherency model of the underlying architecture.

Caches introduce a number of potential problems, mainly because of:

- memory accesses occurring at times other than when the programmer would normally expect them,
- the existence of multiple physical locations where a data item can be held.

The local memory controller supports the following modes of operation:

1. Write-through — access to address spaces with this cache mode are cacheable.
    - If all cacheable spaces are read-only spaces, the cache will contain read-only data and all write to the cache will fault.
    - A write-through read miss on the input bus causes a line read on the output bus of a 32-byte-aligned memory address containing the desired address. This miss data is loaded into the cache and is marked as valid and not modified.
    - A write-through read hit to a valid cache location returns data from the cache with no output bus access.
    - A write-through write miss bypasses the cache and writes to the output bus (no allocate on write miss policy for write-through mode spaces).
    - A write-through write hit updates the cache hit line with the write data and writes to the output bus.
    - The caches are processor-local and do not support hardware cache coherency. If the processor has accessed write-through regions and an external bus master (such as DMA) then needs update these regions, software must first perform explicit cache clears to any needed cache memory range to ensure all modified cache lines update their associated memories before being modified by external masters and subsequent processor accesses will get the updated memory.
2. Non-cacheable — access to address spaces with this cache mode are not cacheable. These accesses bypass the cache and access the output bus.

7.3 Memory Map and Registers

The cache programmer's model provides a variety of registers for configuring and controlling the cache, as well as indirect access paths to all cache tag and data storage.

NOTE

The LMEM registers are accessible in supervisor mode only.

7.3.1 LMEM64 register descriptions

7.3.1.1 LMEM Memory map

LMEM base address: E008_2000h

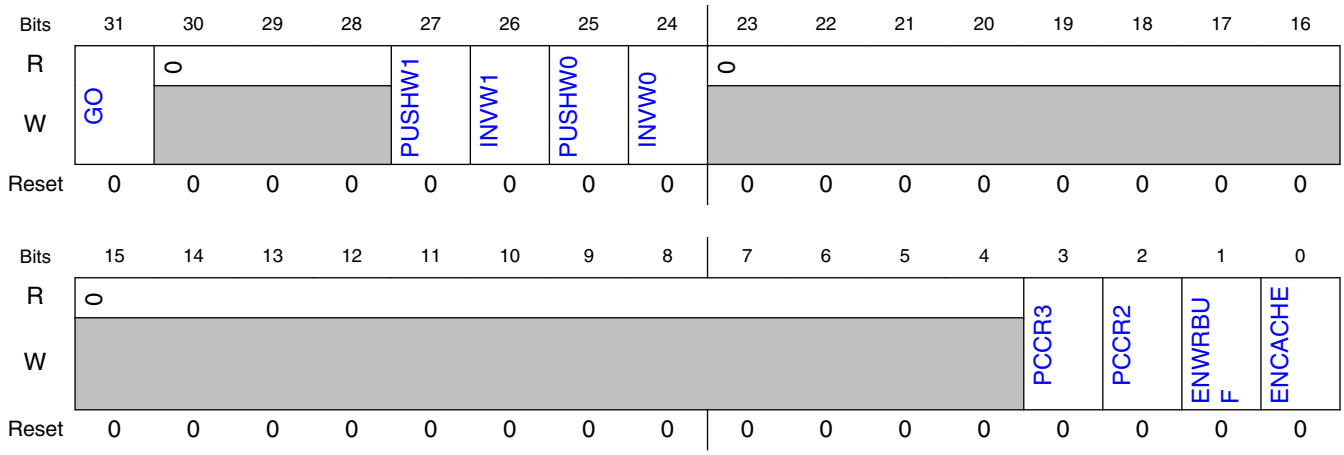
Offset	Register	Width (In bits)	Access	Reset value
0h	Cache control register (PCCCR)	32	RW	0000_0000h
4h	Cache line control register (PCCLCR)	32	RW	0000_0000h
8h	Cache search address register (PCCSAR)	32	RW	0000_0000h
Ch	Cache read/write value register (PCCCVR)	32	RW	0000_0000h

7.3.1.2 Cache control register (PCCCR)

7.3.1.2.1 Offset

Register	Offset
PCCCR	0h

7.3.1.2.2 Diagram



7.3.1.2.3 Fields

Field	Function
31 GO	Initiate Cache Command Setting this bit initiates the cache command indicated by bits 27-24. Reading this bit indicates if a command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. 0b - Write: no effect. Read: no cache command active. 1b - Write: initiate command indicated by bits 27-24. Read: cache command active.
30-28 —	Reserved
27 PUSHW1	Push Way 1 0b - No operation 1b - When setting the GO bit, push all modified lines in way 1
26 INVW1	Invalidate Way 1 **NOTE:** If the PUSHW1 and INVW1 bits are set, then after setting the GO bit, push all modified lines in way 1 and invalidate all lines in way 1 (clear way 1). 0b - No operation 1b - When setting the GO bit, invalidate all lines in way 1
25 PUSHW0	Push Way 0 0b - No operation 1b - When setting the GO bit, push all modified lines in way 0
24 INVW0	Invalidate Way 0 **NOTE:** If the PUSHW0 and INVW0 bits are set, then after setting the GO bit, push all modified lines in way 0 and invalidate all lines in way 0 (clear way 0). 0b - No operation 1b - When setting the GO bit, invalidate all lines in way 0.
23-4 —	Reserved
3	Forces no allocation on cache misses (must also have PCCR2 asserted)

Table continues on the next page...

Field	Function
PCCR3	
2 PCCR2	Forces all cacheable spaces to write through
1 ENWRBUF	Enable Write Buffer 0b - Write buffer disabled 1b - Write buffer enabled
0 ENCACHE	Cache enable 0b - Cache disabled 1b - Cache enabled

7.3.1.3 Cache line control register (PCCLCR)

7.3.1.3.1 Offset

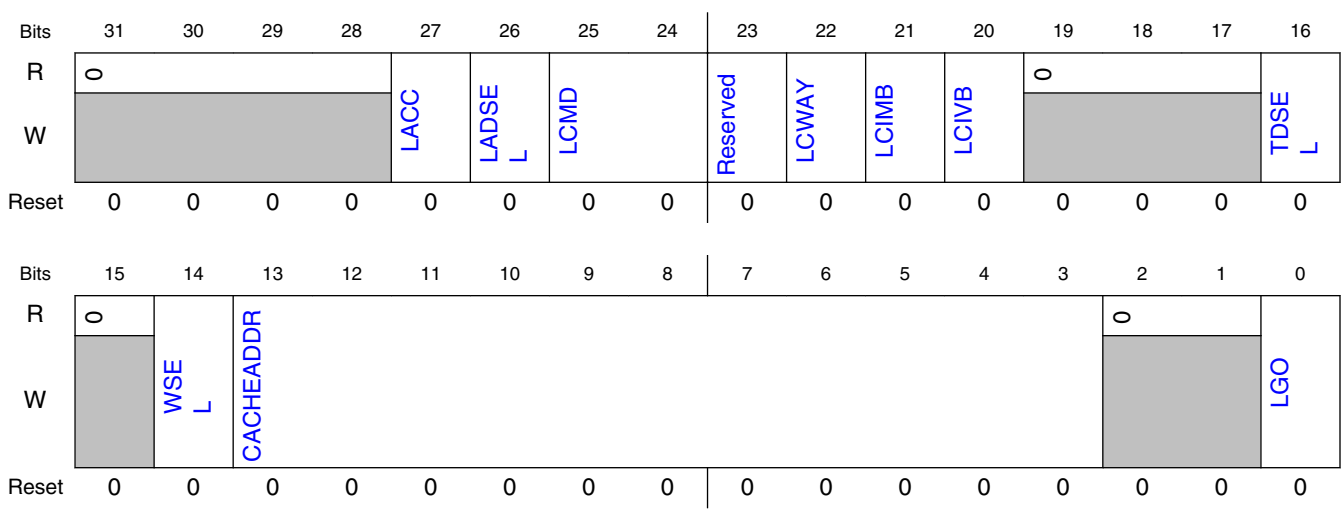
Register	Offset
PCCLCR	4h

7.3.1.3.2 Function

This register defines specific line-sized cache operations to be performed using a specific cache line address or a physical address.

If a physical address is specified, both ways of the cache are searched, and the command is only performed on the way which hits.

7.3.1.3.3 Diagram



7.3.1.3.4 Fields

Field	Function
31-28 —	Reserved
27 LACC	Line access type 0b - Read 1b - Write
26 LADSEL	Line Address Select When using the cache address, the way must also be specified in CLCR[WSEL]. When using the physical address, both ways are searched and the command is performed only if a hit. 0b - Cache address 1b - Physical address
25-24 LCMD	Line Command 00b - Search and read or write 01b - Invalidate 10b - Push 11b - Clear
23 —	Reserved
22 LCWAY	Line Command Way Indicates the way used by the line command. Only applies if valid bit LCIVB = 1.
21 LCIMB	Line Command Initial Modified Bit If command used cache address and way, then this bit shows the initial state of the modified bit If command used physical address and a hit, then this bit shows the initial state of the modified bit. If a miss, this bit reads zero.
20 LCIVB	Line Command Initial Valid Bit If command used cache address and way, then this bit shows the initial state of the valid bit If command used physical address and a hit, then this bit shows the initial state of the valid bit. If a miss, this bit reads zero.
19-17 —	Reserved
16 TDSEL	Tag/Data Select Selects tag or data for search and read or write commands. 0b - Data 1b - Tag
15 —	Reserved
14 WSEL	Way select Selects the way for line commands. 0b - Way 0

Table continues on the next page...

Field	Function
	1b - Way 1
13-3 CACHEADDR	Cache address CLCR[11:5] bits are used to access the tag arrays CLCR[11:3] bits are used to access the data arrays
2-1 —	Reserved
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CSAR[LGO] 0b - Write: no effect. Read: no line command active. 1b - Write: initiate line command indicated by bits 27-24. Read: line command active.

7.3.1.4 Cache search address register (PCCSAR)

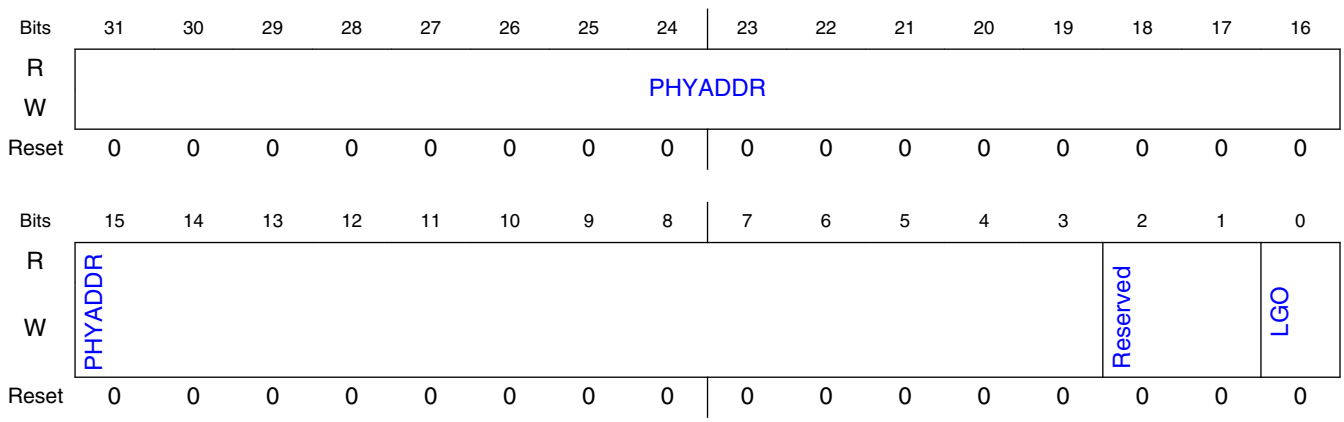
7.3.1.4.1 Offset

Register	Offset
PCCSAR	8h

7.3.1.4.2 Function

The CSAR register is used to define the explicit cache address or the physical address for line-sized commands specified in the CLCR[LADSEL] bit.

7.3.1.4.3 Diagram



7.3.1.4.4 Fields

Field	Function
31-3 PHYADDR	Physical Address PHYADDR represents bits [31:3] of the system address. CSAR[31:12] bits are used for tag compare CSAR[11:5] bits are used to access the tag arrays CSAR[11:3] bits are used to access the data arrays
2-1 —	Reserved
0 LGO	Initiate Cache Line Command Setting this bit initiates the cache line command indicated by bits 27-24. Reading this bit indicates if a line command is active **NOTE:** This bit stays set until the command completes. Writing zero has no effect. **NOTE:** This bit is shared with CLCR[LGO] 0b - Write: no effect. Read: no line command active. 1b - Write: initiate line command indicated by bits CLCR[27:24]. Read: line command active.

7.3.1.5 Cache read/write value register (PCCCVR)

7.3.1.5.1 Offset

Register	Offset
PCCCVR	Ch

7.3.1.5.2 Function

The CCVR register is used to source write data or return read data for the commands specified in the CLCR register.

7.3.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

7.3.1.5.4 Fields

Field	Function
31-0 DATA	Cache read/write Data For tag search, read or write: • CCVR[31:12] bits are used for tag array R/W value • CCVR[11:5] bits are used for tag set address on reads; unused on writes • CCVR[4:2] bits are reserved • CCVR[1] tag modify bit • CCVR[0] tag valid bit For data search, read or write: • CCVR[31:0] bits are used for data array R/W value

7.4 Functional Description

7.4.1 LMEM Function

The Local Memory Controller receives the following requests:

- Core master bus requests on the Processor Code (PC) bus,
- Core master bus requests on the Processor Space (PS) bus, and
- SRAM controller requests from all other bus masters on the backdoor port.

The Local Memory Controller address decode logic routes these accesses and also provides any crossbar switch slave target logic. Finally, the Local Memory controller provides the needed MPU connections for checking all SRAM controller and cacheable accesses.

The programming model for the Code Cache is accessed via the core's Private Peripheral Bus (PPB).

7.4.1.1 Processor Code accesses

Processor Code accesses are routed to the SRAM_L if they are mapped to that space. All other PC accesses are routed to the Code Cache Memory Controller. This controller then processes the cacheable accesses as needed, while bypassing the non-cacheable, cache write-through, cache miss, and cache maintenance accesses to the CCM bus and the crossbar switch using the Master0 port.

7.4.1.2 Processor System accesses

Processor Space accesses are routed to the SRAM_U if they are mapped to that space. All other PS accesses are routed to the CCM bus and the crossbar switch using the Master1 port.

7.4.1.3 Backdoor port accesses

All LMEM backdoor port accesses are for the SRAM controller. These accesses go to the SRAM_L or the SRAM_U depending on their specific address.

7.4.2 SRAM Function

7.4.2.1 SRAM Configuration

The figure below shows how the SRAM controller is configured.

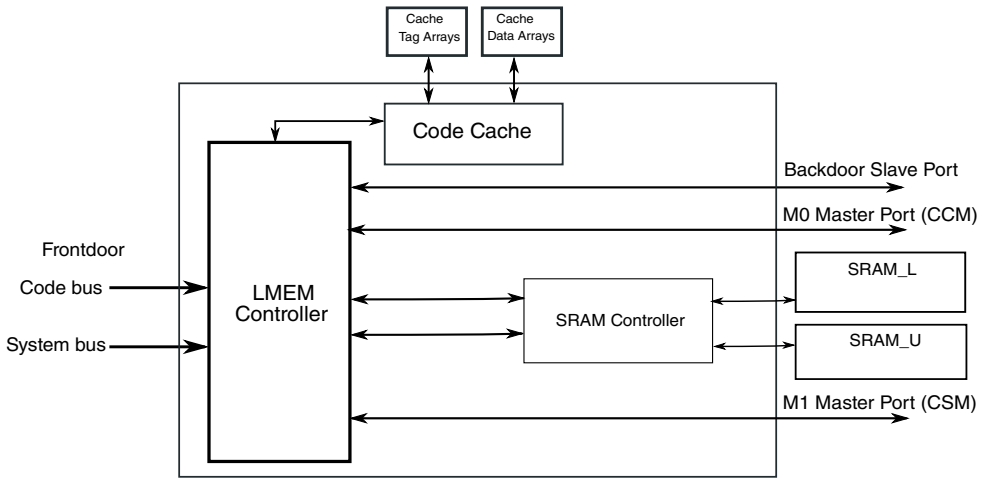


Figure 7-2. LMEM Interface to SRAM

7.4.2.2 SRAM Arrays

The on-chip SRAM is split into two logical arrays, SRAM_L and SRAM_U.

SRAM_L size is 192 KBytes.

SRAM_U size is 64 KBytes.

Valid address ranges for SRAM_L and SRAM_U are then defined as:

- SRAM_L = (0x20000_0000 - 192 KBytes) - 0x1fff_ffff
- SRAM_U = 0x2000_0000 - (0x2000_0000 + 64 KBytes)

SRAML_SIZE and SRAMU_SIZE do not have to be equal.

7.4.2.3 SRAM Accesses

The SRAM is split into two logical arrays that are 64-bits wide:

- SRAM_L — Accessible by the code bus of the core and by the backdoor port.
- SRAM_U — Accessible by the system bus of the core and by the backdoor port.

The backdoor port makes the SRAM accessible to the non-core bus masters (such as DMA).

The diagram below illustrates the SRAM accesses within the device.

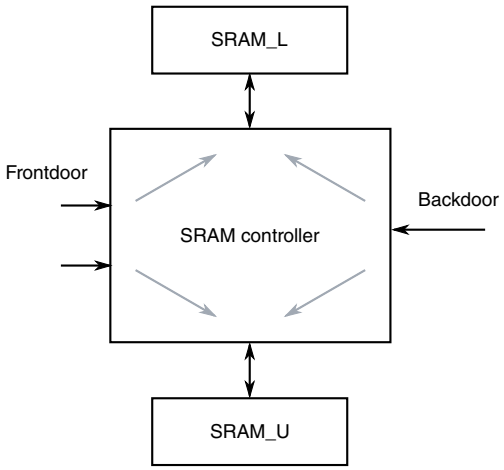


Figure 7-3. SRAM access diagram

The following simultaneous accesses can be made to different logical halves of the SRAM:

- Core code and core system
- Core code and non-core master
- Core system and non-core master

NOTE

Two non-core masters cannot access SRAM simultaneously. The required arbitration and serialization is provided by the crossbar switch. The SRAM_L and SRAM_U arbitration is controlled by the SRAM controller based on the configuration bits in the MCM module.

NOTE

Burst-access cannot occur across the 0x2000_0000 boundary that separates the two SRAM arrays. The two arrays should be treated as separate memory ranges for burst accesses.

7.4.3 Cache Function

The cache on this device is structured as follows. The cache has a 2-way set-associative cache structure with a total size of 8 KBytes for the Code Cache. The cache has 32-bit address, 64-bit data paths and a 32-byte line size. The cache tags and data storage use single-port, synchronous RAMs.

For the Code 8-KByte cache, each cache TAG function uses two 128 x 22-bit RAM arrays and the cache DATA function uses two 512 x 64-bit RAM arrays. The cache TAG entries store 20 bits of upper address as well as a modified and valid bit per cache line. The cache DATA entries store eight byes of code or data.

All normal cache accesses use physical addresses. This leads to the following cache address use:

CODE CACHE - 8 KByte size = (128 sets) x (32-byte lines) x (2-way set associative)

Table 7-2. Tag Cache Address Use

Tag Cache Address Use	Code Cache
Tag Hit Address Range	Address[31:12]
Tag Set Select Address Range	Address[11:5] used to select 1 of 128 sets
Not Used	Address[4:0]

Table 7-3. Data Cache Address Use

Data Cache Address Use	Code Cache
Not Used	Address[31:12]
Data Set Select Address Range	Address[11:5] used to select one of 128 sets
64-bit word select	Address[4:3] used to select one of four 64-bit words within a set
Byte select	Address[2:0] used to select the byte within the 64-bit word

7.4.4 Cache Control

The Code Cache is disabled at reset. Cache tag and data arrays are not cleared at reset. Therefore, to enable the cache, cache commands must be done to clear and initialize the required tag array bits and to configure and enable the caches.

7.4.4.1 Cache set commands

The cache set commands may operate on:

- all of way 0,
- all of way 1, or
- all of both ways (complete cache).

Cache set commands are initiated using the upper bits in the CCR register. Cache set commands perform their operation on the cache independent of the cache enable bit, CCR[ENCACHE].

A cache set command is initiated by setting the CCR[GO] bit. This bit also acts as a busy bit for set commands. It stays set while the command is active and is cleared by the hardware when the set command completes.

Supported cache set commands are given in Table 7-4. Set commands work as follows:

- Invalidate – Unconditionally clear valid and modify bits of a cache entry.
- Push – Push a cache entry if it is valid and modified, then clear the modify bit. If entry not valid or not modified, leave as is.
- Clear – Push a cache entry if it is valid and modified, then clear the valid and modify bits. If entry not valid or not modified, clear the valid bit.

Table 7-4. Cache Set Commands

CCR[27:24]				Command
PUSHW1	INVW1	PUSHW0	INVW0	
0	0	0	0	NOP
0	0	0	1	Invalidate all way 0
0	0	1	0	Push all way 0
0	0	1	1	Clear all way 0
0	1	0	0	Invalidate all way 1
0	1	0	1	Invalidate all way 1; invalidate all way 0 (invalidate cache)
0	1	1	0	Invalidate all way 1; push all way 0
0	1	1	1	Invalidate all way 1; clear all way 0
1	0	0	0	Push all way 1
1	0	0	1	Push all way 1; invalidate all way 0
1	0	1	0	Push all way 1; push all way 0 (push cache)
1	0	1	1	Push all way 1; clear all way 0
1	1	0	0	Clear all way 1
1	1	0	1	Clear all way 1; invalidate all way 0
1	1	1	0	Clear all way 1; push all way 0
1	1	1	1	Clear all way 1; clear all way 0 (clear cache)

After a reset, complete an invalidate cache command before using the cache. It is possible to combine the cache invalidate command with the cache enable. That is, setting CCR to 0x8500_0003 will invalidate the cache and enable the cache and write buffer.

7.4.4.2 Cache line commands

Cache line commands operate on a single line in the cache at a time. Cache line commands can be performed using a physical or cache address.

- A cache address consists of a set address and a way select. The line command acts on the specified cache line.
- Cache line commands with physical addresses first search both ways of the cache set specified by the following physical address bits. If they hit, the commands perform their action on the hit way:
  - For Code Cache - [11:5]

Cache line commands are specified using the upper bits in the CLCR register. Cache line commands perform their operation on the cache independent of the cache enable bit (CCR[ENCACHE]). Using a cache address, the command can be completely specified using the CLCR register. Using a physical address, the command must also use the CSAR register to specify the physical address.

A line cache command is initiated by setting the line command go bit (CLCR[LGO] or CSAR[LGO]). This bit also acts a a busy bit for line commands. It stays set while the command is active and is cleared by the hardware when the command completes.

The CLCR[27:24] bits select the line command as follows:

Table 7-5. Cache Line Commands

CLCR[27:24]			Command
LACC	LADSEL	LCMD	
0	0	00	Search by cache address and way
0	0	01	Invalidate by cache address and way
0	0	10	Push by cache address and way
0	0	11	Clear by cache address and way
0	1	00	Search by physical address
0	1	01	Invalidate by physical address
0	1	10	Push by physical address
0	1	11	Clear by physical address
1	0	00	Write by cache address and way
1	0	01	Reserved, NOP
1	0	10	Reserved, NOP
1	0	11	Reserved, NOP
1	1	*xx*	Reserved, NOP

7.4.4.2.1 Executing a series of line commands using cache addresses

A series of line commands with incremental cache addresses can be performed by just writing to the CLCR.

- Place the command in CLCR[27:24],
- Set the way (CLCR[WSEL]) and tag/data (CLCR[TDSEL]) controls as needed,
- Place the cache address in CLCR[CACHEADDR], and
- Set the line command go bit (CLCR[LGO]).

When one line command completes, initiate the next command by following these steps:

- Increment the cache address (at bit 3 to step through data or at bit 5 to step through lines), and
- Set the line command go bit (CLCR[LGO]).

7.4.4.2.2 Executing a series of line commands using physical addresses

Perform a series of line commands with incremental physical addresses using the following steps:

- Write to the CLCR.
  - Place the command in CLCR[27:24]
  - Set the tag/data (CLCR[TDSEL]) control
- Place the physical address in CSAR[PHYADDR] and set the line command go bit (CSAR[LGO]).

When one line command completes, initiate the next command by following these steps:

- Increment the physical address (at bit 3 to step through data or at bit 5 to step through lines), and
- Set the line command go bit (CSAR[LGO]).

The line command go bit is shared between the CLCR and CSAR registers, so that the above steps can be completed in a single write to the CSAR register.

7.4.4.2.3 Line command results

At completion of a line command, the CLCR register contains information on the initial state of the line targeted by the command. For line commands with cache addresses, this information is read before the line command action is performed from the targeted cache line. For line commands with physical addresses, this information is read on a hit before the line command action is performed from the hit cache line or has initial valid bit

cleared if the command misses. In general, if the valid indicator (CLCR[LCIVB] is cleared, the targeted line was invalid at the start of the line command and no line operation was performed.

Table 7-6. Line command results

CLCR[22:20]			For cache address commands	For physical address commands
LCWAY	LCIMB	LCIVB		
0	0	0	Way 0 line was invalid	No hit
0	0	1	Way 0 valid, not modified	Way 0 valid, not modified
0	1	0	Way 0 line was invalid	No hit
0	1	1	Way 0 valid and modified	Way 0 valid and modified
1	0	0	Way 1 line was invalid	No hit
1	0	1	Way 1 valid, not modified	Way 1 valid, not modified
1	1	0	Way 1 line was invalid	No hit
1	1	1	Way 1 valid and modified	Way 1 valid and modified

At completion of a line command other than a write, the CCVR (Cache R/W Value Register) contains information on the initial state of the line tag or data targeted by the command. For line commands, CLCR[TDSEL] selects between tag and data. If the line command used a physical address and missed, the data is don't care. For write commands, the CCVR holds the write data.

Chapter 8
Bit Manipulation Engine (BME)

8.1 Chip-specific BME information

Table 8-1. Reference links to related information

Topic	Related module	Reference
Full description	BME	BME
System memory map		System memory map

8.2 BME

The Bit Manipulation Engine (BME) provides hardware support for atomic read-modify-write memory operations to the peripheral address space in Arm Cortex-M based microcontrollers. This architectural capability is also known as "decorated storage" as it defines a mechanism for providing additional semantics for load and store operations to memory-mapped peripherals beyond just the reading and writing of data values to the addressed memory locations. In the BME definition, the "decoration", that is, the additional semantic information, is encoded into the peripheral address used to reference the memory.

Table 8-2. BME configuration

Parameter	Description
Name	BME
Supported standard verison	NA
Instances	2
Configurable features	NA
Interface speed	Platform bus clock frequency
External I/O pins	NA

8.3 Introduction

The Bit Manipulation Engine (BME) provides hardware support for atomic read-modify-write memory operations to the peripheral address space in Cortex-M4 based microcontrollers.

This architectural capability is also known as "decorated storage" as it defines a mechanism for providing additional semantics for load and store operations to memory-mapped peripherals beyond just the reading and writing of data values to the addressed memory locations. In the BME definition, the "decoration", that is, the additional semantic information, is encoded into the peripheral address used to reference the memory.

By combining the basic load and store instructions of the Arm Cortex-M instruction set architecture (v6M, v7M) with the concept of decorated storage provided by the BME, the resulting implementation provides a robust and efficient read-modify-write capability to this class of ultra low-end microcontrollers. The resulting architectural capability defined by this core platform function is targeted at the manipulation of n-bit fields in peripheral registers and is consistent with I/O hardware addressing in the Embedded C standard. For most BME commands, a single core read or write bus cycle is converted into an atomic read-modify-write, that is, an indivisible "read followed by a write" bus sequence.

BME decorated references are only available on system bus transactions generated by the processor core and targeted at the standard 512 KB peripheral address space based at 0x4100_0000. The decoration semantic is embedded into address bits[28:19], creating a 448 MB space at addresses 0x4400_0000–0x5FFF_FFFF for AIPS; these bits are stripped out of the actual address sent to the peripheral bus controller and used by the BME to define and control its operation.

8.3.1 Features

The key features of the BME include:

- Lightweight implementation of decorated storage for selected address spaces
- Additional access semantics encoded into the reference address
- Resides between a switch slave port and a peripheral bridge bus controller
- Two-stage pipeline design matching the AHB system bus protocol
- Combinationally passes non-decorated accesses to peripheral bridge bus controller
- Conversion of decorated loads and stores from processor core into atomic read-modify-writes

- Decorated loads support unsigned bit field extracts, load-and-{set,clear} 1-bit operations
- Decorated stores support bit field inserts, logical AND, OR, and XOR operations
- Support for byte, halfword and word-sized decorated operations
- Supports minimum signal toggling on AHB output bus to reduce power dissipation

8.3.2 Modes of operation

The BME module does not support any special modes of operation. As a memory-mapped device located on a crossbar slave AHB system bus port, BME responds strictly on the basis of memory addresses for accesses to the peripheral bridge bus controller.

All functionality associated with the BME module resides in the core platform's clock domain; this includes its connections with the crossbar slave port and the PBRIDGE bus controller.

8.4 Memory map and register definition

The BME module provides a memory-mapped capability and does not include any programming model registers.

The exact set of functions supported by the BME are detailed in the Functional description.

The peripheral address space occupies a 516 KB region: 512 KB based at 0x4100_0000 plus a 4 KB space based at 0x4100_F000 for GPIO accesses; the decorated address space is mapped to the 448 MB region located at 0x4400_0000–0x5FFF_FFFF.

8.5 Functional description

Information found here details the specific functions supported by the BME.

Recall the combination of the basic load and store instructions of the Cortex-M instruction set architecture (v6M, v7M) plus the concept of decorated storage provided by the BME, the resulting implementation provides a robust and efficient read-modify-write capability to this class of ultra low-end microcontrollers. The resulting architectural capability defined by this core platform function is targeted at the manipulation of n-bit fields in peripheral registers and is consistent with I/O hardware addressing in the

Embedded C standard. For most BME commands, a single core read or write bus cycle is converted into an atomic read-modify-write, that is, an indivisible "read followed by a write" bus sequence.

Consider decorated store operations first, then decorated loads.

8.5.1 BME decorated stores

The functions supported by the BME's decorated stores include three logical operators (AND, OR, XOR) plus a bit field insert.

For all these operations, BME converts a single decorated AHB store transaction into a 2-cycle atomic read-modify-write sequence, where the combined read-modify operation is performed in the first AHB data phase, and then the write is performed in the second AHB data phase.

A generic timing diagram of a decorated store showing a peripheral bit field insert operation is shown as follows:

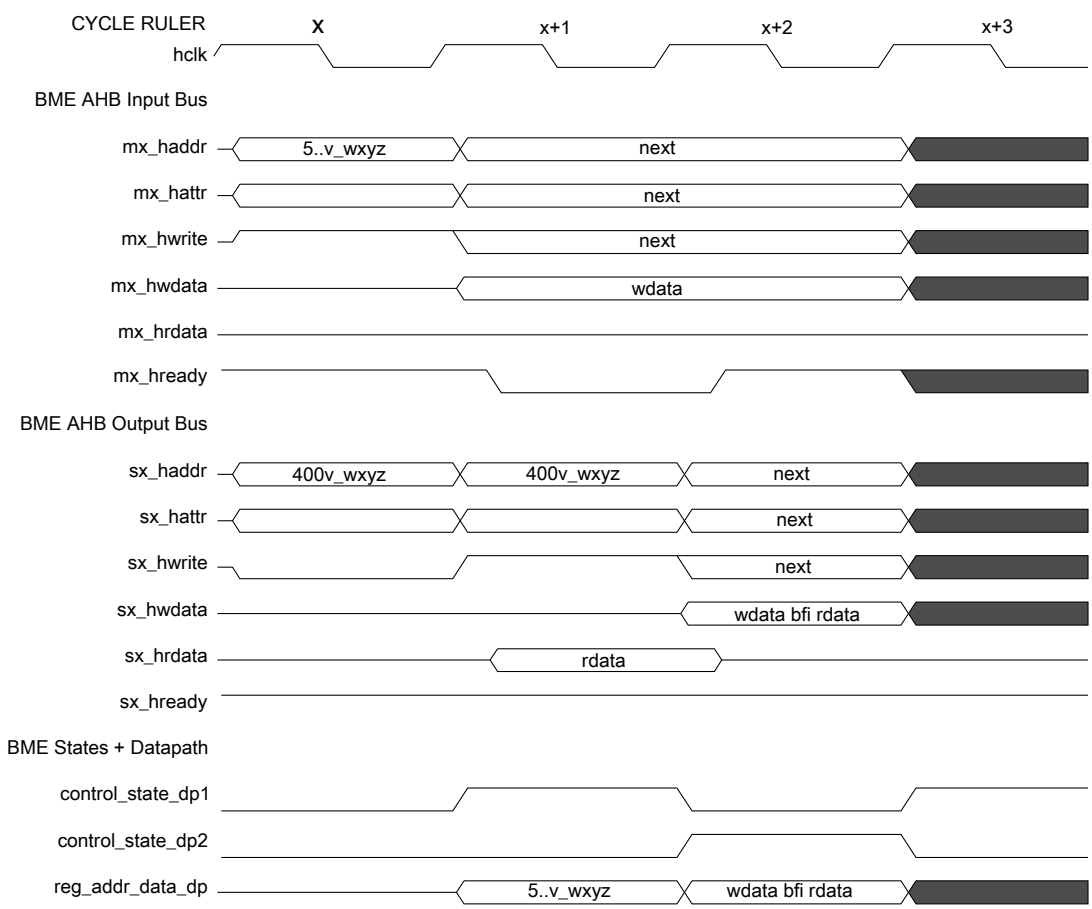


Figure 8-1. Decorated store: bit field insert timing diagram

All the decorated store operations follow the same execution template shown in Figure 8-1, a two-cycle read-modify-write operation:

1. Cycle x, 1st AHB address phase: Write from input bus is translated into a read operation on the output bus using the actual memory address (with the decoration removed) and then captured in a register.
2. Cycle x+1, 2nd AHB address phase: Write access with the registered (but actual) memory address is output
3. Cycle x+1, 1st AHB data phase: Memory read data is modified using the input bus write data and the function defined by the decoration and captured in a data register; the input bus cycle is stalled.
4. Cycle x+2, 2nd AHB data phase: Registered write data is sourced onto the output write data bus.

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

8.5.1.1 Decorated store logical AND (AND)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read;
2. It is then modified by performing a logical AND operation using the write data operand sourced for the system bus cycle
3. Finally, the result of the AND operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The core performs the required write data lane replication on byte and halfword transfers.

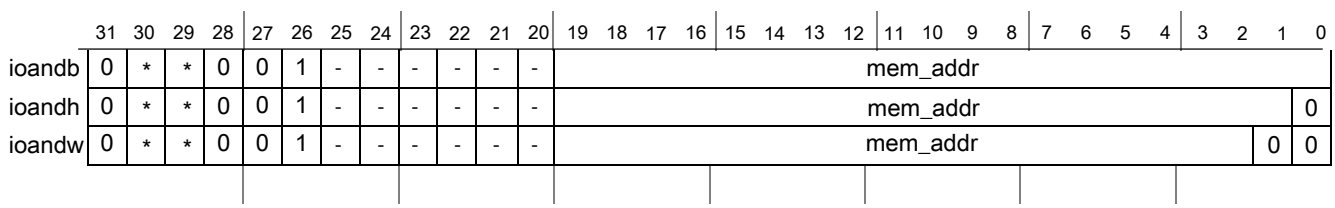


Figure 8-2. Decorated store address: logical AND

See Figure 8-2, where addr[30:29] = 10 for peripheral, addr[28:26] = 001 specifies the AND operation, and mem_addr[19:0] specifies the address offset into the space based at 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated AND write operation is defined in the following pseudo-code as:

```
ioand<sz>(accessAddress, wdata)                  // decorated store AND
tmp   =  mem[accessAddress & 0xE00FFFFF, size]   // memory read
tmp   =  tmp & wdata                             // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp      // memory write
```

where the operand size <sz> is defined as b(yte, 8-bit), h(alfword, 16-bit) and w(ord, 32-bit). This notation is used throughout the document.

In the cycle definition tables, the notations AHB_ap and AHB_dp refer to the address and data phases of the BME AHB transaction. The cycle-by-cycle BME operations are detailed in the following table.

Table 8-3. Cycle definitions of decorated store: logical AND

Pipeline stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form (rdata & wdata) and capture destination data in register	Perform write sending registered data to memory

8.5.1.2 Decorated store logical OR (OR)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read.
2. It is then modified by performing a logical OR operation using the write data operand sourced for the system bus cycle.
3. Finally, the result of the OR operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The core performs the required write data lane replication on byte and halfword transfers.

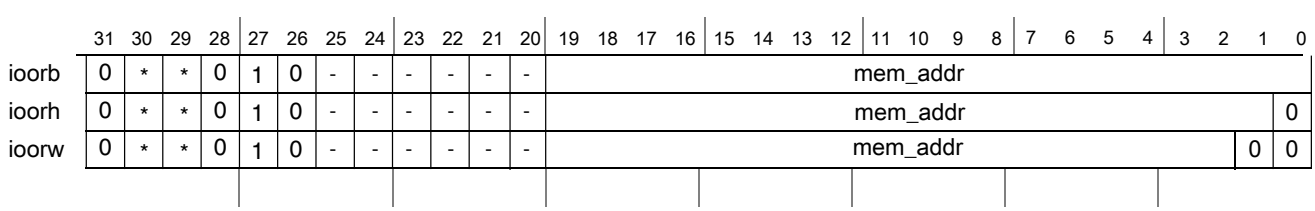


Figure 8-3. Decorated address store: logical OR

See Figure 8-3,where addr[30:29] =10 for peripheral, addr[28:26] = 010 specifies the OR operation, and mem_addr[19:0] specifies the address offset into the space based at 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated OR write operation is defined in the following pseudo-code as:

```
ioor<sz>(accessAddress, wdata)                  // decorated store OR

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
tmp   =  tmp | wdata                            // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp     // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 8-4. Cycle definitions of decorated store: logical OR

Pipeline stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form (rdata | wdata) and capture destination data in register	Perform write sending registered data to memory

8.5.1.3 Decorated store logical XOR (XOR)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read.
2. It is then modified by performing a logical XOR (exclusive-OR) operation using the write data operand sourced for the system bus cycle.
3. Finally, the result of the XOR operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The core performs the required write data lane replication on byte and halfword transfers.

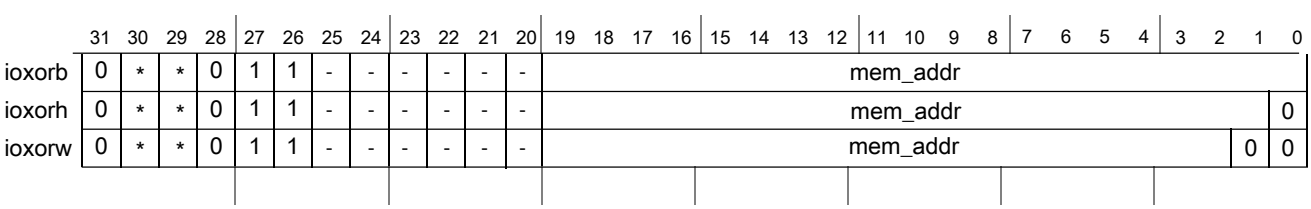


Figure 8-4. Decorated address store: logical XOR

See Figure 8-4, where addr[30:29] =10 for peripheral, addr[28:26] = 011 specifies the XOR operation, and mem_addr[19:0] specifies the address offset into the peripheral space based at 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care".

The decorated XOR write operation is defined in the following pseudo-code as:

```
ioxor<sz>(accessAddress, wdata)                 // decorated store XOR

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
tmp   =  tmp ^ wdata                            // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp     // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 8-5. Cycle definitions of decorated store: logical XOR

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form (rdata ^ wdata) and capture destination data in register	Perform write sending registered data to memory

8.5.1.4 Decorated store bit field insert (BFI)

This command inserts a bit field contained in the write data operand, defined by LSB position (b) and the bit field width (w+1), into the memory "container" defined by the access size associated with the store instruction using an atomic read-modify-write sequence.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit).

NOTE

For the word sized operation, the maximum bit field width is 16 bits. The core performs the required write data lane replication on byte and halfword transfers.

The BFI operation can be used to insert a single bit into a peripheral. For this case, the w field is simply set to 0, indicating a bit field width of 1.

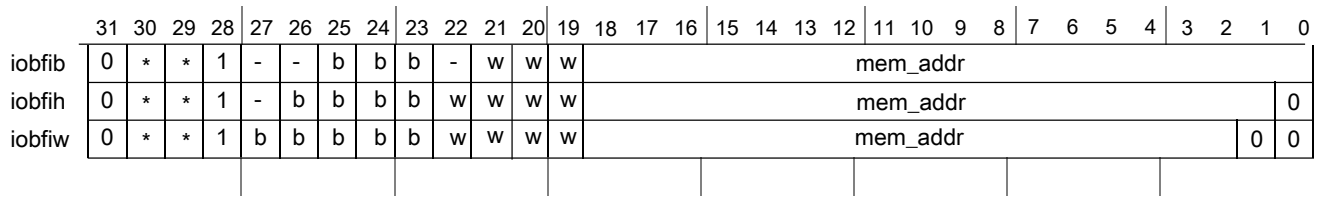


Figure 8-5. Decorated address store: bit field insert

where addr[30:29] =10 for peripheral,addr[28] = 1 signals a BFI operation, addr[27:23] is "b", the LSB identifier, addr[22:19] is "w", the bit field width minus 1 identifier, and addr[18:0] specifies the address offset into the peripheral space based at 0x4000_0000 for peripherals. The "-" indicates an address bit "don't care". Note, unlike the other decorated store operations, BFI uses addr[19] as the least significant bit in the "w" specifier and not as an address bit.

The decorated BFI write operation is defined in the following pseudo-code as:

```
iobfi<sz>(accessAddress, wdata)                 // decorated bit field insert

tmp   =  mem[accessAddress & 0xE007FFFF, size]  // memory read
mask  = ((1 << (w+1)) - 1) << b                 // generate bit mask
tmp   =  tmp   & ~mask                          // modify
      |  wdata &  mask
mem[accessAddress & 0xE007FFFF, size] = tmp     // memory write
```

The write data operand (wdata) associated with the store instruction contains the bit field to be inserted. It must be properly aligned within a right-aligned container, that is, within the lower 8 bits for a byte operation, the lower 16 bits for a halfword, or the entire 32 bits for a word operation.

To illustrate, consider the following example of the insertion of the 3-bit field "xyz" into an 8-bit memory container, initially set to "abcd_efgh". For all cases, w is 2, signaling a bit field width of 3.

```
if b = 0 and the decorated store (strb) Rt register[7:0] = ----_-xyz,
   then destination is "abcd_exyz"
if b = 1 and the decorated store (strb) Rt register[7:0] = ----_xyz-,
   then destination is "abcd_xyzh"
if b = 2 and the decorated store (strb) Rt register[7:0] = ---x_yz--,
   then destination is "abcx_yzgh"
if b = 3 and the decorated store (strb) Rt register[7:0] = --xy_z---,
   then destination is "abxy_zfgh"
if b = 4 and the decorated store (strb) Rt register[7:0] = -xyz_----,
   then destination is "axyz_efgh"
if b = 5 and the decorated store (strb) Rt register[7:0] = xyz-_----,
   then destination is "xyzd_efgh"
if b = 6 and the decorated store (strb) Rt register[7:0] = yz--_----,
   then destination is "yzcd_efgh"
if b = 7 and the decorated store (strb) Rt register[7:0] = z---_----,
   then destination is "zbcd_efgh"
```

Note from the example, when the starting bit position plus the field width exceeds the container size, only part of the source bit field is inserted into the destination memory location. Stated differently, if (b + w+1) > container_width, only the low-order "container_width - b" bits are actually inserted.

The cycle-by-cycle BME operations are detailed in the following table.

Table 8-6. Cycle definitions of decorated store: bit field insert

Pipeline stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Form bitwise ((mask) ? wdata : rdata)) and capture destination data in register	Perform write sending registered data to memory

8.5.2 BME decorated loads

The functions supported by the BME's decorated loads include two single-bit load-and-{set, clear} operators plus unsigned bit field extracts.

For the two load-and-{set, clear} operations, BME converts a single decorated AHB load transaction into a two-cycle atomic read-modify-write sequence, where the combined read-modify operations are performed in the first AHB data phase, and then the write is performed in the second AHB data phase as the original read data is returned to the processor core. For an unsigned bit field extract, the decorated load transaction is stalled for one cycle in the BME as the data field is extracted, then aligned and returned to the processor in the second AHB data phase. This is the only decorated transaction that is not an atomic read-modify-write, as it is a simple data read.

A generic timing diagram of a decorated load showing a peripheral load-and-set 1-bit operation is shown as follows.

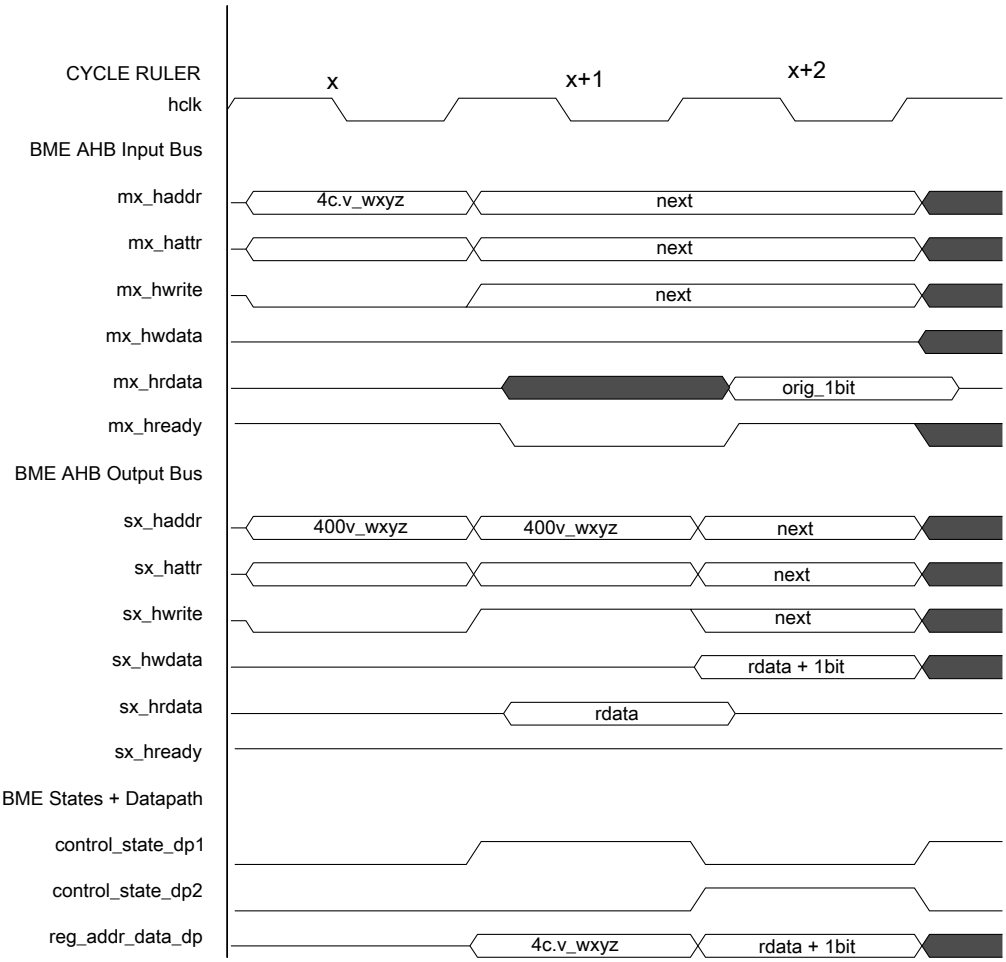


Figure 8-6. Decorated load: load-and-set 1-bit field insert timing diagram

Decorated load-and-{set, clear} 1-bit operations follow the execution template shown in the above figure: a 2-cycle read-modify-write operation:

1. Cycle x, first AHB address phase: Read from input bus is translated into a read operation on the output bus with the actual memory address (with the decoration removed) and then captured in a register
2. Cycle x+1, second AHB address phase: Write access with the registered (but actual) memory address is output
3. Cycle x+1, first AHB data phase: The "original" 1-bit memory read data is captured in a register, while the 1-bit field is set or clear based on the function defined by the decoration with the modified data captured in a register; the input bus cycle is stalled
4. Cycle x+2, second AHB data phase: The selected original 1-bit is right-justified, zero-filled and then driven onto the input read data bus, while the registered write data is sourced onto the output write data bus

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

A generic timing diagram of a decorated load showing an unsigned peripheral bit field operation is shown in the following figure.

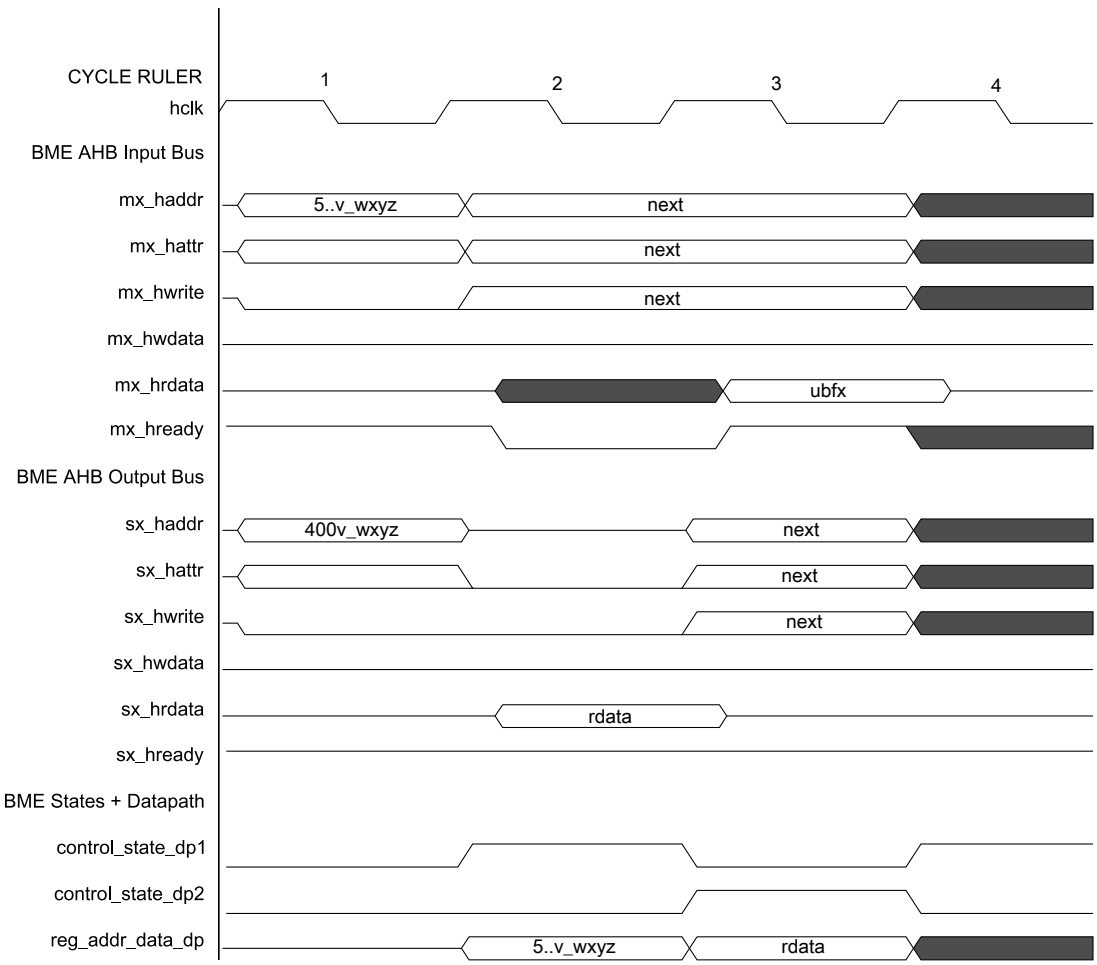


Figure 8-7. Decorated load: unsigned bit field insert timing diagram

The decorated unsigned bit field extract follows the same execution template shown in the above figure, a 2-cycle read operation:

- Cycle x, 1st AHB address phase: Read from input bus is translated into a read operation on the output bus with the actual memory address (with the decoration removed) and then captured in a register
- Cycle x+1, 2nd AHB address phase: Idle cycle

- Cycle x+1, 1st AHB data phase: A bit mask is generated based on the starting bit position and the field width; the mask is AND'ed with the memory read data to isolate the bit field; the resulting data is captured in a data register; the input bus cycle is stalled
- Cycle x+2, 2nd AHB data phase: Registered data is logically right-aligned for proper alignment and driven onto the input read data bus

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

8.5.2.1 Decorated load: load-and-clear 1 bit (LAC1)

This command loads a 1-bit field defined by the LSB position (b) into the core's general purpose destination register (Rt) and zeroes the bit in the memory space after performing an atomic read-modify-write sequence.

The extracted 1-bit data field from the memory address is right-justified and zero-filled in the operand returned to the core.

The data size is specified by the read operation and can be byte (8-bit), halfword (16-bit) or word (32-bit).

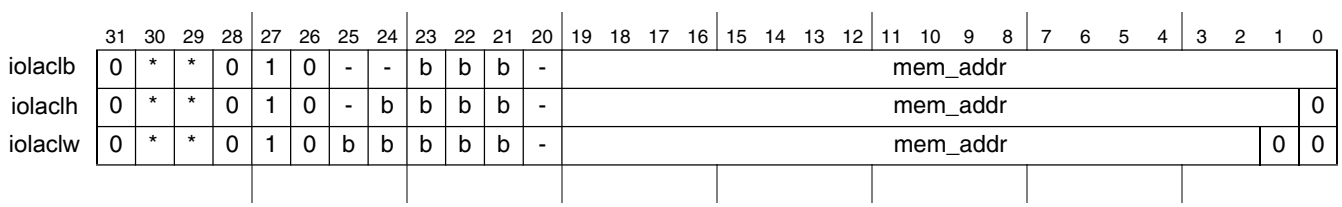


Figure 8-8. Decorated load address: load-and-clear 1 bit

See Figure 8-8, where addr[30:29] = 10 for peripheral, addr[28:26] = 010 specifies the load-and-clear 1 bit operation, addr[25:21] is "b", the bit identifier, and mem_addr[19:0] specifies the address offset into the space based at 0x4000_0000 for peripheral. The "-" indicates an address bit "don't care".

The decorated load-and-clear 1-bit read operation is defined in the following pseudo-code as:

```
rdata =  iolac1<sz>(accessAddress)              // decorated load-and-clear 1

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
mask  =  1 << b                                 // generate bit mask
rdata = (tmp &  mask) >> b                      // read data returned to core
tmp   =  tmp & ~mask                            // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp     // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 8-7. Cycle definitions of decorated load: load-and-clear 1 bit

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Extract bit from rdata; Form (rdata & ~mask) and capture destination data in register	Return extracted bit to master; Perform write sending registered data to memory

8.5.2.2 Decorated Load: Load-and-Set 1 Bit (LAS1)

This command loads a 1-bit field defined by the LSB position (b) into the core's general purpose destination register (Rt) and sets the bit in the memory space after performing an atomic read-modify-write sequence.

The extracted one bit data field from the memory address is right justified and zero filled in the operand returned to the core.

The data size is specified by the read operation and can be byte (8-bit), halfword (16-bit) or word (32-bit).

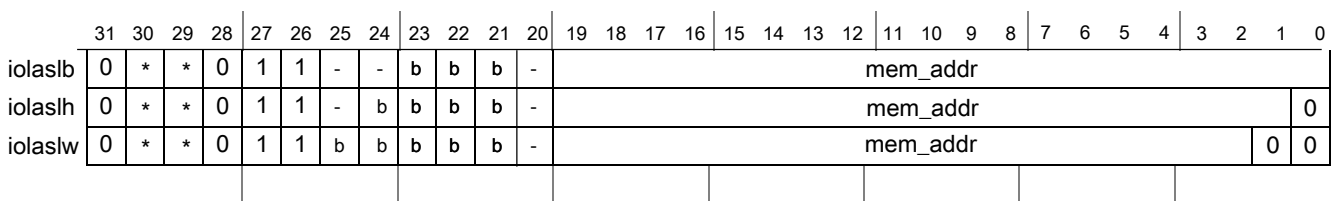


Figure 8-9. Decorated load address: load-and-set 1 bit

where addr[30:29] = 10 for peripheral, addr[28:26] = 011 specifies the load-and-set 1 bit operation, addr[25:21] is "b", the bit identifier, and mem_addr[19:0] specifies the address offset into the space based at 0x4000_0000 for peripheral. The "-" indicates an address bit "don't care".

The decorated Load-and-Set 1 Bit read operation is defined in the following pseudo-code as:

```
rdata =  iolas1<sz>(accessAddress)              // decorated load-and-set 1

tmp   =  mem[accessAddress & 0xE00FFFFF, size]  // memory read
mask  =  1 << b                                 // generate bit mask
rdata = (tmp &  mask) >> b                      // read data returned to core
```

```
tmp    =  tmp |  mask                              // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp       // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 8-8. Cycle definitions of decorated load: load-and-set 1-bit

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Capture address, attributes	Recirculate captured addr + attr to memory as slave_wt	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Extract bit from rdata; Form (rdata | mask) and capture destination data in register	Return extracted bit to master; Perform write sending registered data to memory

8.5.2.3 Decorated load unsigned bit field extract (UBFX)

This command extracts a bit field defined by LSB position (b) and the bit field width (w +1) from the memory "container" defined by the access size associated with the load instruction using a two-cycle read sequence.

The extracted bit field from the memory address is right-justified and zero-filled in the operand returned to the core. Recall this is the only decorated operation that does not perform a memory write, that is, UBFX only performs a read.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). Note for the word sized operation, the maximum bit field width is 16 bits.

The use of a UBFX operation is recommended to extract a single bit. For this case, the w field is simply set to 0, indicating a bit field width of 1.

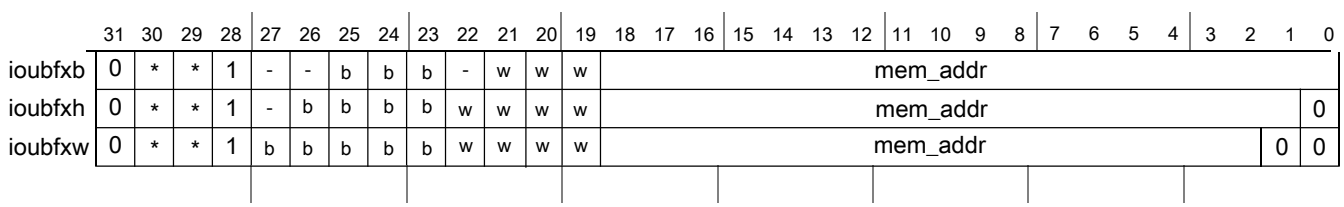


Figure 8-10. Decorated load address: unsigned bit field extract

See Figure 8-10, where addr[30:29] = 10 for peripheral, addr[28] = 1 specifies the unsigned bit field extract operation, addr[27:23] is "b", the LSB identifier, addr[22:19] is "w", the bit field width minus 1 identifier, and mem_addr[18:0] specifies the address

offset into the space based at 0x4000_0000 for peripheral. The "-" indicates an address bit "don't care". Note, unlike the other decorated load operations, UBFX uses addr[19] as the least significant bit in the "w" specifier and not as an address bit.

The decorated unsigned bit field extract read operation is defined in the following pseudo-code as:

```
rdata =  ioubfx<sz>(accessAddress)              // unsigned bit field extract

tmp   =  mem[accessAddress & 0xE007FFFF, size]  // memory read
mask  = ((1 << (w+1)) - 1) << b                 // generate bit mask
rdata = (tmp &  mask) >> b                      // read data returned to core
```

Like the BFI operation, when the starting bit position plus the field width exceeds the container size, only part of the source bit field is extracted from the destination memory location. Stated differently, if (b + w+1) > container_width, only the low-order "container_width - b" bits are actually extracted. The cycle-by-cycle BME operations are detailed in the following table.

Table 8-9. Cycle definitions of decorated load: unsigned bit field extract

Pipeline Stage	Cycle		
	x	x+1	x+2
BME AHB_ap	Forward addr to memory; Decode decoration; Capture address, attributes	Idle AHB address phase	<next>
BME AHB_dp	<previous>	Perform memory read; Form bit mask; Form (rdata & mask) and capture destination data in register	Logically right shift registered data; Return justified rdata to master

8.5.3 Additional details on decorated addresses and GPIO accesses

As previously noted, the peripheral address space occupies a 516 KB region: 512 KB based at 0x4000_0000 plus a 4 KB space based at 0x400F_F000 for GPIO accesses. This memory layout provides compatibility with the Kinetis K Family and provides 129 address "slots", each 4 KB in size.

The GPIO address space is multiply-mapped by the hardware: it appears at the "standard" system address 0x400F_F000 and is physically located in the address slot corresponding to address 0x4000_F000. Decorated loads and stores create a slight complication involving accesses to the GPIO. Recall the use of address[19] varies by decorated operation; for AND, OR, XOR, LAC1 and LAS1, this bit functions as a true address bit, while for BFI and UBFX, this bit defines the least significant bit of the "w" bit field specifier.

As a result, undecorated GPIO references and decorated AND, OR, XOR, LAC1 and LAS1 operations can use the standard 0x400F_F000 base address, while decorated BFI and UBFX operations must use the alternate 0x4000_F000 base address. Another implementation can simply use 0x400F_F000 as the base address for all undecorated GPIO accesses and 0x4000_F000 as the base address for all decorated accesses. Both implementations are supported by the hardware.

Table 8-10. Decorated peripheral and GPIO address details

Peripheral address space	Description
0x4000_0000–0x4007_FFFF	Undecorated (normal) peripheral accesses
0x4008_0000–0x400F_EFFF	Illegal addresses; attempted references are aborted and error terminated
0x400F_F000–0x400F_FFFF	Undecorated (normal) GPIO accesses using standard address
0x4010_0000–0x43FF_FFFF	Illegal addresses; attempted references are aborted and error terminated
0x4400_0000–0x4FFF_FFFF	Decorated AND, OR, XOR, LAC1, LAS1 references to peripherals and GPIO based at either 0x4000_F000 or 0x400F_F000
0x5000_0000–0x5FFF_FFFF	Decorated BFI, UBFX references to peripherals and GPIO only based at 0x4000_F000

8.6 Application information

In this section, GNU assembler macros with C expression operands are presented as examples of the required instructions to perform decorated operations.

This section specifically presents a partial bme.h file defining the assembly language expressions for decorated logical stores: AND, OR, and XOR. Comparable functions for BFI and the decorated loads are more complex and available in the complete BME header file.

These macros use the same function names presented in Functional description.

```
#define IOANDW(ADDR,WDATA)          \
    __asm("ldr    r3, =(1<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "str    r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOANDH(ADDR,WDATA)          \
    __asm("ldr    r3, =(1<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strh   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOANDB(ADDR,WDATA)          \
    __asm("ldr    r3, =(1<<26);"    \
          "orr    r3, %[addr];"     \
          "mov    r2, %[wdata];"    \
          "strb   r2, [r3];"        \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");
```

```
#define IOORW(ADDR,WDATA)            \
    __asm("ldr    r3, =(1<<27);"     \
          "orr    r3, %[addr];"      \
          "mov    r2, %[wdata];"     \
          "str    r2, [r3];"         \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORH(ADDR,WDATA)            \
    __asm("ldr    r3, =(1<<27);"     \
          "orr    r3, %[addr];"      \
          "mov    r2, %[wdata];"     \
          "strh   r2, [r3];"         \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORB(ADDR,WDATA)            \
    __asm("ldr    r3, =(1<<27);"     \
          "orr    r3, %[addr];"      \
          "mov    r2, %[wdata];"     \
          "strb   r2, [r3];"         \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOXORW(ADDR,WDATA)           \
    __asm("ldr    r3, =(3<<26);"     \
          "orr    r3, %[addr];"      \
          "mov    r2, %[wdata];"     \
          "str    r2, [r3];"         \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOXORH(ADDR,WDATA)           \
    __asm("ldr    r3, =(3<<26);"     \
          "orr    r3, %[addr];"      \
          "mov    r2, %[wdata];"     \
          "strh   r2, [r3];"         \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOXORB(ADDR,WDATA)           \
    __asm("ldr    r3, =(3<<26);"     \
          "orr    r3, %[addr];"      \
          "mov    r2, %[wdata];"     \
          "strb   r2, [r3];"         \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");
```

Chapter 9
Messaging Unit (MU)

9.1 Chip-specific MU information

Table 9-1. Reference links to related information

Topic	Related module	Reference
Full description	MU	MU
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

9.1.1 Power mode status through Status Register (SR)

MU_SR[PM] bit field in MU_A and MU_B memory maps changes upon power mode entry request, while a power mode is effectively entered only after that request is acknowledged. For accurate power mode status information, the PMC 0 Power Mode Status register (PM_STAT) can be used.

NOTE

It should be noted that Processor B is not able to check Processor A's power mode status through PMC0 whenever the platform clock is gated in the latter's domain. Under such scenario, Processor A's power mode status is only available via MU.

9.1.2 Mode of operation: Wait and Stop

NOTE

For i.MX 7ULP, CA7 WAIT mode is now also seen as STOP (since PSTOP3 is used to emulate WAIT in the application domain). See the description in the PM field.

NOTE

Whenever Processor B's domain (Application Domain) resets and Processor B's reset hold control is active (MU_ACR[RSTH] == 0x1), the following procedure should be observed before deactivating Processor B's reset hold control:

- Ensure Processor B's domain is out of reset (MU_ASR[RS] == 0x0);
- Wait at least 513 Processor B core clock cycles (to ensure the CA7 cache invalidate operation completes).

9.1.3 Messaging Unit

The Messaging Unit (MU) is a shared peripheral with a 32-bit IP bus interface and interrupt request signals to each host processor. The MU exposes a set of registers to each processor which facilitate inter-processor communication via 32-bit words, interrupts and flags. Interrupts may be independently masked by each processor to allow polled-mode operation. The single non-maskable interrupt cannot be masked in the MU.

NOTE

In this chapter, Processor A or core0 refers to CM4 core and Processor B or core1 refers to CA7 core

Table 9-2. Messaging unit configuration

Parameter	Description
Name	Messaging Unit (MU)
Instances	1
Configurable features	NA
Interface speed	NA
External I/O pins	NA

9.2 Overview

The Messaging Unit module enables two processors within the SoC to communicate and coordinate by passing messages (e.g. data, status and control) through the MU interface. The MU also provides the ability for one processor to signal the other processor using interrupts.

Because the MU manages the messaging between processors potentially using different clocks, the MU must synchronize the accesses from one side to the other. The MU accomplishes synchronization using two sets of matching registers (Processor A-facing, Processor B-facing).

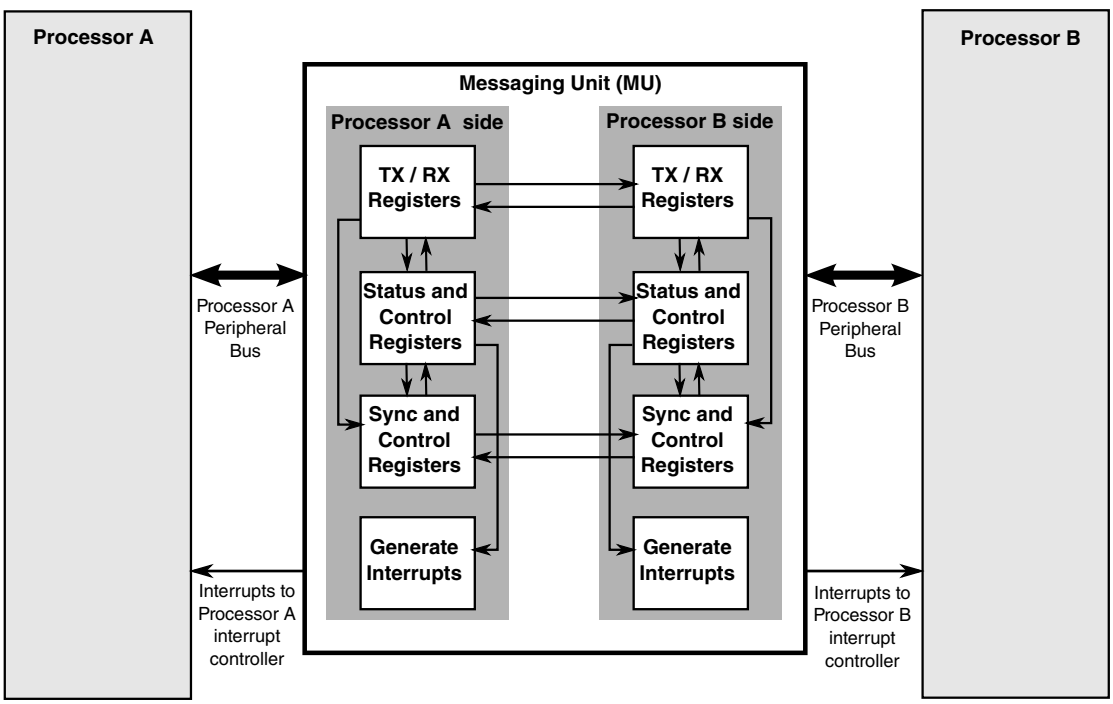


Figure 9-1. MU Block Diagram

9.2.1 Features

The MU includes the following features:

- Messaging control by interrupts or by polling
- Symmetrical processor interfaces with each side supporting the following:
  - Three general-purpose flags reflected to the other side
  - Four general-purpose interrupt requests reflected to the other side

- Four receive registers with maskable interrupt
- Four transmit registers with maskable interrupt

- The Processor B can take the Processor A out of low-power modes by asserting one of the interrupts to the Processor A and vice versa

NOTE

MU chapter references to Processor A may also correspond to Core 0 and references to Processor B may also correspond to Core 1 in other parts of this document.

9.2.2 Modes of Operation

The MU supports the modes described in the indicated sections:

- Low Power Modes

9.3 Register Definition

The MU provides transmit and receive data registers for the communication between Processor A and Processor B. Some control and status registers to the Processor A and Processor B sides for control operations (such as interrupts and reset), and for status checking of the other MU-side. The following diagram shows the MU registers schematic.

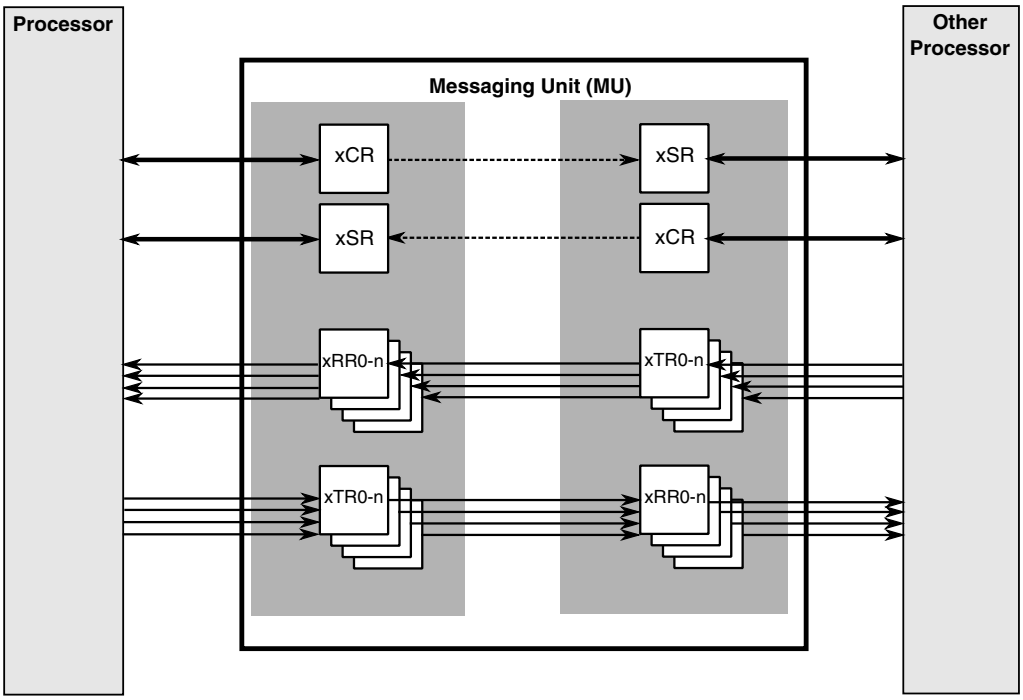


Figure 9-2. MU Registers

The detailed MU Register Definition can be found below.

NOTE

Writing to an RRn register can cause a transfer error.

9.3.1 MUA register descriptions

This section contains the detailed register descriptions for the MUA registers.

9.3.1.1 MU Memory map

MUA base address: 4102_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	RO	0100_0001h
4h	Parameter Register (PAR)	32	RO	0000_0000h
20h - 2Ch	Transmit Register (TR0 - TR3)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
40h - 4Ch	Receive Register (RR0 - RR3)	32	RO	0000_0000h
60h	Status Register (SR)	32	W1C	00F0_0080h
64h	Control Register (CR)	32	RW	See description.

9.3.1.2 Version ID Register (VER)

9.3.1.2.1 Offset

Register	Offset
VER	0h

9.3.1.2.2 Function

Use Version ID register to determine the version ID and feature set number of MUA.

9.3.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

9.3.1.2.4 Fields

Field	Function
31-24	Major Version Number
MAJOR	This read only field returns the major version number for MUA.
23-16	Minor Version Number

Table continues on the next page...

Field	Function
MINOR	This read only field returns the minor version number for MUA.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number for MUA. 000000000000x1xxb - Core Control and Status Registers are implemented in both MUA and MUB. 000000000000xx1xb - RAIP/RAIE register bits are implemented. 000000000000xxx0b - Standard features implemented

9.3.1.3 Parameter Register (PAR)

9.3.1.3.1 Offset

Register	Offset
PAR	4h

9.3.1.3.2 Function

Use Parameter register to determine the parameter settings of MUA.

9.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PARAMETER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PARAMETER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.1.3.4 Fields

Field	Function
31-0 PARAMETER	This bitfield contains the parameter settings of MUA.

9.3.1.4 Transmit Register (TR0 - TR3)

9.3.1.4.1 Offset

Register	Offset
TR0	20h
TR1	24h
TR2	28h
TR3	2Ch

9.3.1.4.2 Function

Use MUA Transmit Register (TRn, 32-bit, write-only) to transmit a message or data to MUB.

- You can only write to the TRn register when the TEn bit in SR register is set to "1".
- Reading the TRn register returns all zeros.

9.3.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.1.4.4 Fields

Field	Function
31-0 DATA	DATA MUA Transmit Register Data. (Write-only) • Data written to the TRn register is reflected in the MUB Receive Register n (RRn). The TRn and RRn registers are not double-buffered, a write to the TRn register overrides the data readable at the RRn register.

Field	Function
	• A write to the transmit register clears a "transmitter empty" bit (TEn) in the MUA Status Register (SR) on the transmitter side, and sets a "receiver full" bit (RFn) in the MUB Status Register (SR) on the receiver side (optionally triggering an interrupt 0 on the MUA side). • Any write to the TRn register will update all status information.

9.3.1.5 Receive Register (RR0 - RR3)

9.3.1.5.1 Offset

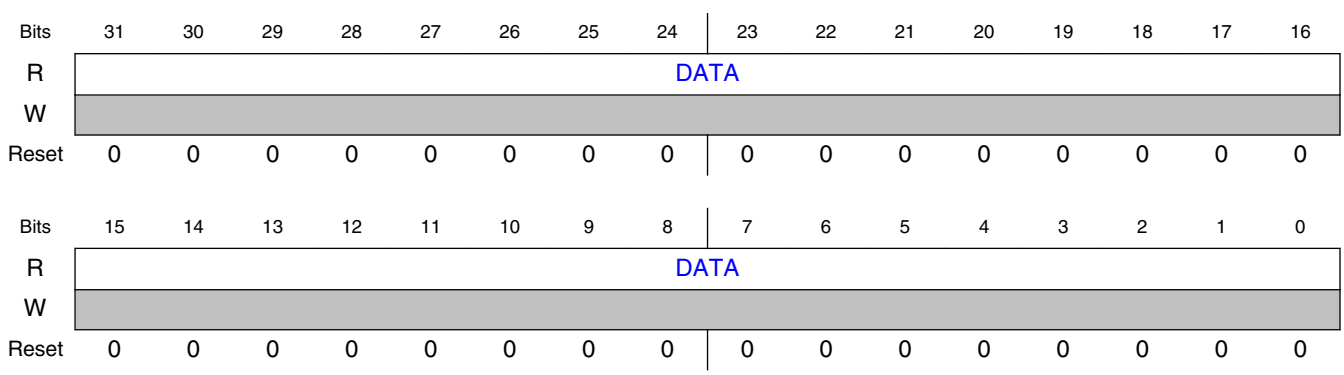
Register	Offset
RR0	40h
RR1	44h
RR2	48h
RR3	4Ch

9.3.1.5.2 Function

Use MUA Receive Register (RRn, 32-bit, read-only) to receive a message or data from the MUB.

- Data written to the MUB TRn register is immediately reflected in the MUA RRn register.
- You can only read the RRn register when the RFn bit in the SR register is set to "1".
- Writing to the RRn register generates an error response to the MUA.

9.3.1.5.3 Diagram



9.3.1.5.4 Fields

Field	Function
31-0 DATA	DATA MUA Receive Register. (Read-only) • Reflects the data written to MUB Transmit Register 0 (TRn). • Reading the RRn register clears the "receiver full" bit (RFn) in the MUA Status Register (SR) on the receiver side, and sets the "transmitter empty" bit (TEn) in the MUB Status Register on the transmitter side (optionally triggering a transmit interrupt 0 on the MUB side). • Any read of the RRn register will update all status information.

9.3.1.6 Status Register (SR)

9.3.1.6.1 Offset

Register	Offset
SR	60h

9.3.1.6.2 Function

Use the Processor A Status Register (SR, 32-bit, read-write) to show interrupt status from the Processor B, general purpose flags, the Processor B power mode, and to set dual function control-status bits.

- Some dual-purpose bits are set by the MU logic, and cleared by the Processor A-side programmer
- Other dual-purpose bits are set by the Processor A-side programmer, and cleared by the MU logic.

9.3.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GIPn				RFn				TEn				Reserved			
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved			Reserved	RAIP	RDIP	FUP	RS	PM		EP	NMIC	Fn		
W						W1C	W1C						W1C			
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

9.3.1.6.4 Fields

Field	Function
31-28 GIPn	GIPn For n = {0, 1, 2, 3} MUA General Interrupt Request n Pending. (Read-Write) • GIPn bit signals the MUA that the GIRn bit in the CR register on the MUB side was set from "0" to "1". If the GIEn bit in the MUA CR register is set to "1", a General Interrupt n request is issued. • The GIPn bit is cleared by writing it back as "1". Writing "0", or writing "1" when the GIPn bit is cleared is ignored. Use this feature in the interrupt routine, where the GIPn bit is cleared in order to de-assert the interrupt request source at the interrupt controller. An example of a proper bit clearing sequence is: clear MUA register, set the desired bit, and write it to the MUA SR register, thus clearing the GIPn bit. • GIPn bit is cleared when the MU is reset. 0000b - MUA general purpose interrupt n is not pending. (default) 0001b - MUA general purpose interrupt n is pending.
27-24 RFn	RFn For n = {0, 1, 2, 3} MUA Receive Register n Full. (Read-only) • The RFn bit is set to "1" when the MUB TRn register is written on the MUB side. • After the RFn bit is set to "1", the RFn bit signals the MUA side that new data is ready to be read by the MUA in the MUA RRn register, and a Receive n interrupt is issued on the MUA side (if the RIEn bit in the MUA CR register has been set to "1"). • RFn bit is cleared when the MUA RRn register is read, and when the MU is reset. 0000b - MUA RRn register is not full (default). 0001b - MUA RRn register has received data from MUB TRn register and is ready to be read by the MUA.
23-20 TEn	TEn For n = {0, 1, 2, 3} MUA Transmit Register n Empty. (Read-only) • The TEn bit is set to "1" after the MUB RRn register is read on the MUB side. • After the TEn bit is set to "1", the TEn bit signals the MUA side that the MUA TRn register is ready to be written on the MUA side, and a Transmit n interrupt is issued on the MUA side (if the TEn bit in the MUA CR register is set to "1"). • TEn bit is cleared after the MUA TRn register is written on the MUA side. • TEn bit is set to "1" when the MU is reset. 0000b - MUA TRn register is not empty. 0001b - MUA TRn register is empty (default).
19-15 —	Reserved
14-12 —	Reserved
11 —	Reserved
10 RAIP	RAIP Processor B Reset Asserted Interrupt Pending. (Read-Write) • RAIP bit signals the Processor A-side that the Processor B-side has entered reset.

Table continues on the next page...

Field	Function
	• RAIP bit is set to "1" after the MU Processor B-side enters reset (after synchronization). The interrupt generated by a Processor B-side reset assertion is ORed with the Processor A general purpose interrupt 3. The Processor A general purpose interrupt 3 is issued when the Processor B-side enters reset, if the interrupt is enabled by the RAIE bit. • To clear the RAIP bit, write "1", which also clears general purpose interrupt 3. • When Processor A-side of MU comes out of reset, the RAIP bit has value "0" (default). Processor A will then monitor the reset of Processor B-side and will assert RAIP if its reset asserts. This bitfield is only available on the Processor A side. 0b - Processor B-side did not enter reset 1b - Processor B-side entered reset
9 RDIP	RDIP Processor B Reset De-asserted Interrupt Pending. (Read-Write) • RDIP bit signals the Processor A-side that the Processor B-side has come out of reset. • RDIP bit is set to "1" after the MU Processor B-side comes out of reset, after synchronization. The interrupt generated by a Processor B-side reset de-assertion is ORed with the Processor A general purpose interrupt 3. The Processor A general purpose interrupt 3 is issued when the Processor B-side comes out of reset if the interrupt is enabled by the RDIE bit. • To clear the RDIP bit, write "1", which also clears general purpose interrupt 3. • When Processor A-side of MU comes out of reset, the RDIP bit has value "0"(default). Processor A then monitors the reset of Processor B and will assert RDIP reset deasserts. This takes 5-6 clock cycles, so may result in the RDIP bit being asserted after Processor A exits reset. This bitfield is only available on the Processor A side. 0b - Processor B-side did not exit reset 1b - Processor B-side exited from reset
8 FUP	FUP MUA Flags Update Pending. (Read-only) • FUP bit is set to "1" when the MUA side sends a Flags Update request to the MUB side. • A Flags Update request is generated when there is a change to the Fn[2:0] bits of the MUA CR register. No flag update changes are allowed while the FUP bit is set to "1". Any write to the Fn[2:0] bits of the MUA CR register, while the FUP bit is set to "1", will not generate a Flags Update event, and the Fn[2:0] bits will stay unchanged. • FUP bit is cleared when this Flags Update request is internally acknowledged (that the flag is updated) from the MU MUB side, and during MU reset. 0b - No flags updated, initiated by the MUA, in progress (default) 1b - MUA initiated flags update, processing
7 RS	RS MUB Reset State. (Read-only) • RS bit indicates if the MUB side of the MU is in a reset state or not. • If the RS bit is set to "1", then the MUB side of the MU is still in the reset state. • If the RS bit is cleared, then the MUB side of the MU are out of reset. • The RS bit is set to "1" during: a MUB system reset, or an MU reset (caused by setting the MUR bit at the CR register). • The RS bit is cleared when the reset sequence on the MUB side of the MU ends. After issuing any of the reset events mentioned previously, you should verify that the RS bit is cleared before starting any accesses. • When the MUB processor comes out of reset, the RS bit has value "1"(default). 0b - The MUB side of the MU is not in reset. 1b - The MUB side of the MU is in reset.
6-5	PM

Table continues on the next page...

Field	Function
PM	Processor B-side Power Mode. (Read-only) • PM[1:0] bits indicate the Processor B-side power mode. **NOTE:** The Processor B Power Mode is platform-specific. 00b - The MUB processor is in Run Mode. 01b - The MUB processor is in WAIT Mode. 10b - The MUB processor is in STOP/VLPS Mode. 11b - The MUB processor is in LLS/VLLS Mode.
4 EP	EP MUA Side Event Pending. (Read-only) • EP bit is set to "1" when the MUA side mechanism sends an event update request to the MUB side. • EP bit is cleared when the event update acknowledge is received. An "event" is any hardware message that is reflected in the MUB SR register on the MUB side (for example, "transmit register 0 written"). During normal operations, you do not have to deal with the state of the EP bit because the event update mechanism works automatically. • To ensure events have been posted to MUB before entering STOP mode, you should verify that the EP bit is cleared. If EP bit is set to "1", you should wait and continue to poll it (EP bit) before entering STOP mode. • Reading the MUA SR register (to check the EP bit) should be the last access to the MU that should be performed before entering STOP or WAIT modes; otherwise, the EP bit may be set by subsequent additional actions. • The EP bit is cleared when the MU resets. 0b - The MUA side event is not pending (default). 1b - The MUA side event is pending.
3 NMIC	NMIC Processor A Non-Maskable-Interrupt Clear. (Write-only) • NMIC bit is used by the Processor A-side MU Non-Maskable Interrupt service routine to clear the non-maskable interrupt. • Writing NMIC bit as "1" signals the MU to clear the NMI bit in the MUB CR register, thus de-asserting the interrupts and enabling the NMI bit to receive another interrupt. • The NMIC bit is always read as "0"; therefore it (NMIC bit) cannot be polled. The NMIC bit can only be used as part of the interrupt service routine, in which the NMIC bit should only be written as "1" once. • The NMIC bit is cleared when the MU resets. 0b - Default 1b - Writing "1" clears the NMI bit in the MUB CR register.
2-0 Fn	Fn For n = {0, 1, 2} MUA Side Flag n. (Read-only) • Fn bit is the MUA side flag that reflects the values written to the Fn bit in the MUB control register. • Every time that the Fn bit in the MUB CR register is written, the Fn bit in the MUB CR register write event updates the Fn bit in the MUA SR register after the event update latency, which is measured in terms of the number of clocks of the MUB and the MUA. 000b - Fn bit in the MUB CR register is written 0 (default). 001b - Fn bit in the MUB CR register is written 1.

9.3.1.7 Control Register (CR)

9.3.1.7.1 Offset

Register	Offset
CR	64h

9.3.1.7.2 Function

Use the Control Register (CR, 32-bit, read-write) to enable the MU interrupts on the MUA-side, and trigger events and interrupts on the MUB-side (general purpose interrupt, flag update).

9.3.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIEn				RIEn				TIEn				GIRn			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			RAIE	Reserved	BBOOT		CLKE	RSTH	RDIE	MUR	BHR	NMI	Fn		
Reset	0	0	0	0	0	u[1]	u	0	u[1]	0	0	0	0	0	0	0

1. Reset value loaded during processor A reset from Flash IFR.

9.3.1.7.4 Fields

Field	Function
31-28 GIEn	GIEn For n = {0, 1, 2, 3} MUA General Purpose Interrupt Enable n. (Read-Write) • GIEn bit enables MUA General Interrupt n. • If GIEn bit is set to "1" (enabled), then a General Interrupt n request is issued when the GIPn bit in the MUA SR register is set to "1". • If GIEn is cleared (disabled), then the value of the GIPn bit is ignored and no General Interrupt n request will be issued. • GIEn bit is cleared when the MU resets. 0000b - Disables MUA General Interrupt n. (default) 0001b - Enables MUA General Interrupt n.
27-24 RIEn	RIEn For n = {0, 1, 2, 3} MUA Receive Interrupt Enable n. (Read-Write) • RIEn bit enables MUA Receive Interrupt n.

Table continues on the next page...

Field	Function
	• If RIEn bit is set to "1" (enabled), then an MUA Receive Interrupt n request is issued when the RFn bit in the MUA SR register is set to "1". • If RIEn bit is cleared (disabled), then the value of the RFn bit is ignored and no MUA Receive Interrupt n request will be issued. • RIEn bit is cleared when the MU resets. 0000b - Disables MUA Receive Interrupt n. (default) 0001b - Enables MUA Receive Interrupt n.
23-20 TIEn	TIEn For n = {0, 1, 2, 3} MUA Transmit Interrupt Enable n. (Read-Write) • TIEn bit enables MUA Transmit Interrupt n. • If TIEn bit is set to "1" (enabled), then an MUA Transmit Interrupt n request is issued when the TEn bit in the MUA SR register is set to "1". • If TIEn bit is cleared (disabled), then the value of the TEn bit is ignored and no MUA Transmit Interrupt n request will be issued. • TIEn bit is cleared when the MU resets. 0000b - Disables MUA Transmit Interrupt n. (default) 0001b - Enables MUA Transmit Interrupt n.
19-16 GIRn	GIRn For n = {0, 1, 2, 3} MUA General Purpose Interrupt Request n. (Read-Write) • Writing "1" to the GIRn bit sets the GIPn bit in the MUB SR register on the Processor B-side. If the GIEn bit in the MUB CR register is set to "1" on the MUB side, a General Purpose Interrupt n request is triggered. • The GIRn bit is cleared if the GIPn bit (in the MUB SR register on the MUB side) is cleared by writing it (GIPn bit) as "1", thereby signalling the MUA that the interrupt was accepted (cleared by the software). The GIPn bit cannot be written as "0" on the MUA side. • To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit). • GIRn bit is cleared when the MU resets. 0000b - MUA General Interrupt n is not requested to the MUB (default). 0001b - MUA General Interrupt n is requested to the MUB.
15-13 —	Reserved
12 RAIE	RAIE Processor B Reset Assertion Interrupt Enable. (Read-Write) • RAIE bit enables Processor A General Interrupt 3. • If RAIE bit is set to "1", then General Interrupt 3 request is issued to the Processor A when the RAIP bit in the SR register is set to "1". • If RAIE is cleared, then the value of the RAIP bit is ignored and no General Interrupt 3 request will be issued. • The RAIE bit is cleared when the MU resets. This bit is only available on the Processor A side. 0b - Disables Processor A General Purpose Interrupt 3 request due to Processor B reset assertion. 1b - Enables Processor A General Purpose Interrupt 3 request due to Processor B reset assertion.
11 —	Reserved
10-9 BBOOT	Processor B Boot Config. Configures the boot source for Processor B.

Table continues on the next page...

Field	Function
	These bits are only available on the Processor A side. 00b - Boot from 0x0 01b - Boot from DMEM Base 10b - Boot from IMEM Base 11b - Reserved
8 CLKE	MUB clock enable When set, forces MUB platform clock to remain enabled even after MUB has entered a stop mode. When clear, allows MUB to gate its platform clock whenever desired after entering a stop mode. Note that if MUA also enters a stop mode, then this bit is overidden to allow MUB to gate its platform clock. 0b - MUB platform clock gated when MUB enters a stop mode. 1b - MUB platform clock kept running after MUB enters a stop mode, until MUA also enters a stop mode.
7 RSTH	Processor B Reset Hold When set, will cause Processor B to be held in reset following any reset event. When cleared, allows Processor B to be released from reset. Setting this bit will not force Processor B into reset. It will only hold Processor B if it resets itself. This bit is only available on the Processor A side. 0b - Release Processor B from reset 1b - Hold Processor B in reset
6 RDIE	RDIE Processor B Reset De-assertion Interrupt Enable. (Read-Write) • RDIE bit enables Processor A General Interrupt 3. • If RDIE bit is set to "1", then General Interrupt 3 request is issued to the Processor A when the RDIP bit in the MUA SR register is set to "1". • If RDIE is cleared, then the value of the RDIP bit is ignored and no General Interrupt 3 request will be issued. • The RDIE bit is cleared when the MU resets. This bit is only available on the Processor A side. 0b - Disables Processor A General Purpose Interrupt 3 request due to Processor B reset de-assertion. 1b - Enables Processor A General Purpose Interrupt 3 request due to Processor B reset de-assertion.
5 MUR	MUR MU Reset. • Setting MUR bit to "1" resets both the Processor A and the Processor B sides of the MU module, forcing all control and status registers to return to their default values and all internal states to be cleared. • The BOOT and RSTH fields will not be affected by the MUR bit. • Before setting the MUR bit to "1", it is advisable to interrupt the Processor B , because setting the MUR bit may affect the ongoing Processor B program. • After setting the MUR bit, you should monitor the value of the RS bit in the MUA SR register to know when the reset sequence on the Processor B-side has ended. • MUR bit can only be written as "1". • MUR bit is always read as "0". • MUR bit is cleared during the MU reset sequence. This bit is only available on the Processor A side. 0b - N/A. Self clearing bit (default). 1b - Asserts the MU reset.

Table continues on the next page...

Field	Function
4 BHR	BHR Processor B Hardware Reset. (Read-Write) • BHR bit asserts and de-asserts the hardware reset of the Processor B. • Set BHR bit to "1" to start a hardware reset of the Processor B. • Clear the BHR bit to de-assert the Processor B hardware reset input. • Assert the BHR bit for a minimum of 3 clock cycles. The RS bit in MUA SR register indicates the state of the Processor B. As soon as the Processor B goes into Reset (MUA RS bit is set to "1"), the BHR bit can be de-asserted. • Strobe-setting the BHR bit will not cause an internal MU reset but will be routed outside MU to Processor B domain reset logic • After clearing the BHR bit, monitor the value of the RS bit at the MUA SR to know when the Processor B reset sequence has ended. • The BHR reset issued by the Processor A to the Processor B is maskable by the Processor B (according to the settings of the BHRM bit in the BCR register). If the BHRM bit (in the BCR register) is set to "1", then the BHR reset is masked; if the BHRM bit (in the BCR register) is cleared (default), then the BHR reset is enabled. • The BHR bit does not return to the reset value during the software (MUR) reset. 0b - De-assert Hardware reset to the Processor B. (default) 1b - Assert Hardware reset to the Processor B.
3 NMI	NMI MUB Non-maskable Interrupt. (Read-Write) • When NMI bit is set to "1", it initiates a Non-Maskable Interrupt to the Processor B. • NMI bit is cleared by the MU after the MUB asserts the NMIC bit in the MUB SR register. • After the NMI bit is cleared, the MUA can initiate another non-maskable interrupt to the MUB. • The NMI bit is cleared when the MU resets. 0b - Non-maskable interrupt is not issued to the Processor B by the Processor A (default). 1b - Non-maskable interrupt is issued to the Processor B by the Processor A.
2-0 Fn	Fn For n = {0, 1, 2} MUA to MUB Flag n. (Read-Write) • Fn bit is a read-write flag that is reflected in Fn bit in the MUB SR register on the MUB side. • Fn bit is cleared when the MU resets. 000b - Clears the Fn bit in the SR register. 001b - Sets the Fn bit in the SR register.

9.3.2 MUB register descriptions

This section contains the detailed register descriptions for the MUB registers.

9.3.2.1 MU Memory map

MUB base address: 4022_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	RO	0100_0001h
4h	Parameter Register (PAR)	32	RO	0000_0000h
20h - 2Ch	Transmit Register (TR0 - TR3)	32	RW	0000_0000h
40h - 4Ch	Receive Register (RR0 - RR3)	32	RO	0000_0000h
60h	Status Register (SR)	32	W1C	00F0_0080h
64h	Control Register (CR)	32	RW	0000_0000h

9.3.2.2 Version ID Register (VER)

9.3.2.2.1 Offset

Register	Offset
VER	0h

9.3.2.2.2 Function

Use Version ID register to determine the version ID and feature set number of MUB.

9.3.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

9.3.2.2.4 Fields

Field	Function
31-24	Major Version Number

Table continues on the next page...

Field	Function
MAJOR	This read only field returns the major version number for MUB.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for MUB.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number for MUB. 000000000000x1xxb - Core Control and Status Registers are implemented in both MUA and MUB. 000000000000xx1xb - RAIP/RAIE register bits are implemented. 000000000000xxx0b - Standard features implemented

9.3.2.3 Parameter Register (PAR)

9.3.2.3.1 Offset

Register	Offset
PAR	4h

9.3.2.3.2 Function

Use Parameter register to determine the parameter settings of MUB.

9.3.2.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PARAMETER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PARAMETER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.2.3.4 Fields

Field	Function
31-0 PARAMETER	This bitfield contains the parameter settings of MUB.

9.3.2.4 Transmit Register (TR0 - TR3)

9.3.2.4.1 Offset

Register	Offset
TR0	20h
TR1	24h
TR2	28h
TR3	2Ch

9.3.2.4.2 Function

Use MUB Transmit Register (TRn, 32-bit, write-only) to transmit a message or data to MUA.

- You can only write to the TRn register when the TEn bit in SR register is set to "1".
- Reading the TRn register returns all zeros.

9.3.2.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.2.4.4 Fields

Field	Function
31-0 DATA	DATA MUB Transmit Register Data. (Write-only) • Data written to the TRn register is reflected in the MUA Receive Register n (RRn). The TRn and RRn registers are not double-buffered, a write to the TRn register overrides the data readable at the RRn register.

Field	Function
	• A write to the transmit register clears a "transmitter empty" bit (TEn) in the MUB Status Register (SR) on the transmitter side, and sets a "receiver full" bit (RFn) in the MUA Status Register (SR) on the receiver side (optionally triggering an interrupt 0 on the MUB side). • Any write to the TRn register will update all status information.

9.3.2.5 Receive Register (RR0 - RR3)

9.3.2.5.1 Offset

Register	Offset
RR0	40h
RR1	44h
RR2	48h
RR3	4Ch

9.3.2.5.2 Function

Use MUB Receive Register (RRn, 32-bit, read-only) to receive a message or data from the MUA.

- Data written to the MUA TRn register is immediately reflected in the MUB RRn register.
- You can only read the RRn register when the RFn bit in the SR register is set to "1".
- Writing to the RRn register generates an error response to the MUB.

9.3.2.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9.3.2.5.4 Fields

Field	Function
31-0 DATA	DATA MUB Receive Register. (Read-only) • Reflects the data written to MUA Transmit Register 0 (TRn). • Reading the RRn register clears the "receiver full" bit (RFn) in the MUB Status Register (SR) on the receiver side, and sets the "transmitter empty" bit (TEn) in the MUA Status Register on the transmitter side (optionally triggering a transmit interrupt 0 on the MUA side). • Any read of the RRn register will update all status information.

9.3.2.6 Status Register (SR)

9.3.2.6.1 Offset

Register	Offset
SR	60h

9.3.2.6.2 Function

Use the Processor B Status Register (SR, 32-bit, read-write) to show interrupt status from the Processor A, general purpose flags, the Processor A power mode, and to set dual function control-status bits.

- Some dual-purpose bits are set by the MU logic, and cleared by the Processor B-side programmer
- Other dual-purpose bits are set by the Processor B-side programmer, and cleared by the MU logic.

9.3.2.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GIPn				RFn				TEn				Reserved			
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved			Reserved	Reserved	Reserved	FUP	RS	PM		EP	NMIC	Fn		
W													W1C			
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

9.3.2.6.4 Fields

Field	Function
31-28 GIPn	GIPn For n = {0, 1, 2, 3} MUB General Interrupt Request n Pending. (Read-Write) • GIPn bit signals the MUB that the GIRn bit in the CR register on the MUA side was set from "0" to "1". If the GIEn bit in the MUB CR register is set to "1", a General Interrupt n request is issued. • The GIPn bit is cleared by writing it back as "1". Writing "0", or writing "1" when the GIPn bit is cleared is ignored. Use this feature in the interrupt routine, where the GIPn bit is cleared in order to de-assert the interrupt request source at the interrupt controller. An example of a proper bit clearing sequence is: clear MUB register, set the desired bit, and write it to the MUB SR register, thus clearing the GIPn bit. • GIPn bit is cleared when the MU is reset. 0000b - MUB general purpose interrupt n is not pending. (default) 0001b - MUB general purpose interrupt n is pending.
27-24 RFn	RFn For n = {0, 1, 2, 3} MUB Receive Register n Full. (Read-only) • The RFn bit is set to "1" when the MUA TRn register is written on the MUA side. • After the RFn bit is set to "1", the RFn bit signals the MUB side that new data is ready to be read by the MUB in the MUB RRn register, and a Receive n interrupt is issued on the MUB side (if the RIEn bit in the MUB CR register has been set to "1"). • RFn bit is cleared when the MUB RRn register is read, and when the MU is reset. 0000b - MUB RRn register is not full (default). 0001b - MUB RRn register has received data from MUA TRn register and is ready to be read by the MUB.
23-20 TEn	TEn For n = {0, 1, 2, 3} MUB Transmit Register n Empty. (Read-only) • The TEn bit is set to "1" after the MUA RRn register is read on the MUA side. • After the TEn bit is set to "1", the TEn bit signals the MUB side that the MUB TRn register is ready to be written on the MUB side, and a Transmit n interrupt is issued on the MUB side (if the TEn bit in the MUB CR register is set to "1"). • TEn bit is cleared after the MUB TRn register is written on the MUB side. • TEn bit is set to "1" when the MU is reset. 0000b - MUB TRn register is not empty. 0001b - MUB TRn register is empty (default).
19-15 —	Reserved
14-12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved

Table continues on the next page...

Field	Function
8 FUP	FUP MUB Flags Update Pending. (Read-only) • FUP bit is set to "1" when the MUB side sends a Flags Update request to the MUA side. • A Flags Update request is generated when there is a change to the Fn[2:0] bits of the MUB CR register. No flag update changes are allowed while the FUP bit is set to "1". Any write to the Fn[2:0] bits of the MUB CR register, while the FUP bit is set to "1", will not generate a Flags Update event, and the Fn[2:0] bits will stay unchanged. • FUP bit is cleared when this Flags Update request is internally acknowledged (that the flag is updated) from the MU MUA side, and during MU reset. 0b - No flags updated, initiated by the MUB, in progress (default) 1b - MUB initiated flags update, processing
7 RS	RS MUA Reset State. (Read-only) • RS bit indicates if the MUA side of the MU is in a reset state or not. • If the RS bit is set to "1", then the MUA side of the MU is still in the reset state. • If the RS bit is cleared, then the MUA side of the MU are out of reset. • The RS bit is set to "1" during: a MUA system reset, or an MU reset (caused by setting the MUR bit at the CR register). • The RS bit is cleared when the reset sequence on the MUA side of the MU ends. After issuing any of the reset events mentioned previously, you should verify that the RS bit is cleared before starting any accesses. • When the MUA processor comes out of reset, the RS bit has value "1"(default). 0b - The MUA side of the MU is not in reset. 1b - The MUA side of the MU is in reset.
6-5 PM	PM Processor A-side Power Mode. (Read-only) • PM[1:0] bits indicate the Processor A-side power mode. **NOTE:** The Processor A Power Mode is platform-specific. 00b - The MUA processor is in Run Mode. 01b - The MUA processor is in WAIT Mode. 10b - The MUA processor is in STOP/VLPS Mode. 11b - The MUA processor is in LLS/VLLS Mode.
4 EP	EP MUB Side Event Pending. (Read-only) • EP bit is set to "1" when the MUB side mechanism sends an event update request to the MUA side. • EP bit is cleared when the event update acknowledge is received. An "event" is any hardware message that is reflected in the MUA SR register on the MUA side (for example, "transmit register 0 written"). During normal operations, you do not have to deal with the state of the EP bit because the event update mechanism works automatically. • To ensure events have been posted to MUA before entering STOP mode, you should verify that the EP bit is cleared. If EP bit is set to "1", you should wait and continue to poll it (EP bit) before entering STOP mode. • Reading the MUB SR register (to check the EP bit) should be the last access to the MU that should be performed before entering STOP or WAIT modes; otherwise, the EP bit may be set by subsequent additional actions. • The EP bit is cleared when the MU resets. 0b - The MUB side event is not pending (default). 1b - The MUB side event is pending.
3	NMIC

Table continues on the next page...

Field	Function
NMIC	Processor B Non-Maskable-Interrupt Clear. (Write-only) • NMIC bit is used by the Processor B-side MU Non-Maskable Interrupt service routine to clear the non-maskable interrupt. • Writing NMIC bit as "1" signals the MU to clear the NMI bit in the MUA CR register, thus de-asserting the interrupts and enabling the NMI bit to receive another interrupt. • The NMIC bit is always read as "0"; therefore it (NMIC bit) cannot be polled. The NMIC bit can only be used as part of the interrupt service routine, in which the NMIC bit should only be written as "1" once. • The NMIC bit is cleared when the MU resets. 0b - Default 1b - Writing "1" clears the NMI bit in the MUA CR register.
2-0 Fn	Fn For n = {0, 1, 2} MUB Side Flag n. (Read-only) • Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA control register. • Every time that the Fn bit in the MUA CR register is written, the Fn bit in the MUA CR register write event updates the Fn bit in the MUB SR register after the event update latency, which is measured in terms of the number of clocks of the MUA and the MUB. 000b - Fn bit in the MUA CR register is written 0 (default). 001b - Fn bit in the MUA CR register is written 1.

9.3.2.7 Control Register (CR)

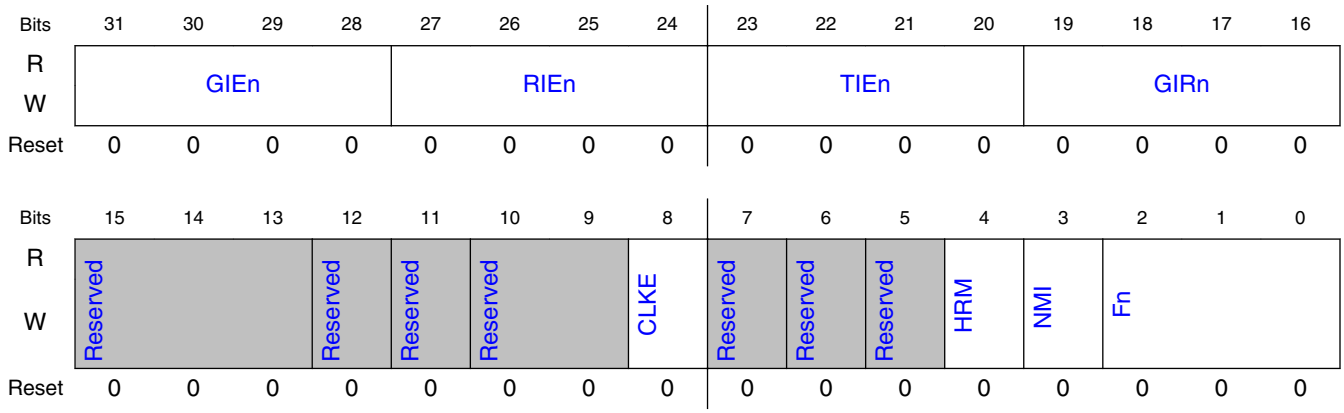
9.3.2.7.1 Offset

Register	Offset
CR	64h

9.3.2.7.2 Function

Use the Control Register (CR, 32-bit, read-write) to enable the MU interrupts on the MUB-side, and trigger events and interrupts on the MUA-side (general purpose interrupt, flag update).

9.3.2.7.3 Diagram



9.3.2.7.4 Fields

Field	Function
31-28 GIEn	GIEn For n = {0, 1, 2, 3} MUB General Purpose Interrupt Enable n. (Read-Write) • GIEn bit enables MUB General Interrupt n. • If GIEn bit is set to "1" (enabled), then a General Interrupt n request is issued when the GIPn bit in the MUB SR register is set to "1". • If GIEn is cleared (disabled), then the value of the GIPn bit is ignored and no General Interrupt n request will be issued. • GIEn bit is cleared when the MU resets. 0000b - Disables MUB General Interrupt n. (default) 0001b - Enables MUB General Interrupt n.
27-24 RIEn	RIEn For n = {0, 1, 2, 3} MUB Receive Interrupt Enable n. (Read-Write) • RIEn bit enables MUB Receive Interrupt n. • If RIEn bit is set to "1" (enabled), then an MUB Receive Interrupt n request is issued when the RFn bit in the MUB SR register is set to "1". • If RIEn bit is cleared (disabled), then the value of the RFn bit is ignored and no MUB Receive Interrupt n request will be issued. • RIEn bit is cleared when the MU resets. 0000b - Disables MUB Receive Interrupt n. (default) 0001b - Enables MUB Receive Interrupt n.
23-20 TIEn	TIEn For n = {0, 1, 2, 3} MUB Transmit Interrupt Enable n. (Read-Write) • TIEn bit enables MUB Transmit Interrupt n. • If TIEn bit is set to "1" (enabled), then an MUB Transmit Interrupt n request is issued when the TEn bit in the MUB SR register is set to "1". • If TIEn bit is cleared (disabled), then the value of the TEn bit is ignored and no MUB Transmit Interrupt n request will be issued. • TIEn bit is cleared when the MU resets. 0000b - Disables MUB Transmit Interrupt n. (default) 0001b - Enables MUB Transmit Interrupt n.

Table continues on the next page...

Field	Function
19-16 GIRn	GIRn For n = {0, 1, 2, 3} MUB General Purpose Interrupt Request n. (Read-Write) • Writing "1" to the GIRn bit sets the GIPn bit in the MUA SR register on the Processor A-side. If the GIEn bit in the MUA CR register is set to "1" on the MUA side, a General Purpose Interrupt n request is triggered. • The GIRn bit is cleared if the GIPn bit (in the MUA SR register on the MUA side) is cleared by writing it (GIPn bit) as "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software). The GIPn bit cannot be written as "0" on the MUB side. • To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit). • GIRn bit is cleared when the MU resets. 0000b - MUB General Interrupt n is not requested to the MUA (default). 0001b - MUB General Interrupt n is requested to the MUA.
15-13 —	Reserved
12 —	Reserved
11 —	Reserved
10-9 —	Reserved
8 CLKE	MUA clock enable When set, forces MUA platform clock to remain enabled even after MUA has entered a stop mode. When clear, allows MUA to gate its platform clock whenever desired after entering a stop mode. Note that if MUB also enters a stop mode, then this bit is overidden to allow MUA to gate its platform clock. 0b - MUA platform clock gated when MUA enters a stop mode. 1b - MUA platform clock kept running after MUA enters a stop mode, until MUB also enters a stop mode.
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 HRM	HRM Processor B Hardware Reset Mask. (Read-Write) • The Processor A can give a hardware reset to the Processor B by setting the BHR bit in the MUA CR Register to “1”. • When the BHRM bit is set to “1” by the Processor B, the BHR reset issued by the Processor A is masked (disabled by the Processor B). • When the BHRM bit is cleared, the BHR reset issued by the Processor A to the Processor B is not masked (enabled by the Processor B). 0b - BHR bit in MUA CR is not masked, enables the hardware reset to the Processor B (default after hardware reset). 1b - BHR bit in MUA CR is masked, disables the hardware reset request to the Processor B.
3	NMI

Table continues on the next page...

Field	Function
NMI	MUA Non-maskable Interrupt. (Read-Write) • When NMI bit is set to "1", it initiates a Non-Maskable Interrupt to the Processor A. • NMI bit is cleared by the MU after the MUA asserts the NMIC bit in the MUA SR register. • After the NMI bit is cleared, the MUB can initiate another non-maskable interrupt to the MUA. • The NMI bit is cleared when the MU resets. 0b - Non-maskable interrupt is not issued to the Processor A by the Processor B (default). 1b - Non-maskable interrupt is issued to the Processor A by the Processor B.
2-0 Fn	Fn For n = {0, 1, 2} MUB to MUA Flag n. (Read-Write) • Fn bit is a read-write flag that is reflected in Fn bit in the MUA SR register on the MUA side. • Fn bit is cleared when the MU resets. 000b - Clears the Fn bit in the SR register. 001b - Sets the Fn bit in the SR register.

9.4 Functional Description

Table 9-3. Major Features of the MU

Major Feature	Description
Interprocessor Interrupts	• The MU has 12 interrupt sources on each side (Processor A-side, Processor B-side) that are used for signaling the other processor. The interrupts can be used for notification of RX/TX events and general-purpose signaling between the processors.
MU Reset	• The Processor A can issue a reset to the entire MU, using a control bit (MUR) in the Processor A Control Register (CR). • The MUR bit is a self-clearing bit.

Processor B Boot Configuration	• The Boot Source for Processor B can be configured with the BOOT bits in the CR register.
	• Boot Source Options are:
	  • DMEM Base Address
	  • IMEM Base Address
	  • Address 0x00

Processor B Reset Hold	• Processor B can be held in reset following any reset event. This is done by setting the RSTH bit in the CR register. • Processor B will be released from reset when this bit is cleared. • The value at reset is loaded from Flash IFR.
Processor A/B Clock Enable	• The Processor A/B platform clock can be enabled to continue running when Processor A/B enters Stop Mode, until Processor B/A also enters Stop Mode. This allows Processor B/A to continue accessing peripherals on Processor A/B's AIPS bus even when it has entered a Stop Mode.
Status and Control Communications between Cores	• The MU provides a way for the two cores to communicate using the status and control registers present on both the Processor B and Processor A sides of the MU. • The status register of one MU side reflects the status of the other MU side. • The control register is used for control operations, such as enabling an interrupt and sending an interrupt to the other processor.

Table continues on the next page...

Table 9-3. Major Features of the MU (continued)

Major Feature	Description
Synchronized Message Transfers between Cores	• The transfer of data messages between cores uses transmit empty and receive full flags provided on both sides of the MU. • The update of these transmit and receive flags is accomplished using a synchronization mechanism. There is inherent latency between updating the flag on one side and reflecting its status on other side. For more about latency, see Event Update Timing
Accessing Shared Memory Directly and Avoiding Collisions	• For sending data or messages from one MU-side to the other MU-side, the MU provides 4 transmit registers and 4 receive registers on each side of the MU. • The Processor A or Processor B can access shared memory resources of the SoC directly. However, to avoid simultaneous access to shared memory by both cores, the MU provides a method (to prevent simultaneous access) using interrupts and transmit-receive registers for both processors.
Memory-Mapped Registers	• The MU is connected as a peripheral under the Peripheral bus on both sides—on the Processor A-side, the Processor A Peripheral Bus, and on the Processor B-side, the Processor B Peripheral Bus.

9.4.1 Processor A Side Memory-Mapping

The messaging, control, and status registers of the Processor A-side for the MU are mapped to the Processor A memory as a regular peripheral. The Peripheral bus data bus is 32 bits wide inside the MU module.

9.4.2 Processor B Side Memory-Mapping

The messaging, control, and status registers of the Processor B-side for the MU are mapped to the Processor B memory as a regular peripheral. The Peripheral bus data bus is 32 bits wide inside the MU module.

9.4.3 MU Messaging

The MU provides 32-bit status and control registers to the Processor B and Processor A sides for control operations (such as interrupts and reset), and for status checking of the other MU-side.

For messaging, the MU has four, 32-bit write-only transmit registers and four, 32-bit read-only receive registers on the Processor B and Processor A-sides. These registers are used for sending messages to each other. These messages can be also be controlled using the 3 general purpose flags provided in the control and status registers of either MU-side.

9.4.3.1 Programmer Model

The messaging logic is used in conjunction with external memory. You have various messaging methods, which you can use to implement a messaging protocol. Some of these messages could mean “I have just written a message of N words, starting at offset X in the memory,” or “I have just finished reading the previous data block that was sent.” Having the messaging logic independent from the memory array does not restrict you to a predefined hardware protocol. On the other hand, the software needed to manage the messaging is short and straightforward.

Most of the messaging mechanisms are symmetric; they are duplicated and are available on both the Processor B-side and the Processor A-side. The messaging mechanisms are:

- Four, 32-bit write-only transmit registers, which are each reflected in four, read-only receive registers in the other processor’s side. You can use these registers to transfer 32-bit word messages or frame information of messages written to the shared memory (number of words, initial address, and message type code).
- A write to a transmit register on the transmitter side clears a “transmitter empty” bit in the Status Register on the transmitter side, and sets a “receiver full” bit in the Status Register on the receiver side. The setting of the bit at the receiver side can optionally trigger an interrupt at the receiver side (maskable receive interrupt).
- A read of one of the receive registers at the receiver side clears the “receiver full” bit in the Status Register at the receiver side, and sets the “transmitter empty” bit in the Status Register on the transmitter side. The setting of the “transmitter empty” bit can optionally trigger an interrupt at the transmitter side (maskable transmit interrupt).
- Four general purpose flags are reflected in the Status Register on the receiver side
- A read/write access to any reserved location and a write to a read-only register on the Processor A-side of the MU will generate a module transfer error acknowledge to the Processor A.
- A read/write access to any reserved location and write to a read-only register on the Processor B-side of the MU will generate a module transfer error acknowledge to the Processor B.

9.4.3.2 Messaging Examples

The following are messaging examples:

- **Passing short messages:** Transmit register(s) can be used to pass short messages from one to four words in length. For example, when a four-word message is desired, only one of the registers needs to have its corresponding interrupt enable bit set at the

receiver side; the message's first three words are written to the registers whose interrupt is masked, and the fourth word is written to the other register (which triggers an interrupt at the receiver side).

- **Passing frame information:** Transmit registers can be used to pass frame information for long messages written to the shared system (SDRAM and SyncFLASH). Such frame information normally includes a start address, number of words, and perhaps a message type code.
- **Passing event notices and requests:** Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the general interrupts, such as acknowledging that a long message was read from the shared system memory.
- **Passing fixed length data:** Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general interrupt (Processor A or Processor B) to signal the other processor that the data is ready.
- **Passing announcements:** The three flags can be used by a processor to announce its current program state or other billboard messages to the other processor.

Figure 9-2 shows the MU registers schematic.

9.4.4 Low Power Modes

This section describes the low power operating modes of the MU module.

9.4.4.1 Processor Low Power Modes

The Processors have four power modes:

- Run
- WAIT
- STOP/VLPS
- LLS/VLLS

The Processor can be awakened from a low-power mode by any enabled Processor side MU interrupt, as reflected in the xSR "status" register (RF0–3, TE0–3, GIP0–3 bits are set) and enabled in the xCR control register. Using these bits, the Processor can actively control when to wake the other Processor.

While the Processor is in STOP/VLPS mode (such that the xSR register bits cannot be updated with events), special logic drives the enabled Processor interrupts directly from the other Processor-side (instead of from the xSR register).

While the Processor is in STOP/VLPS mode, the asynchronous Processor interrupt will be asserted to wake the Processor:

- If any transmit data register of the other Processor-side is full, because of a write to it (transmit data register); that is, its “empty” bit in the xSR register is cleared while its corresponding receive interrupt is enabled on the Processor-side.
- If any receive data register of the other Processor-side is empty, because of a read on the other Processor -side; that is, its “full” bit in the xSR register is cleared while its corresponding transmit interrupt is enabled on the Processor-side.
- If any general purpose interrupt is set in the xCR register while the corresponding interrupt is enabled on the Processor-side.
- If the other Processor issues a non-maskable interrupt to the Processor.

The logic enables the other Processor to operate independently while the Processor is in any power mode (including STOP/VLPS). However, the Processor power mode change protocol should be handled with care regarding:

- The interrupts that are enabled on the Processor-side
- The events that could be triggered by the other Processor-side
- The compatibility with the other Processor protocol of entering STOP/VLPS mode

If the Processor is in STOP/VLPS mode and an event on the other Processor is triggered, the EP bit (in the xSR register) will remain high until the Processor wakes up.

Before entering STOP/VLPS mode, the Processor programmer should verify that the EP bit (in the xSR register) is cleared. This check is needed to ensure that all pending updates from the Processor, including the power mode change when STOP/VLPS or WAIT is executed, will be updated in the xSR register.

- If the other Processor is in STOP/VLPS mode or LLS/VLLS mode, the EP bit (in the xSR register) may be stuck high; in this case, the Processor need not check the EP bit before entering STOP/VLPS mode.

9.4.5 Event Update Timing

Each processor’s MU messaging side (Processor A or Processor B) has a hardware mechanism to send “event update requests” to the other processor’s side. An “event” is considered when any information change should be reflected at the Status Register of the receiving processor. The event update latency is the delay between the event being ready at one processor and the resulting update at the Status Register of the other processor.

- The minimum event latency is “1 clock of the sending side” + “2 1/2 clocks of the receiving side”. The minimum case is if there is no event pending when the new event occurs.
- The maximum event latency is “6 clocks of the sending side” + “6 1/2 clocks of the receiving side.” The maximum case is if the event occurred just after a previous event was sent to the other side. The event update latency will vary between the above-mentioned minimum and maximum latencies, depending on the time at which the subsequent event is triggered.

9.4.6 Interrupts

The MU controls the Processor B interrupt requests to the Processor A, and the Processor A interrupt requests to the Processor B.

This section describes all the interrupts that the module generates.

9.4.6.1 Interrupts to the Processors

There are 12 interrupt sources from the MU to the Processors:

- Four receive interrupts (asserted when the Processors receive full bits are set and enabled in the xCR register) for each of the receive registers
- Four transmit interrupts (asserted when the Processor transmit empty bits are set and enabled in the xCR register) for each of the transmit registers
- Four general purpose interrupts (asserted when the GIP bits are set and enabled in the xCR register)

All the interrupts are maskable in the Processor Control Register (xCR). The MU does not assume any internal priority of these interrupts. Multiple interrupts (for example, Receive 0 and Receive 1 interrupts or any of the transmit and general purpose interrupts) can be asserted at one time. The priority of these interrupts should be resolved by the interrupt controller at the chip level.

The General Purpose Interrupt Pending bits (GIP0, GIP1, GIP2, and GIP3) should be cleared by the software (as part of the interrupt service routine) to de-assert the request to the interrupt controller.

9.4.6.2 General Purpose Interrupt Clearing Sequence

When a Processor writes to the general interrupt bit (GIR), the write event is synchronized to the other Processor clock to set the general interrupt request pending bit (GIP). When the GIP bit is set, and if the general purpose interrupt is enabled on the transmitting Processor side (GIE bit is set), then the receiving Processor general purpose interrupt is issued to the transmitting Processor. The transmitting Processor clears this interrupt by writing a “1” on the GIP bit. The interrupt is de-asserted as soon as the GIP bit is written. The write event of the GIP bit is synchronized to the other Processor clock. The synchronized signal clears the GIR bit. The software should not write the GIR bit again until the GIR bit is cleared.

9.4.7 Interrupt Messaging Protocols

9.4.7.1 Messaging Protocols using Interrupts

The example below describes a four-word messaging sequence sent by the Processor to the other Processor.

In this example, the first, second, and third receive interrupts are disabled, and the fourth receive interrupt is enabled. We write registers sequentially for n = 0, 1, 2, 3. For n = 0, 1, 2, the interrupts are disabled, therefore no interrupt will go to the other core (although interrupt conditions occur). For n = 3, the interrupt is enabled, and the last Receive Interrupt request is generated.

1. Write Sequence
   - The Processor writes the message information sequentially to its Transmit Registers 0, 1, 2.
   - When the write to the Transmit Register 3 occurs, the RF3 bit of the xSR is set after synchronization, and it immediately trigger the Receive Full 3 interrupt to the other Processor.
2. Read Sequence
   - The other Processor receives the Receive Full 3 interrupt and starts reading the message transferred from the receive registers.
   - After Receive Register 3 is read, the interrupt bit is cleared.

The following table and diagram describe the messaging model using transmit/receive registers and interrupt messaging protocol.

Table 9-4. Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A Data write	A data write to the TRn register by Processor A is immediately reflected in the Processor B RRn register.
2	Clear Tx Empty bit and Set Rx Full bit	The data write to the TRn register • Clears the transmitter empty bit (TEn) in the Processor A Transmit Status Register • Sets the receiver full bit (RFn) in the Processor B Receive Status Register
3	Generate Receive Interrupt request	The setting of the receiver full bit (RFn) in the Receive Status Register generates a Receive Interrupt request to Processor B.
4	Processor B Data read	After receiving the Receive Interrupt request, Processor B performs a data read of the RRn register.
5	Clear Rx Full bit and Set Tx Empty bit	Reading the data out of the RRn register • Clears the receiver full bit (RFn) in the Processor B Receive Status Register • Sets the transmitter empty bit (TEn) in the Processor A Transmit Status Register
6	Generate Transmit Interrupt request	The setting of the transmitter empty bit (TEn) in the Transmit Status Register generates a Transmit Interrupt request to Processor A.

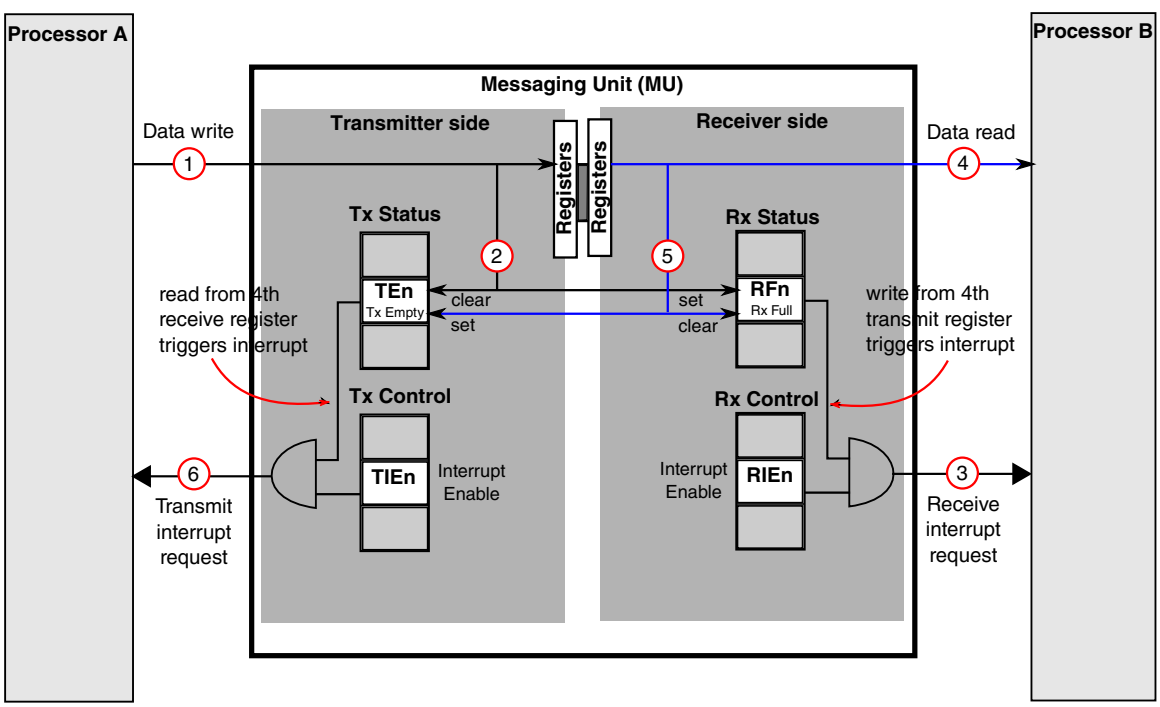


Figure 9-3. Messaging Model Using Transmit and Receive Registers

The messaging hardware can be used by software to implement messaging protocols for a wide array of message types. Full support is given for both interrupt and polling management schemes.

9.4.7.2 Messaging Protocols using Event Interrupts

Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the two general interrupts.

Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general purpose interrupt to signal the other processor that the data is ready.

The three flags can be used by a processor to announce to the other processor the program state it is currently in, or to announce similar messages.

Table 9-5 and Figure 9-4 describe the event sequence when the Processor triggers an interrupt.

Table 9-5. Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A sets General Interrupt request bit	Processor A sets its associated General Interrupt request bit (GIRn = 1) in the control register (ACR).
2	General Interrupt Request Pending status bit is set	The General Interrupt Request Pending status bit (GIPn) in the status register (BSR) is set to "1"
3	General Interrupt request to Processor B is generated	Setting the GIPn bit generates the General Interrupt request to Processor B (Interrupt Request Enable bit, GIEn, must be set for Processor B)
4	Processor B reads status register	The Processor B reads the GIPn bit in the BSR register.
5	Processor B services the interrupt	-
6	Processor B sets GIPn bit to clear interrupt	The Processor B writes "1" to the corresponding GIPn bit to clear the interrupt
7	GIRn bit is cleared	Setting the GIPn bit to "1" clears the General Interrupt request bit (GIRn) in the Processor A control register (ACR).

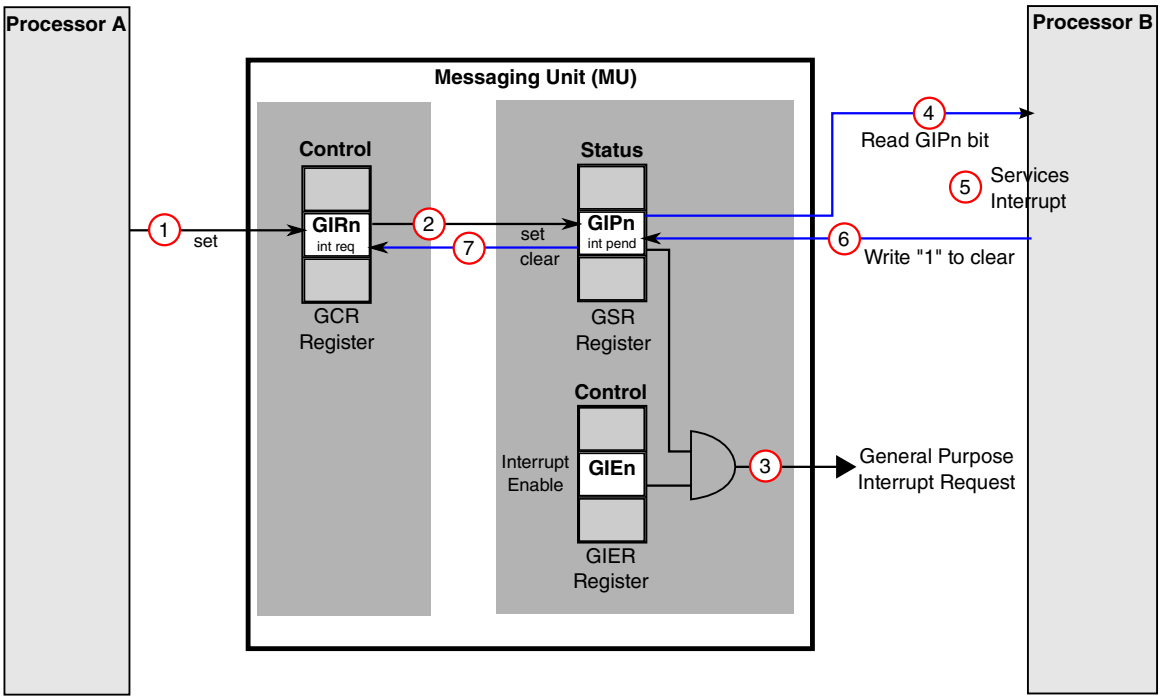


Figure 9-4. Messaging Model Using a General Purpose Interrupt

9.4.8 Exclusive Access to Shared Memory

You can use the MU to signal one processor about its current access to the shared memory, so that the data is not overwritten by the other processor during the exclusive memory access period.

The following tables describe the signaling protocol that the Processor A uses to inform the Processor B about its current access (write) to the shared memory, assuming that the set of bits and registers (GIRn bit, RRn register, TRn register) are reserved to support exclusive access to the shared memory protocol.

Table 9-6. How the Processor A Performs an Exclusive Access to Shared Memory

Sequence	Action	Description
1	Processor A sends GIRn request to Processor B using Processor A control register	When the Processor A wants to perform an exclusive access to the shared memory, the Processor A sends an GIR0 request to the Processor B.
2	Processor A sends an exclusive-access request using a transmit data register (TRn)	The Processor A will send an exclusive-access request (command, location, and length of target access) to Processor B using a selected transmit data register (TR0).
3	Processor A waits for a dedicated interrupt from Processor B	The Processor A waits for a dedicated interrupt (as an acknowledgement) triggered by the Processor B before proceeding.

Table continues on the next page...

Table 9-6. How the Processor A Performs an Exclusive Access to Shared Memory (continued)

Sequence	Action	Description
4	Processor A accesses shared memory	After receiving a dedicated interrupt from the Processor B, Processor A proceeds.

Table 9-7. How the Processor B Scans for Transaction Information

Sequence	Action	Description
1	Processor B receives an interrupt from a receive data register (RRn)	-
2	Processor B reads the receive data register (RRn)	-
3	Processor B scans the receive data register contents	For transaction information (whether Processor A has requested an exclusive-access)

Table 9-8. How the Processor B Accepts Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B triggers a dedicated interrupt	Processor B acknowledges the Processor A request by triggering a dedicated interrupt (ack) to the Processor A.
2	Processor B sends a code message to Processor A	Along with the acknowledge interrupt, the Processor B sends a code message to the Processor A through the selected transmit register (TRn). The message informs the Processor A that it can exclusively access the shared memory.

Table 9-9. How the Processor B Rejects Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B ignores Processor A request for exclusive access	If the Processor B does not want to give go-ahead permission to the Processor A, Processor B ignores the exclusive access request.

9.4.9 Packet Data Transfers

The following example describes the packet transfer sequence between the Processor B and Processor A subsystems:

Table 9-10. Packet Data Transfer Sequence

Action	Sequence	Description
Processor B requests DMA	1	The Processor B sends a DMA request to initiate the packet data transfer

Table continues on the next page...

Table 9-10. Packet Data Transfer Sequence (continued)

Action	Sequence	Description
DMA data transfer	2	DMA acknowledges.
	3	DMA starts transferring data from the specified Processor B location to the specified shared memory
	4	DMA interrupts the Processor B to signal that the packet transfer has finished.
Processor B informs Processor A that data is in shared memory	5	Using an MU Processor B-side transmit register, the Processor B sends a packet information message to the Processor A to inform the Processor A of the arrival of new packet data that is stored in shared memory. The message contains the command, location, and length of packet data information.
Processor A receives interrupt	6	The Processor A receives an interrupt (assuming its corresponding Processor A MU-side receive interrupt is enabled), and the pending processing task becomes active and processes packet data from memory.
Processor A reads data, writes data	7	The Processor A reads or processes packet data from shared memory.
	8	The Processor A writes the result from packet processing to a separate buffer.
Processor A informs Processor B that transfer is finished	9	After the processing of the packet data finishes, the Processor A informs the Processor B (using the MU Processor A-side transmit register, ATRn).
Processor A sends interrupt to Processor B (request for more data)	10	The Processor B receives the next interrupt from the Processor A, in which the Processor A requests more packet data.

9.4.10 Resets

The MU has two sources of reset, and each reset has a different function from the MU or system perspective.

- One asynchronous system that is connected to both sides of the MU interface.
- One programmable hardware reset (MUR bit) in the CR register (on the Processor A-side).

Table 9-11. MU programmable resets

Reset	Description
Processor A MU reset	• Processor A MU Reset bit (MUR) of the CR register • The MUR reset affects the messaging section on both the Processor A and the Processor B sides. The MUR reset causes all control and status registers to return to their default values and all internal states to be cleared. • It is up to the Processor A software to decide whether to use the MUR reset or not. • The instruction immediately following assertion of the MUR bit should not write to MU registers. Such a write may be overwritten by the reset sequence and the register will remain with the reset value. You should wait at least one instruction (after assertion of the MUR bit) before attempting a write to MU registers.

After issuing MUR bit reset events, the Processor A programmer can verify that the reset sequence on the Processor B-side has ended, by checking the RS bit in the SR register.

NOTE

MUR bit assertion is a delicate operation because it affects the other side's registers asynchronously. MUR bit assertion may cause unpredictable behavior if, for example, the Processor B is concurrently testing an MU register bit (TE bit in Processor B SR register). Before asserting the MUR bit, you should verify that the Processor B is not presently engaged in an MU signalling activity.

9.5 Software Restrictions

This section describes certain software restrictions when accessing the MU.

9.5.1 General Restrictions

This section lists the restrictions that apply to both the sides (Processor A, Processor B) of the MU.

9.5.1.1 Write-After-Write to a Transmit Register

A write to a transmit register signals the receiver side that data is ready for retrieval.

- Writing to the transmit register again without verifying that the data was retrieved is prohibited, because the transmitter side has no way of knowing the exact time that the receiver will attempt to retrieve the data.
- Before attempting to write the transmit register again, the transmitter side should wait for a "Transmitter Empty" interrupt, or should poll the "Transmitter Empty" bit in the Status Register.
- Failure to follow this restriction may result in the wrong data being read on the receiver side of the MU.

9.5.1.2 Read-After-Read from a Receive Register

A read of a receive register signals the transmitter side that data can be written to that register. In the same way, the receiver processor should not read a receive register before receiving a "Receiver Full" interrupt or polling the "Receiver Full" bit in the Status Register.

- Reading the receive register again without verifying that the data was written is prohibited, because the receiver side has no way of knowing the exact time that the transmitter will attempt to write the data.
- Before attempting to read the receive register again, the receiver side should wait for a “Receiver Full” interrupt, or should poll the “Receiver Full” bit in the Status Register.
- Failure to follow this restriction may result in the wrong data being written on the transmitter side of the MU.

9.5.2 Processor Restrictions

This section lists the restrictions that apply each side of the processor in the MU.

9.5.2.1 Before Entering Low Power Mode

Before entering Low Power mode, the Processor should verify that the Processor Event Pending (EP) bit in the Status Register is cleared.

- If the Event Pending bit (EP) is still set to “1”, then the Processor should wait and poll the EP bit until it is cleared, before executing the LPM instruction.
- Note that if the other Processor is in Low Power mode (programmed for clock gating in CCM), the EP bit may be stuck high. In this case, the other Processor clock must be turned ON to get the EP bit cleared before the Processor can enter Low Power mode.
- To discover which power mode the other Processor is in, the Processor can check the PM bits in the xSR register.

9.5.2.2 Before Setting a General Interrupt Request Bit (GIR0–3)

Before setting a General Interrupt Request bit (GIR0–3), you must verify that the GIRn bit is cleared, which means that a general interrupt is not pending. Generally, setting the GIRn bit while the bit is set to “1” will be ignored, but in some cases it may issue a second interrupt. This restriction is meant to prevent this indeterministic behavior.

9.5.2.3 Reset Bit Restrictions

The reset bit (MUR, HR) restrictions are:

- Before asserting the MUR bit in the CR register, verify that the Processor B-side is not engaged in some MU activity.
- Do not write to an MU register in the instruction immediately after the assertion of the MUR bit in the CR register, because the written data can be overridden by the reset value.

Chapter 10
Hardware Semaphore2

10.1 Chip-specific SEMA42 information

Table 10-1. Reference links to related information

Topic	Related module	Reference
Full description	SEMA42	SEMA42
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

10.1.1 Hardware Semaphore2 (SEMA42)

The Hardware Semaphore (SEMA42) module provides the hardware support needed in multicore systems for implementing semaphores and provide a simple mechanism to achieve "lock/unlock" operations via a single write access. This approach eliminates architecture specific implementations like atomic (indivisible) read-modify-write instruction or reservation mechanism. The result is an architecture-neutral solution that provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.

Table 10-2. Hardware semaphore configuration

Parameter	Description
Name	SEMA42
Instances	2
Configurable features	SEMA42: • Support sparsely populated memory map (peripheral list)

Table continues on the next page...

Table 10-2. Hardware semaphore configuration (continued)

Parameter	Description
	• Number of protect peripherals = 16 • Limits access to cores as configured in XRDC peripheral settings
Interface speed	NA
External I/O pins	NA

10.2 Introduction

SEMA42 is a memory-mapped module that provides robust hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve "lock and unlock" operations via a single - write access. The hardware semaphore module provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.

10.2.1 Multi-core programming: software gates

Multi-processor systems require a function that can be used to safely and easily provide a locking mechanism for system software to control access to shared data structures, shared hardware resources, and so on. The software uses the gating mechanisms to serialize (and synchronize) accesses to shared data and/or resources to prevent race conditions and preserve memory coherency between different processes and domains.

Consider the following description of a typical use-case: Domain X enters a section of code where shared data values are to be updated. The domain must first acquire a semaphore. Think of this as the locking (or closing) of a software gate. After the gate locks, a properly-architected software system does not allow other processes (or domains) to execute the same code segment or modify the shared data structure protected by the gate. In other words, the system locks out other processes/domains. Many software implementations include a spin-wait loop within the lock function until the gate locks. After domain X obtains the lock, domain X continues execution and updates the data values protected by the particular lock. After domain X completes the updates, it unlocks (or opens) the software gate, allowing other processes/domains access to the updated data values.

A correctly-implemented system solution must follow these important rules:

- A gate variable must protect all writes to shared data values or shared hardware resources.

- After a domain locks a gate, the system must block other processes/domains from accessing the shared data or resources. Software conventions enforce this.
- The domain that locks a particular gate is the only domain that can open (unlock) that gate.

Information in the hardware gate identifying the locking domain can be extremely useful for system-level debugging.

The Hennessy/Patterson text on computer architecture offers the following description for software gating:

"One of the major requirements of a shared-memory architecture multiprocessor is being able to coordinate processes that are working on a common task. Typically, a programmer will use *lock variables* to synchronize the processes.

The difficulty for the architect of a multiprocessor is to provide a mechanism to decide which processor gets the lock and to provide the operation that locks a variable. Arbitration is easy for shared-bus multiprocessors, since the bus is the only path to memory. The processor that gets the bus locks out all the other processors from memory. If the CPU and bus provide an atomic swap operation, programmers can create locks with the proper semantics. The adjective "atomic" is key, for it means that a processor can both read a location and set it to the locked value in the same bus operation, preventing any other processor from reading or writing memory." [Hennessy/Patterson, *Computer Architecture: A Quantitative Approach*]

The classic text continues with a description of the steps required to lock/unlock a variable using an atomic swap instruction:

"Assume that 0 means unlocked and 1 means locked. A processor first reads the lock variable to test its state. A processor keeps reading and testing until the value indicates that the lock is unlocked. The processor then races against all other processes that were similarly "spin waiting" to see who can lock the variable first. All processes use a swap instruction that reads the old value and stores a 1 into the lock variable. The single winner will see the 0, and the losers will see a 1 that was placed there by the winner. (The losers will continue to set the variable to the locked value, but this does not matter.) The winning processor executes the code after the lock and then stores a 0 into the lock when it exits, starting the race all over again. Testing the old value and then setting to a new value is why the atomic swap instruction is called *test and set* in some instruction sets." [Hennessy/Patterson, *Computer Architecture: A Quantitative Approach*]

10.2.2 Features

SEMA42 implements hardware-enforced semaphores as an IPS-mapped slave peripheral device. The feature set includes:

- Support for 16 hardware-enforced gates in a multi-domain configuration that supports up to 15 domains. In this chapter, dm*X* represents domain *X*.
  - Gates appear as a 16-entry byte-size array with read and write accesses.
    - Domains lock gates by writing "domainID_number+1" to the appropriate gate and must read back the gate value to verify that the lock operation succeeded.
    - After the gate locks, the locking domain unlocks the gate by writing zeroes.
  - Each hardware gate appears as a 16-state, 4-bit state machine.
    - 16-state implementation

      If gate = 0h, then state = unlocked
      if gate = 1h, then state = locked by domain (master) 0
      if gate = 2h, then state = locked by domain (master) 1
      …
      if gate = Fh, then state = locked by domain (master) 14
    - SEMA42 uses the logical domain number and the specified data patterns as reference attributes to validate all write operation.
    - After a gate locks, the locking domain must unlock the gate by writing zeroes.
  - Support for secure reset mechanisms to clear the contents of individual gates, as well as a clear-all capability
- Memory-mapped slave peripheral that offers an interface to the IPS bus for programming-model accesses

The following figure shows a simplified SEMA42 block diagram. In the diagram, the register blocks named gate0, gate1,…, gate (n-2) and gate (n-1) include the finite state machines implementing the semaphore gates.

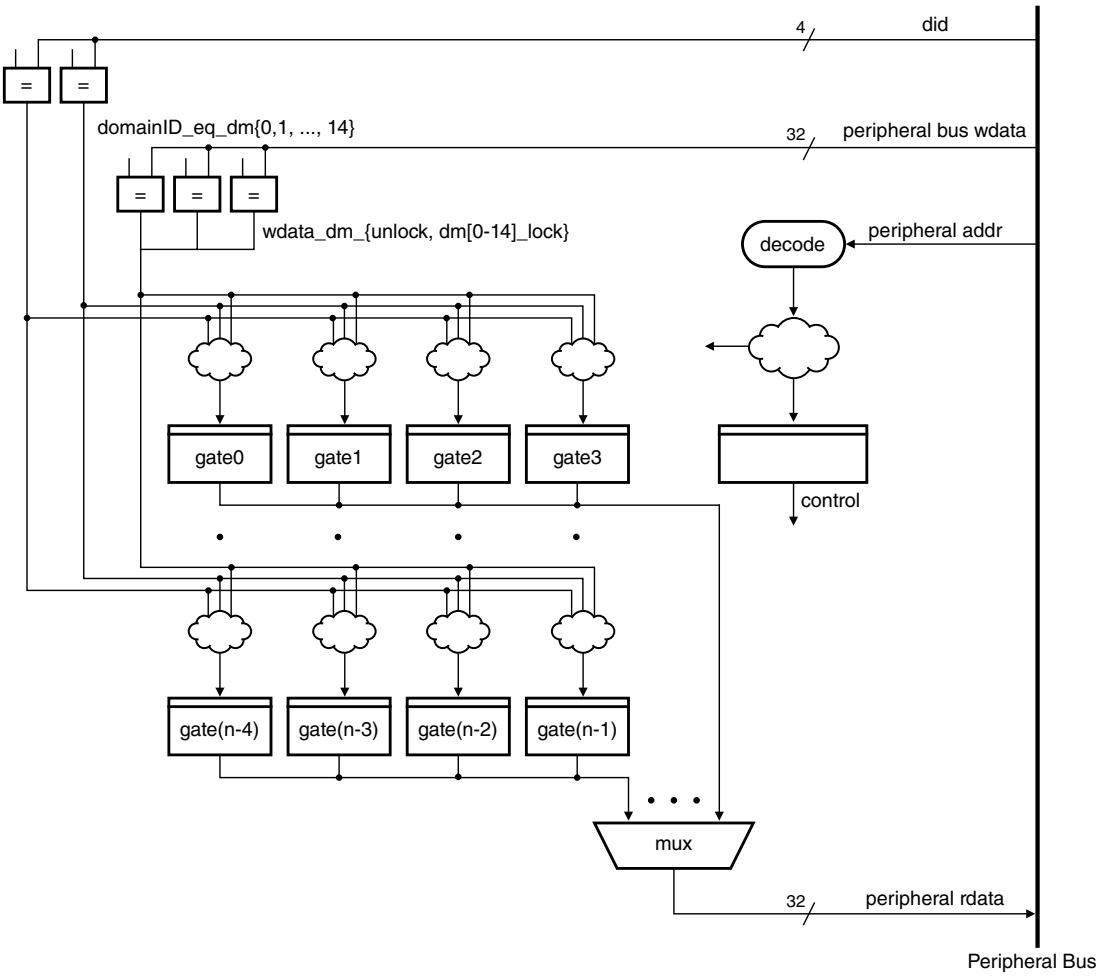


Figure 10-1. SEMA42 block diagram

10.3 Memory map/register definition

You can access these registers only in Supervisor mode. User accesses terminate with an error.

10.3.1 SEMA42 register descriptions

10.3.1.1 SEMA42 Memory map

SEMA42_0 base address: 4101_B000h

SEMA42_1 base address: 401B_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Gate Register (GATE3)	8	RW	00h
1h	Gate Register (GATE2)	8	RW	00h
2h	Gate Register (GATE1)	8	RW	00h
3h	Gate Register (GATE0)	8	RW	00h
4h	Gate Register (GATE7)	8	RW	00h
5h	Gate Register (GATE6)	8	RW	00h
6h	Gate Register (GATE5)	8	RW	00h
7h	Gate Register (GATE4)	8	RW	00h
8h	Gate Register (GATE11)	8	RW	00h
9h	Gate Register (GATE10)	8	RW	00h
Ah	Gate Register (GATE9)	8	RW	00h
Bh	Gate Register (GATE8)	8	RW	00h
Ch	Gate Register (GATE15)	8	RW	00h
Dh	Gate Register (GATE14)	8	RW	00h
Eh	Gate Register (GATE13)	8	RW	00h
Fh	Gate Register (GATE12)	8	RW	00h
42h	Reset Gate Read (RSTGT_R)	16	RO	0000h
42h	Reset Gate Write (RSTGT_W)	16	WO	See description.

10.3.1.2 Gate Register (GATE0 - GATE15)

10.3.1.2.1 Offset

Register	Offset
GATE3	0h
GATE2	1h
GATE1	2h
GATE0	3h
GATE7	4h
GATE6	5h
GATE5	6h
GATE4	7h
GATE11	8h
GATE10	9h
GATE9	Ah
GATE8	Bh

Table continues on the next page...

Register	Offset
GATE15	Ch
GATE14	Dh
GATE13	Eh
GATE12	Fh

10.3.1.2.2 Function

SEMA42 implements each semaphore gate in a 4-bit finite state machine, right-justified in a byte data structure. The hardware uses the logical domain-identifier number in conjunction with the data patterns to validate all attempted write operations. Only domain masters can modify the gate registers. After a gate locks, only the locking domain must open (unlock) the gate.

You can read multiple gate values in a single access. However, you can update only a single gate at a time, via a write operation. If you attempt to write a data value that is neither the unlock value (00h) nor the appropriate lock value (domainID_number + 1), SEMA42 considers this as "no operation" and does not change any gate state. Attempts to write multiple gates in a single-aligned access with a size larger than 8 bits (byte) generate an error termination and do not allow any gate state changes.

10.3.1.2.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0				GTFSM			
W								
Reset	0	0	0	0	0	0	0	0

10.3.1.2.4 Fields

Field	Function
7-4 —	Reserved
3-0 GTFSM	Gate finite state machine The state of the gate reflects the last domain that locked it. This can be useful during system debug. The hardware gate has a 16-state implementation, defined as: 0000b - The gate is unlocked (free). 0001b - Domain 0 locked the gate. 0010b - Domain 1 locked the gate. 0011b - Domain 2 locked the gate.

Field	Function
	0100b - Domain 3 locked the gate. 0101b - Domain 4 locked the gate. 0110b - Domain 5 locked the gate. 0111b - Domain 6 locked the gate. 1000b - Domain 7 locked the gate. 1001b - Domain 8 locked the gate. 1010b - Domain 9 locked the gate. 1011b - Domain 10 locked the gate. 1100b - Domain 11 locked the gate. 1101b - Domain 12 locked the gate. 1110b - Domain 13 locked the gate. 1111b - Domain 14 locked the gate.

10.3.1.3 Reset Gate Read (RSTGT_R)

10.3.1.3.1 Offset

Register	Offset
RSTGT_R	42h

10.3.1.3.2 Function

This section and Reset Gate Write (RSTGT_W) describe the same register. This section describes the general operation of the register and also shows how the register fields appear when you read the register. Reset Gate Write (RSTGT_W) shows how the register fields appear when you write to the register.

The intent of the hardware gate implementation is to specify a protocol where the locking domain must unlock the gate. However, some systems may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset. To support this special gate reset requirement, SEMA42 implements a secure reset mechanism that allows you to initialize a hardware gate (or all the gates) by following a specific dual-write access pattern. The secure-gate reset:

- Uses a technique similar to that required for the servicing of a software watchdog timer
- Requires two consecutive writes with pre-defined data patterns from the same domain

You must do this to force the clearing of the specified gate(s). The required access pattern as follows:

1. A domain performs a 16-bit write to the RSTGT memory location. The most significant byte (RSTGT_W[RSTGDP]) must be E2h. The value of least significant byte is irrelevant for this reference and can be anything.
2. The same domain then performs a second 16-bit write to the RSTGT location. For this write, the upper byte (RSTGT_W[RSTGDP]) is the logical complement of the first data pattern (1Dh) and the lower byte (RSTGT_W[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate or all gates to be cleared. If the same domain writes incorrect data on the second access or another domain performs the second write access, SEMA42 aborts the special gate reset sequence and does not assert an error signal.
3. Reads of the RSTGT location return information on the 2-bit state machine (RSTGT_R[RSTGSM]) that implements these functions:
   - The domain performing the reset (RSTGT_R[RSTGMS])
   - The last-cleared gate number(s) (RSTGT_R[RSTGTN])

   Reads of the RSTGT register do not affect the secure-reset finite state machine in any manner.

10.3.1.3.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		RSTGSM		RSTGMS				RSTGTN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10.3.1.3.4 Fields

Field	Function
15-14 ROZ	ROZ This field always returns the value 0 when you read it.
13-12 RSTGSM	Reset gate finite state machine Reads of the RSTGT register return the encoded state machine value. RSTGSM = 10b is valid for only a single machine cycle, so a read can never return this value. SEMA42 maintains the reset state machine in a 2-bit, 3-state implementation, defined as follows: 00b - Idle, waiting for the first data pattern write. 01b - Waiting for the second data pattern write 10b - The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. 11b - This state encoding is never used and therefore reserved.
11-8 RSTGMS	Reset gate domain

Table continues on the next page...

Field	Function
	This field records the logical number of the domain performing the gate reset function. The logical number is the domain number. This domain number is determined by the XRDC's Master Domain Assignment Controller (XRDC_MDAC). To succeed, this function requires that the same domain initiate the two consecutive writes to this register. SEMA42 updates the field each time a write to this register occurs.
7-0 RSTGTN	Reset gate number This field specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

10.3.1.4 Reset Gate Write (RSTGT_W)

10.3.1.4.1 Offset

Register	Offset
RSTGT_W	42h

10.3.1.4.2 Function

This section describes how the RSTGT fields appear when the register is written. See Reset Gate Read (RSTGT_R) for the description of the register's overall operation and how the register fields appear when you read the register.

10.3.1.4.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	RSTGDP								RSTGTN							
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

1

1. Reset value is not applicable for writes.

10.3.1.4.4 Fields

Field	Function
15-8 RSTGDP	Reset gate data pattern

Table continues on the next page...

Field	Function
	You access this field with the specified data patterns on the two consecutive writes to enable the gate-reset mechanism. For the first write, RSTGDP must be E2h. For the second write, RSTGDP must be 1Dh.
7-0 RSTGTN	Reset gate number This field specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

10.4 Functional description

This section describes more about the SEMA42 functional operation and specific details of the state machines of the GATE*n* registers.

As described previously, each of the GATE*n* registers implements a 4-bit, 16-state machine. The following figure shows a simplified diagram of the state transitions for each gate.

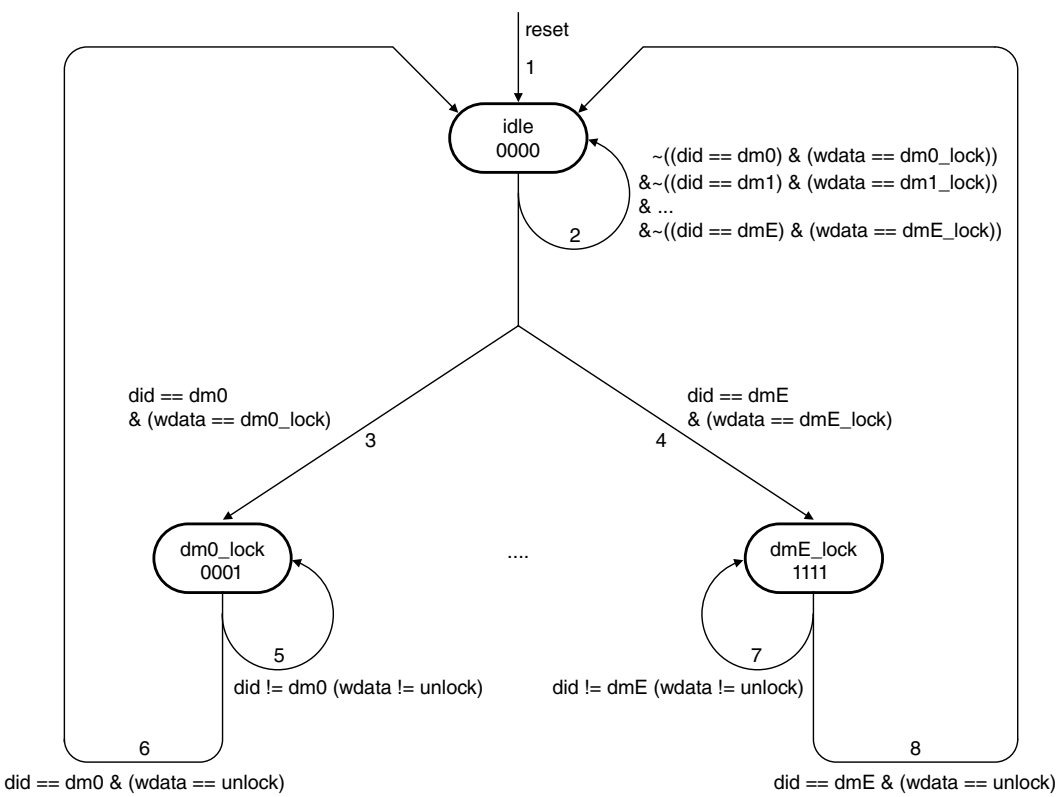


Figure 10-2. GATE*n* state machine

In the figure above, "did" is the domain identifier and "dm" is the domain number. The domain number identifies core domain *X* (dm*X*). Thus, for example, "dmE" represents domain 14 (E in hexadecimal). The platform passes the domain number to SEMA42.

The following table defines the GATE*n* state transitions.

Table 10-3. GATE*n* state transitions

Current state	Next state	Transition	Description
–	idle	1	Any reset, whether a system reset or a software-initiated gate reset, unconditionally forces the gate into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding domain occurs, the gate remains in the idle state.
idle	dm0_lock	3	When domain 0h initiates a write of the dm0_lock data value, the gate transitions into the dm0_lock state.
idle	dmE_lock	4	When domain Eh initiates a write of the dmE_lock value, the gate transitions into the dmE_lock state.
dm0_lock	dm0_lock	5	When in this state, the gate remains here if any attempted write is not from domain 0h with the unlock data value.
dm0_lock	idle	6	The gate returns to the idle (unlocked) state after a write from domain 0h with the unlock data value occurs.
dmE_lock	dmE_lock	7	When in this state, the gate remains here if any attempted write is not from domain Eh with the unlock data value.
dmE_lock	idle	8	The gate returns to the idle (unlocked) state after a write from domain Eh with the unlock data value occurs.

SEMA42 uses these gate data values:

- The lock data value is (domain number) + 1.
- The unlock data value is 00h.

Chapter 11
Extended Resource Domain Control (XRDC)

11.1 Chip-specific XRDC information

Table 11-1. Reference links to related information

Topic	Related module	Reference
Full description	XRDC	XRDC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

11.1.1 Extended Resource Domain Control (XRDC)

The Extended Resource Domain Controller (XRDC) provides an integrated, scalable architectural framework for access control, system memory protection and peripheral isolation. It allows software to assign chip resources (like processor cores, non-core bus masters, memory regions and slave peripherals) to processing domains, to support enforcement of robust operational environments.

The XRDC implementation is distributed across multiple submodules instantiated throughout the device. The XRDC submodules include:

- **XRDC_MGR**: The Manager (MGR) submodule coordinates all programming model reads and writes.
- **XRDC_MDAC**: The Master Domain Assignment Controller (MDAC) handles resource assignments and generation of the domain identifiers.

- **XRDC_MRC**: The Memory Region Controller (MRC) implements the access controls for slave memories based on the pre-programmed region descriptor registers.
- **XRDC_PAC**: The Peripheral Access Controller (PAC) implements the access controls for slave peripherals based on the pre-programmed domain access control registers.

Table 11-2. XRDC configuration

Parameter	Description
Name	Extended Resource Domain Control
Instances	1
Configurable features	Refer to Multicore Architecture for details
Interface speed	NA
External I/O pins	NA

11.1.2 XRDC configuration for MDAC, MRC, and PAC registers

Table 11-3. MDAC configuration

XRDC MDA	MDAC_REG_NUM	DEFAULT_DID	MDAC_CPU	MDAC_INST_NUM
CM4CODE	2	0	1	6'd00
CM4SYS	2	0	1	6'd01
CM4DMA	1	0	0	6'd02
CA7	2	1	1	6'd03
LCDIF	1	1	0	6'd04
GPU3D	1	1	0	6'd05
CA7DMA	1	1	0	6'd06
AXBS2NIC1	1	1	0	6'd07
CAAM[1]	1	1	0	6'd08
USB0/1	1	1	0	6'd09
VIU	1	1	0	6'd10
SDHC0	1	1	0	6'd11
SDHC1	1	1	0	6'd12
GPU2D	1	1	0	6'd13

1. See the i.MX 7ULP Security Reference Manual for this module.

Table 11-4. MRC configuration

MRC#	SLAVE_NUM	Slave Memory	REGION_NUM	SEMA4_NUM	SEMA4_INST
MRC0	1	M4 TCMs	4	16	SEMA4_0

Table continues on the next page...

Table 11-4. MRC configuration (continued)

MRC#	SLAVE_NUM	Slave Memory	REGION_NUM	SEMA4_NUM	SEMA4_INST
MRC1	1	QSPI	8	16	SEMA4_0
MRC2	2	SRAM0	4	16	SEMA4_1
MRC3	2	SecRAM	4	16	SEMA4_1
MRC4	1	FlexBus	4	16	SEMA4_1
MRC5	2	SRAM1	4	16	SEMA4_1
MRC6	2	MMDC	8	16	SEMA4_1

Table 11-5. PAC configuration

PAC #	IP Bus	SEMA4_NUM	SEMA4_INST
PAC0	CM4 AIPS0	16	SEMA4_0
PAC1	CM4 AIPS1	16	SEMA4_0
PAC2	CA7 AHB_PBRIDGE0	16	SEMA4_1
PAC3	CA7 AHB_PBRIDGE1	16	SEMA4_1

11.1.3 XRDC domain identifier (DID) value

In I.MX 7ULP, the default DID values for Master Domain Assignment registers are defined in the following table.

Table 11-6. Default DID values for MDACFG0-13 registers

MDAn	Default DID value
MDA0-2	0
MDA3-13	1

11.1.4 MRC port number to detect access violation

The DERR_W1[EPORT] identifies the encoded port number of the MRC that detected the access violation. For this device, the MRC port number 0 detects access violation.

11.2 Introduction

The Extended Resource Domain Controller (XRDC) provides an integrated, scalable architectural framework for access control, system memory protection and peripheral isolation. It allows software to assign chip resources including processor cores, non-core bus masters, memory regions and slave peripherals to processing *domains* to support enforcement of robust operational environments. First, each bus mastering resource is assigned to a domain identifier (domainID, DID). For processors, there are additional fields that can optionally be used to assign it to multiple domains. Next, the access control policies for the individual domains are programmed into any number of slave memory region descriptors and slave peripheral domain access control registers. Finally, all accesses throughout the device are monitored concurrently to determine the validity of each and every access. If a reference from a given domain has sufficient access rights, it is allowed to continue, else the access is aborted and error information captured.

The access control scheme defined by the XRDC supports a 4-level model, combining the traditional privileged (also known as supervisor) and user modes with an additional signal defining the secure, nonsecure attributes of each memory reference. The result is a 4-level hierarchical access control mechanism, where:

SecurePriv(ileged) > SecureUser > NonsecurePriv(ileged) > NonsecureUser

with different access control policies based on read and write references. Combined with the user/privileged and secure/nonsecure attributes, a domainID is associated with every system bus transaction and forms the hardware basis for implementation of the XRDC's access control mechanisms.

Access to shared memory regions and slave peripherals can be dynamically controlled with the optional inclusion of a hardware semaphore. If a hardware semaphore is enabled for a given address space or peripheral, then writes to the targeted address space are only allowed if the requesting domain owns the semaphore. This capability allows the access control policy for a given resource to be dynamically revised based on hardware semaphore ownership.

Each core contains its own local memory protection unit to control permissions and manage resources among the various tasks executing on that core. The XRDC adds a second layer of protection for the shared resources in a multicore system by providing an integrated, scalable architectural framework for access control, system memory protection, and peripheral isolation. It allows software to assign chip resources including processor cores, non-core bus masters, memory regions, and slave peripherals to processing domains to support enforcement of robust operational environments.

11.2.1 Features

The key features of the XRDC include:

- Assignment of chip resources to processing "domains"
  - Resources are categorized into four groups:
    - Processor cores
    - non-core bus masters
    - slave memories
    - slave peripherals
  - Each processing domain is assigned a unique domain identifier (domainID, DID)
  - DomainID is a new attribute associated with every system bus transaction
  - Used in conjunction with user/privileged, secure/nonsecure attributes
- Access rights to slave targets defined in region descriptor registers for memories and access control registers for peripherals
- Supports sharing of memory and peripherals with optional inclusion of hardware semaphores to dynamically determine access rights
- Built upon a 4-level hierarchical access control model
  - SecurePriv(ileged) > SecureUser > NonsecurePriv(ileged) > NonsecureUser
  - Encoded into a 3-bit per-domain access control policy (ACP) used throughout the XRDC
  - Certain processors do not support the NonsecurePriv state. For these cores, the model simplifies to a 3-state definition: SecurePriv > SecureUser > NonsecureUser
- Programming model and hardware implementation is distributed across multiple submodules
  - Supports a broad, highly-configurable architecture definition
  - Memory region descriptors support a format leveraged from the Cortex-M Core Memory Protection Unit

11.2.2 Block diagram

As previously noted, the XRDC implementation is distributed across multiple submodules instantiated throughout the device. The XRDC submodules include:

- XRDC_MGR
  - The Manager (MGR) submodule coordinates all programming model reads and writes
- XRDC_MDAC

    - The Master Domain Assignment Controller (MDAC) handles resource assignments and generation of the domain identifiers
- XRDC_MRC
    - The Memory Region Controller (MRC) implements the access controls for slave memories based on the pre-programmed region descriptor registers
- XRDC_PAC
    - The Peripheral Access Controller (PAC) implements the access controls for slave peripherals based on the pre-programmed domain access control registers

For chip-specific implementation details of this module's instances, see the chip configuration information.

See Figure 11-1 for a simplified XRDC block diagram focusing on topology and connections.

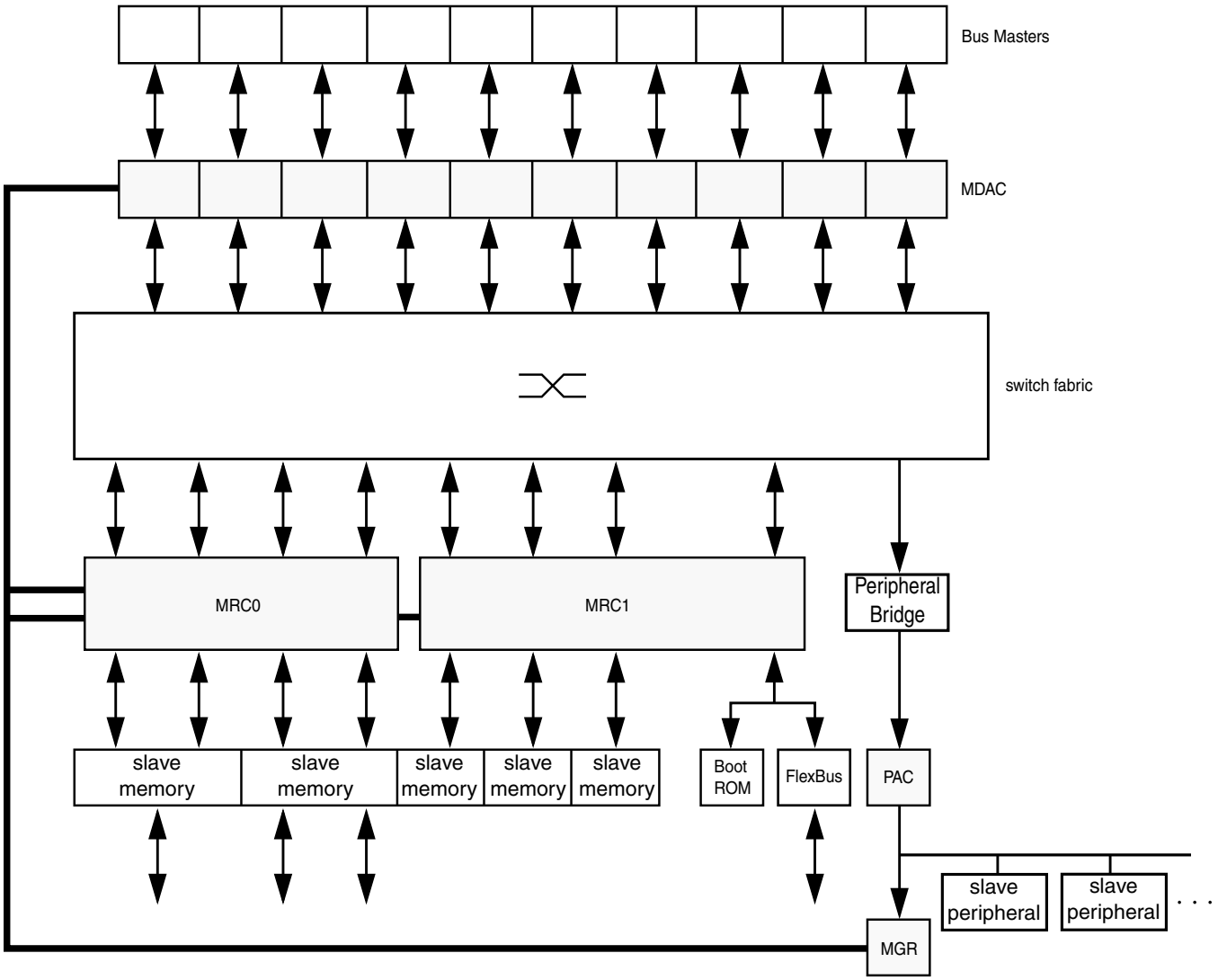


Figure 11-1. Simplified XRDC Block Diagram

11.2.3 Modes of operation

The XRDC module does not support any special modes of operation. As a memory-mapped device located in the core platform's clock domain, it responds based strictly on the memory addresses of the connected system buses. The slave peripheral bus is used to access the XRDC's distributed programming model; these references are only allowed if the requesting bus master is operating in the secure, privileged mode. Domain assignment and access control functions are evaluated on a reference-by-reference basis using the addresses and attributes connected to the system bus port(s). The implementation of the XRDC_MRC submodule supports either AMBA-AHB or AMBA-AXI system bus protocols.

11.3 External signal description

The XRDC module does not directly support any external interfaces.

The *internal* interfaces include a standard 32-bit slave bus for all programming model accesses, connections to the address phase signals associated with AHB and/or AXI system buses and connections to the slave peripheral buses as shown in Figure 11-1.

11.4 Memory map/register definition

The XRDC module supports a large number of architecturally-defined, program-visible registers distributed across four 4 Kbyte address slots. The actual registers present in any given device are chip-specific.

The register resources of the XRDC programming model can only be accessed while in secure, privileged mode. Unless noted otherwise, the programming model registers can be accessed via 8-, 16- or 32-bit reads and 32-bit write references. Attempted accesses in a different operating mode, using unsupported write data sizes, writes to read-only resources, or to reserved spaces are terminated with an error unless noted otherwise.

Accesses to these memory map holes will return an error:

- Any access to an MDAC, PDAC, or MRC that is not there.
- The holes in the 0x0-0xF0 and DERR to PID register space.
- For MDAC, gaps in the registers.
- For MRCs, RGD gaps. So if there are 4 RGDs, access to RGD5 will fail.

Accesses to these memory map holes do not return a bus error:

- For MRCs, current RGDs are 4 words, but have 8 words of address available. Words 4-7.
- For PDAC, reserved slots.

The XRDC programming model is partitioned into 5 groups of registers:

- Basic hardware control and configuration
- Domain errors: location and details
- Master domain assignments
- Peripheral domain access controls
- Memory region descriptors

It should be noted that many of the programming model registers in the XRDC are organized as 2-dimensional data structures. These generic arrays contain "m" words representing the "columns" and "n" instances of the structure representing the "rows". These may be described as `structure[n][m]`. They appear in the address space of the programming model memory map in the standard C language row-major layout, that is, the "m" words representing the "column" appear sequentially with the entire row replicated "n" times. The XRDC register structure uses the naming convention of *XRDC_<regname>_Wm_n*, where the column number "m" appears as a numeric suffix on the W (32-bit word) identifier, and the row identifier "n" appears as the final numerical suffix.

The architectural definition of the complete XRDC programming model is presented in this section. Each implementation is SoC-specific and highly dependent on the device requirements.

11.4.1 XRDC register descriptions

11.4.1.1 XRDC Memory map

XRDC base address: 4101_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_000Ah
F0h	Hardware Configuration 0 (HWCFG0)	32	RO	1306_0D07h
F4h	Hardware Configuration 1 (HWCFG1)	32	RO	See description.
F8h	Hardware Configuration 2 (HWCFG2)	32	RO	0000_0000h
100h	Master Domain Assignment Configuration (MDACFG0)	8	RO	02h
101h	Master Domain Assignment Configuration (MDACFG1)	8	RO	02h
102h	Master Domain Assignment Configuration (MDACFG2)	8	RO	81h
103h	Master Domain Assignment Configuration (MDACFG3)	8	RO	02h
104h	Master Domain Assignment Configuration (MDACFG4)	8	RO	81h
105h	Master Domain Assignment Configuration (MDACFG5)	8	RO	81h
106h	Master Domain Assignment Configuration (MDACFG6)	8	RO	81h
107h	Master Domain Assignment Configuration (MDACFG7)	8	RO	81h
108h	Master Domain Assignment Configuration (MDACFG8)	8	RO	81h
109h	Master Domain Assignment Configuration (MDACFG9)	8	RO	81h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
10Ah	Master Domain Assignment Configuration (MDACFG10)	8	RO	81h
10Bh	Master Domain Assignment Configuration (MDACFG11)	8	RO	81h
10Ch	Master Domain Assignment Configuration (MDACFG12)	8	RO	81h
10Dh	Master Domain Assignment Configuration (MDACFG13)	8	RO	81h
140h	Memory Region Configuration (MRCFG0)	8	RO	04h
141h	Memory Region Configuration (MRCFG1)	8	RO	08h
142h	Memory Region Configuration (MRCFG2)	8	RO	04h
143h	Memory Region Configuration (MRCFG3)	8	RO	04h
144h	Memory Region Configuration (MRCFG4)	8	RO	04h
145h	Memory Region Configuration (MRCFG5)	8	RO	04h
146h	Memory Region Configuration (MRCFG6)	8	RO	08h
200h - 21Ch	Domain Error Location (DERRLOC0 - DERRLOC7)	32	RO	0000_0000h
400h	Domain Error Word0 (DERR_W0_0)	32	RO	0000_0000h
404h	Domain Error Word1 (DERR_W1_0)	32	RO	0000_0000h
40Ch	Domain Error Word3 (DERR_W3_0)	32	WORZ	0000_0000h
410h	Domain Error Word0 (DERR_W0_1)	32	RO	0000_0000h
414h	Domain Error Word1 (DERR_W1_1)	32	RO	0000_0000h
41Ch	Domain Error Word3 (DERR_W3_1)	32	WORZ	0000_0000h
420h	Domain Error Word0 (DERR_W0_2)	32	RO	0000_0000h
424h	Domain Error Word1 (DERR_W1_2)	32	RO	0000_0000h
42Ch	Domain Error Word3 (DERR_W3_2)	32	WORZ	0000_0000h
430h	Domain Error Word0 (DERR_W0_3)	32	RO	0000_0000h
434h	Domain Error Word1 (DERR_W1_3)	32	RO	0000_0000h
43Ch	Domain Error Word3 (DERR_W3_3)	32	WORZ	0000_0000h
440h	Domain Error Word0 (DERR_W0_4)	32	RO	0000_0000h
444h	Domain Error Word1 (DERR_W1_4)	32	RO	0000_0000h
44Ch	Domain Error Word3 (DERR_W3_4)	32	WORZ	0000_0000h
450h	Domain Error Word0 (DERR_W0_5)	32	RO	0000_0000h
454h	Domain Error Word1 (DERR_W1_5)	32	RO	0000_0000h
45Ch	Domain Error Word3 (DERR_W3_5)	32	WORZ	0000_0000h
460h	Domain Error Word0 (DERR_W0_6)	32	RO	0000_0000h
464h	Domain Error Word1 (DERR_W1_6)	32	RO	0000_0000h
46Ch	Domain Error Word3 (DERR_W3_6)	32	WORZ	0000_0000h
500h	Domain Error Word0 (DERR_W0_16)	32	RO	0000_0000h
504h	Domain Error Word1 (DERR_W1_16)	32	RO	0000_0000h
50Ch	Domain Error Word3 (DERR_W3_16)	32	WORZ	0000_0000h
510h	Domain Error Word0 (DERR_W0_17)	32	RO	0000_0000h
514h	Domain Error Word1 (DERR_W1_17)	32	RO	0000_0000h
51Ch	Domain Error Word3 (DERR_W3_17)	32	WORZ	0000_0000h
520h	Domain Error Word0 (DERR_W0_18)	32	RO	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
524h	Domain Error Word1 (DERR_W1_18)	32	RO	0000_0000h
52Ch	Domain Error Word3 (DERR_W3_18)	32	WORZ	0000_0000h
530h	Domain Error Word0 (DERR_W0_19)	32	RO	0000_0000h
534h	Domain Error Word1 (DERR_W1_19)	32	RO	0000_0000h
53Ch	Domain Error Word3 (DERR_W3_19)	32	WORZ	0000_0000h
700h	Process Identifier (PID0)	32	RW	0000_0000h
704h	Process Identifier (PID1)	32	RW	0000_0000h
70Ch	Process Identifier (PID3)	32	RW	0000_0000h
800h	Master Domain Assignment (MDA_W0_0_DFMT0)	32	RW	0000_0000h
804h	Master Domain Assignment (MDA_W1_0_DFMT0)	32	RW	0000_0000h
820h	Master Domain Assignment (MDA_W0_1_DFMT0)	32	RW	0000_0000h
824h	Master Domain Assignment (MDA_W1_1_DFMT0)	32	RW	0000_0000h
840h	Master Domain Assignment (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	Master Domain Assignment (MDA_W0_3_DFMT0)	32	RW	0000_0000h
864h	Master Domain Assignment (MDA_W1_3_DFMT0)	32	RW	0000_0000h
880h	Master Domain Assignment (MDA_W0_4_DFMT1)	32	RW	2000_0000h
8A0h	Master Domain Assignment (MDA_W0_5_DFMT1)	32	RW	2000_0000h
8C0h	Master Domain Assignment (MDA_W0_6_DFMT1)	32	RW	2000_0000h
8E0h	Master Domain Assignment (MDA_W0_7_DFMT1)	32	RW	2000_0000h
900h	Master Domain Assignment (MDA_W0_8_DFMT1)	32	RW	2000_0000h
920h	Master Domain Assignment (MDA_W0_9_DFMT1)	32	RW	2000_0000h
940h	Master Domain Assignment (MDA_W0_10_DFMT1)	32	RW	2000_0000h
960h	Master Domain Assignment (MDA_W0_11_DFMT1)	32	RW	2000_0000h
980h	Master Domain Assignment (MDA_W0_12_DFMT1)	32	RW	2000_0000h
9A0h	Master Domain Assignment (MDA_W0_13_DFMT1)	32	RW	2000_0000h
1040h	Peripheral Domain Access Control (PDAC_W0_8)	32	RW	0000_0000h
1044h	Peripheral Domain Access Control (PDAC_W1_8)	32	RW	0000_0000h
1048h	Peripheral Domain Access Control (PDAC_W0_9)	32	RW	0000_0000h
104Ch	Peripheral Domain Access Control (PDAC_W1_9)	32	RW	0000_0000h
1078h	Peripheral Domain Access Control (PDAC_W0_15)	32	RW	0000_0000h
107Ch	Peripheral Domain Access Control (PDAC_W1_15)	32	RW	0000_0000h
10A0h	Peripheral Domain Access Control (PDAC_W0_20)	32	RW	0000_0000h
10A4h	Peripheral Domain Access Control (PDAC_W1_20)	32	RW	0000_0000h
10A8h	Peripheral Domain Access Control (PDAC_W0_21)	32	RW	0000_0000h
10ACh	Peripheral Domain Access Control (PDAC_W1_21)	32	RW	0000_0000h
10B0h	Peripheral Domain Access Control (PDAC_W0_22)	32	RW	0000_0000h
10B4h	Peripheral Domain Access Control (PDAC_W1_22)	32	RW	0000_0000h
10B8h	Peripheral Domain Access Control (PDAC_W0_23)	32	RW	0000_0000h
10BCh	Peripheral Domain Access Control (PDAC_W1_23)	32	RW	0000_0000h
10D8h	Peripheral Domain Access Control (PDAC_W0_27)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
10DCh	Peripheral Domain Access Control (PDAC_W1_27)	32	RW	0000_0000h
1100h	Peripheral Domain Access Control (PDAC_W0_32)	32	RW	0000_0000h
1104h	Peripheral Domain Access Control (PDAC_W1_32)	32	RW	0000_0000h
1108h	Peripheral Domain Access Control (PDAC_W0_33)	32	RW	0000_0000h
110Ch	Peripheral Domain Access Control (PDAC_W1_33)	32	RW	0000_0000h
1110h	Peripheral Domain Access Control (PDAC_W0_34)	32	RW	0000_0000h
1114h	Peripheral Domain Access Control (PDAC_W1_34)	32	RW	0000_0000h
1120h	Peripheral Domain Access Control (PDAC_W0_36)	32	RW	0000_0000h
1124h	Peripheral Domain Access Control (PDAC_W1_36)	32	RW	0000_0000h
1128h	Peripheral Domain Access Control (PDAC_W0_37)	32	RW	0000_0000h
112Ch	Peripheral Domain Access Control (PDAC_W1_37)	32	RW	0000_0000h
1130h	Peripheral Domain Access Control (PDAC_W0_38)	32	RW	0000_0000h
1134h	Peripheral Domain Access Control (PDAC_W1_38)	32	RW	0000_0000h
1138h	Peripheral Domain Access Control (PDAC_W0_39)	32	RW	0000_0000h
113Ch	Peripheral Domain Access Control (PDAC_W1_39)	32	RW	0000_0000h
1148h	Peripheral Domain Access Control (PDAC_W0_41)	32	RW	0000_0000h
114Ch	Peripheral Domain Access Control (PDAC_W1_41)	32	RW	0000_0000h
1150h	Peripheral Domain Access Control (PDAC_W0_42)	32	RW	0000_0000h
1154h	Peripheral Domain Access Control (PDAC_W1_42)	32	RW	0000_0000h
1160h	Peripheral Domain Access Control (PDAC_W0_44)	32	RW	0000_0000h
1164h	Peripheral Domain Access Control (PDAC_W1_44)	32	RW	0000_0000h
1168h	Peripheral Domain Access Control (PDAC_W0_45)	32	RW	0000_0000h
116Ch	Peripheral Domain Access Control (PDAC_W1_45)	32	RW	0000_0000h
1170h	Peripheral Domain Access Control (PDAC_W0_46)	32	RW	0000_0000h
1174h	Peripheral Domain Access Control (PDAC_W1_46)	32	RW	0000_0000h
1178h	Peripheral Domain Access Control (PDAC_W0_47)	32	RW	0000_0000h
117Ch	Peripheral Domain Access Control (PDAC_W1_47)	32	RW	0000_0000h
1180h	Peripheral Domain Access Control (PDAC_W0_48)	32	RW	0000_0000h
1184h	Peripheral Domain Access Control (PDAC_W1_48)	32	RW	0000_0000h
1188h	Peripheral Domain Access Control (PDAC_W0_49)	32	RW	0000_0000h
118Ch	Peripheral Domain Access Control (PDAC_W1_49)	32	RW	0000_0000h
1190h	Peripheral Domain Access Control (PDAC_W0_50)	32	RW	0000_0000h
1194h	Peripheral Domain Access Control (PDAC_W1_50)	32	RW	0000_0000h
1198h	Peripheral Domain Access Control (PDAC_W0_51)	32	RW	0000_0000h
119Ch	Peripheral Domain Access Control (PDAC_W1_51)	32	RW	0000_0000h
11A0h	Peripheral Domain Access Control (PDAC_W0_52)	32	RW	0000_0000h
11A4h	Peripheral Domain Access Control (PDAC_W1_52)	32	RW	0000_0000h
11A8h	Peripheral Domain Access Control (PDAC_W0_53)	32	RW	0000_0000h
11ACh	Peripheral Domain Access Control (PDAC_W1_53)	32	RW	0000_0000h
11B0h	Peripheral Domain Access Control (PDAC_W0_54)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
11B4h	Peripheral Domain Access Control (PDAC_W1_54)	32	RW	0000_0000h
11B8h	Peripheral Domain Access Control (PDAC_W0_55)	32	RW	0000_0000h
11BCh	Peripheral Domain Access Control (PDAC_W1_55)	32	RW	0000_0000h
11C0h	Peripheral Domain Access Control (PDAC_W0_56)	32	RW	0000_0000h
11C4h	Peripheral Domain Access Control (PDAC_W1_56)	32	RW	0000_0000h
11C8h	Peripheral Domain Access Control (PDAC_W0_57)	32	RW	0000_0000h
11CCh	Peripheral Domain Access Control (PDAC_W1_57)	32	RW	0000_0000h
11D0h	Peripheral Domain Access Control (PDAC_W0_58)	32	RW	0000_0000h
11D4h	Peripheral Domain Access Control (PDAC_W1_58)	32	RW	0000_0000h
11D8h	Peripheral Domain Access Control (PDAC_W0_59)	32	RW	0000_0000h
11DCh	Peripheral Domain Access Control (PDAC_W1_59)	32	RW	0000_0000h
11E8h	Peripheral Domain Access Control (PDAC_W0_61)	32	RW	0000_0000h
11ECh	Peripheral Domain Access Control (PDAC_W1_61)	32	RW	0000_0000h
11F8h	Peripheral Domain Access Control (PDAC_W0_63)	32	RW	0000_0000h
11FCh	Peripheral Domain Access Control (PDAC_W1_63)	32	RW	0000_0000h
1200h	Peripheral Domain Access Control (PDAC_W0_64)	32	RW	0000_0000h
1204h	Peripheral Domain Access Control (PDAC_W1_64)	32	RW	0000_0000h
1208h	Peripheral Domain Access Control (PDAC_W0_65)	32	RW	0000_0000h
120Ch	Peripheral Domain Access Control (PDAC_W1_65)	32	RW	0000_0000h
1210h	Peripheral Domain Access Control (PDAC_W0_66)	32	RW	0000_0000h
1214h	Peripheral Domain Access Control (PDAC_W1_66)	32	RW	0000_0000h
1218h	Peripheral Domain Access Control (PDAC_W0_67)	32	RW	0000_0000h
121Ch	Peripheral Domain Access Control (PDAC_W1_67)	32	RW	0000_0000h
1220h	Peripheral Domain Access Control (PDAC_W0_68)	32	RW	0000_0000h
1224h	Peripheral Domain Access Control (PDAC_W1_68)	32	RW	0000_0000h
1228h	Peripheral Domain Access Control (PDAC_W0_69)	32	RW	0000_0000h
122Ch	Peripheral Domain Access Control (PDAC_W1_69)	32	RW	0000_0000h
1238h	Peripheral Domain Access Control (PDAC_W0_71)	32	RW	0000_0000h
123Ch	Peripheral Domain Access Control (PDAC_W1_71)	32	RW	0000_0000h
1240h	Peripheral Domain Access Control (PDAC_W0_72)	32	RW	0000_0000h
1244h	Peripheral Domain Access Control (PDAC_W1_72)	32	RW	0000_0000h
1248h	Peripheral Domain Access Control (PDAC_W0_73)	32	RW	0000_0000h
124Ch	Peripheral Domain Access Control (PDAC_W1_73)	32	RW	0000_0000h
1250h	Peripheral Domain Access Control (PDAC_W0_74)	32	RW	0000_0000h
1254h	Peripheral Domain Access Control (PDAC_W1_74)	32	RW	0000_0000h
1258h	Peripheral Domain Access Control (PDAC_W0_75)	32	RW	0000_0000h
125Ch	Peripheral Domain Access Control (PDAC_W1_75)	32	RW	0000_0000h
1480h	Peripheral Domain Access Control (PDAC_W0_144)	32	RW	0000_0000h
1484h	Peripheral Domain Access Control (PDAC_W1_144)	32	RW	0000_0000h
1488h	Peripheral Domain Access Control (PDAC_W0_145)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
148Ch	Peripheral Domain Access Control (PDAC_W1_145)	32	RW	0000_0000h
1490h	Peripheral Domain Access Control (PDAC_W0_146)	32	RW	0000_0000h
1494h	Peripheral Domain Access Control (PDAC_W1_146)	32	RW	0000_0000h
1498h	Peripheral Domain Access Control (PDAC_W0_147)	32	RW	0000_0000h
149Ch	Peripheral Domain Access Control (PDAC_W1_147)	32	RW	0000_0000h
14A0h	Peripheral Domain Access Control (PDAC_W0_148)	32	RW	0000_0000h
14A4h	Peripheral Domain Access Control (PDAC_W1_148)	32	RW	0000_0000h
14A8h	Peripheral Domain Access Control (PDAC_W0_149)	32	RW	0000_0000h
14ACh	Peripheral Domain Access Control (PDAC_W1_149)	32	RW	0000_0000h
14B0h	Peripheral Domain Access Control (PDAC_W0_150)	32	RW	0000_0000h
14B4h	Peripheral Domain Access Control (PDAC_W1_150)	32	RW	0000_0000h
14B8h	Peripheral Domain Access Control (PDAC_W0_151)	32	RW	0000_0000h
14BCh	Peripheral Domain Access Control (PDAC_W1_151)	32	RW	0000_0000h
14C0h	Peripheral Domain Access Control (PDAC_W0_152)	32	RW	0000_0000h
14C4h	Peripheral Domain Access Control (PDAC_W1_152)	32	RW	0000_0000h
14D0h	Peripheral Domain Access Control (PDAC_W0_154)	32	RW	0000_0000h
14D4h	Peripheral Domain Access Control (PDAC_W1_154)	32	RW	0000_0000h
14D8h	Peripheral Domain Access Control (PDAC_W0_155)	32	RW	0000_0000h
14DCh	Peripheral Domain Access Control (PDAC_W1_155)	32	RW	0000_0000h
14E0h	Peripheral Domain Access Control (PDAC_W0_156)	32	RW	0000_0000h
14E4h	Peripheral Domain Access Control (PDAC_W1_156)	32	RW	0000_0000h
14E8h	Peripheral Domain Access Control (PDAC_W0_157)	32	RW	0000_0000h
14ECh	Peripheral Domain Access Control (PDAC_W1_157)	32	RW	0000_0000h
14F0h	Peripheral Domain Access Control (PDAC_W0_158)	32	RW	0000_0000h
14F4h	Peripheral Domain Access Control (PDAC_W1_158)	32	RW	0000_0000h
1500h	Peripheral Domain Access Control (PDAC_W0_160)	32	RW	0000_0000h
1504h	Peripheral Domain Access Control (PDAC_W1_160)	32	RW	0000_0000h
1508h	Peripheral Domain Access Control (PDAC_W0_161)	32	RW	0000_0000h
150Ch	Peripheral Domain Access Control (PDAC_W1_161)	32	RW	0000_0000h
1518h	Peripheral Domain Access Control (PDAC_W0_163)	32	RW	0000_0000h
151Ch	Peripheral Domain Access Control (PDAC_W1_163)	32	RW	0000_0000h
1520h	Peripheral Domain Access Control (PDAC_W0_164)	32	RW	0000_0000h
1524h	Peripheral Domain Access Control (PDAC_W1_164)	32	RW	0000_0000h
1528h	Peripheral Domain Access Control (PDAC_W0_165)	32	RW	0000_0000h
152Ch	Peripheral Domain Access Control (PDAC_W1_165)	32	RW	0000_0000h
1530h	Peripheral Domain Access Control (PDAC_W0_166)	32	RW	0000_0000h
1534h	Peripheral Domain Access Control (PDAC_W1_166)	32	RW	0000_0000h
1538h	Peripheral Domain Access Control (PDAC_W0_167)	32	RW	0000_0000h
153Ch	Peripheral Domain Access Control (PDAC_W1_167)	32	RW	0000_0000h
1540h	Peripheral Domain Access Control (PDAC_W0_168)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1544h	Peripheral Domain Access Control (PDAC_W1_168)	32	RW	0000_0000h
1548h	Peripheral Domain Access Control (PDAC_W0_169)	32	RW	0000_0000h
154Ch	Peripheral Domain Access Control (PDAC_W1_169)	32	RW	0000_0000h
1550h	Peripheral Domain Access Control (PDAC_W0_170)	32	RW	0000_0000h
1554h	Peripheral Domain Access Control (PDAC_W1_170)	32	RW	0000_0000h
1558h	Peripheral Domain Access Control (PDAC_W0_171)	32	RW	0000_0000h
155Ch	Peripheral Domain Access Control (PDAC_W1_171)	32	RW	0000_0000h
1560h	Peripheral Domain Access Control (PDAC_W0_172)	32	RW	0000_0000h
1564h	Peripheral Domain Access Control (PDAC_W1_172)	32	RW	0000_0000h
1568h	Peripheral Domain Access Control (PDAC_W0_173)	32	RW	0000_0000h
156Ch	Peripheral Domain Access Control (PDAC_W1_173)	32	RW	0000_0000h
1590h	Peripheral Domain Access Control (PDAC_W0_178)	32	RW	0000_0000h
1594h	Peripheral Domain Access Control (PDAC_W1_178)	32	RW	0000_0000h
1840h	Peripheral Domain Access Control (PDAC_W0_264)	32	RW	0000_0000h
1844h	Peripheral Domain Access Control (PDAC_W1_264)	32	RW	0000_0000h
1848h	Peripheral Domain Access Control (PDAC_W0_265)	32	RW	0000_0000h
184Ch	Peripheral Domain Access Control (PDAC_W1_265)	32	RW	0000_0000h
1878h	Peripheral Domain Access Control (PDAC_W0_271)	32	RW	0000_0000h
187Ch	Peripheral Domain Access Control (PDAC_W1_271)	32	RW	0000_0000h
1880h	Peripheral Domain Access Control (PDAC_W0_272)	32	RW	0000_0000h
1884h	Peripheral Domain Access Control (PDAC_W1_272)	32	RW	0000_0000h
18D8h	Peripheral Domain Access Control (PDAC_W0_283)	32	RW	0000_0000h
18DCh	Peripheral Domain Access Control (PDAC_W1_283)	32	RW	0000_0000h
1908h	Peripheral Domain Access Control (PDAC_W0_289)	32	RW	0000_0000h
190Ch	Peripheral Domain Access Control (PDAC_W1_289)	32	RW	0000_0000h
1910h	Peripheral Domain Access Control (PDAC_W0_290)	32	RW	0000_0000h
1914h	Peripheral Domain Access Control (PDAC_W1_290)	32	RW	0000_0000h
1918h	Peripheral Domain Access Control (PDAC_W0_291)	32	RW	0000_0000h
191Ch	Peripheral Domain Access Control (PDAC_W1_291)	32	RW	0000_0000h
1920h	Peripheral Domain Access Control (PDAC_W0_292)	32	RW	0000_0000h
1924h	Peripheral Domain Access Control (PDAC_W1_292)	32	RW	0000_0000h
1928h	Peripheral Domain Access Control (PDAC_W0_293)	32	RW	0000_0000h
192Ch	Peripheral Domain Access Control (PDAC_W1_293)	32	RW	0000_0000h
1930h	Peripheral Domain Access Control (PDAC_W0_294)	32	RW	0000_0000h
1934h	Peripheral Domain Access Control (PDAC_W1_294)	32	RW	0000_0000h
1938h	Peripheral Domain Access Control (PDAC_W0_295)	32	RW	0000_0000h
193Ch	Peripheral Domain Access Control (PDAC_W1_295)	32	RW	0000_0000h
1948h	Peripheral Domain Access Control (PDAC_W0_297)	32	RW	0000_0000h
194Ch	Peripheral Domain Access Control (PDAC_W1_297)	32	RW	0000_0000h
1950h	Peripheral Domain Access Control (PDAC_W0_298)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1954h	Peripheral Domain Access Control (PDAC_W1_298)	32	RW	0000_0000h
1958h	Peripheral Domain Access Control (PDAC_W0_299)	32	RW	0000_0000h
195Ch	Peripheral Domain Access Control (PDAC_W1_299)	32	RW	0000_0000h
1960h	Peripheral Domain Access Control (PDAC_W0_300)	32	RW	0000_0000h
1964h	Peripheral Domain Access Control (PDAC_W1_300)	32	RW	0000_0000h
1968h	Peripheral Domain Access Control (PDAC_W0_301)	32	RW	0000_0000h
196Ch	Peripheral Domain Access Control (PDAC_W1_301)	32	RW	0000_0000h
1970h	Peripheral Domain Access Control (PDAC_W0_302)	32	RW	0000_0000h
1974h	Peripheral Domain Access Control (PDAC_W1_302)	32	RW	0000_0000h
1988h	Peripheral Domain Access Control (PDAC_W0_305)	32	RW	0000_0000h
198Ch	Peripheral Domain Access Control (PDAC_W1_305)	32	RW	0000_0000h
1998h	Peripheral Domain Access Control (PDAC_W0_307)	32	RW	0000_0000h
199Ch	Peripheral Domain Access Control (PDAC_W1_307)	32	RW	0000_0000h
19A0h	Peripheral Domain Access Control (PDAC_W0_308)	32	RW	0000_0000h
19A4h	Peripheral Domain Access Control (PDAC_W1_308)	32	RW	0000_0000h
19A8h	Peripheral Domain Access Control (PDAC_W0_309)	32	RW	0000_0000h
19ACh	Peripheral Domain Access Control (PDAC_W1_309)	32	RW	0000_0000h
19B0h	Peripheral Domain Access Control (PDAC_W0_310)	32	RW	0000_0000h
19B4h	Peripheral Domain Access Control (PDAC_W1_310)	32	RW	0000_0000h
19B8h	Peripheral Domain Access Control (PDAC_W0_311)	32	RW	0000_0000h
19BCh	Peripheral Domain Access Control (PDAC_W1_311)	32	RW	0000_0000h
19C0h	Peripheral Domain Access Control (PDAC_W0_312)	32	RW	0000_0000h
19C4h	Peripheral Domain Access Control (PDAC_W1_312)	32	RW	0000_0000h
19D0h	Peripheral Domain Access Control (PDAC_W0_314)	32	RW	0000_0000h
19D4h	Peripheral Domain Access Control (PDAC_W1_314)	32	RW	0000_0000h
19E8h	Peripheral Domain Access Control (PDAC_W0_317)	32	RW	0000_0000h
19ECh	Peripheral Domain Access Control (PDAC_W1_317)	32	RW	0000_0000h
19F0h	Peripheral Domain Access Control (PDAC_W0_318)	32	RW	0000_0000h
19F4h	Peripheral Domain Access Control (PDAC_W1_318)	32	RW	0000_0000h
19F8h	Peripheral Domain Access Control (PDAC_W0_319)	32	RW	0000_0000h
19FCh	Peripheral Domain Access Control (PDAC_W1_319)	32	RW	0000_0000h
1A00h	Peripheral Domain Access Control (PDAC_W0_320)	32	RW	0000_0000h
1A04h	Peripheral Domain Access Control (PDAC_W1_320)	32	RW	0000_0000h
1A08h	Peripheral Domain Access Control (PDAC_W0_321)	32	RW	0000_0000h
1A0Ch	Peripheral Domain Access Control (PDAC_W1_321)	32	RW	0000_0000h
1A18h	Peripheral Domain Access Control (PDAC_W0_323)	32	RW	0000_0000h
1A1Ch	Peripheral Domain Access Control (PDAC_W1_323)	32	RW	0000_0000h
1C80h	Peripheral Domain Access Control (PDAC_W0_400)	32	RW	0000_0000h
1C84h	Peripheral Domain Access Control (PDAC_W1_400)	32	RW	0000_0000h
1D08h	Peripheral Domain Access Control (PDAC_W0_417)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1D0Ch	Peripheral Domain Access Control (PDAC_W1_417)	32	RW	0000_0000h
1D10h	Peripheral Domain Access Control (PDAC_W0_418)	32	RW	0000_0000h
1D14h	Peripheral Domain Access Control (PDAC_W1_418)	32	RW	0000_0000h
1D20h	Peripheral Domain Access Control (PDAC_W0_420)	32	RW	0000_0000h
1D24h	Peripheral Domain Access Control (PDAC_W1_420)	32	RW	0000_0000h
1D28h	Peripheral Domain Access Control (PDAC_W0_421)	32	RW	0000_0000h
1D2Ch	Peripheral Domain Access Control (PDAC_W1_421)	32	RW	0000_0000h
1D30h	Peripheral Domain Access Control (PDAC_W0_422)	32	RW	0000_0000h
1D34h	Peripheral Domain Access Control (PDAC_W1_422)	32	RW	0000_0000h
1D38h	Peripheral Domain Access Control (PDAC_W0_423)	32	RW	0000_0000h
1D3Ch	Peripheral Domain Access Control (PDAC_W1_423)	32	RW	0000_0000h
1D40h	Peripheral Domain Access Control (PDAC_W0_424)	32	RW	0000_0000h
1D44h	Peripheral Domain Access Control (PDAC_W1_424)	32	RW	0000_0000h
1D48h	Peripheral Domain Access Control (PDAC_W0_425)	32	RW	0000_0000h
1D4Ch	Peripheral Domain Access Control (PDAC_W1_425)	32	RW	0000_0000h
1D50h	Peripheral Domain Access Control (PDAC_W0_426)	32	RW	0000_0000h
1D54h	Peripheral Domain Access Control (PDAC_W1_426)	32	RW	0000_0000h
1D58h	Peripheral Domain Access Control (PDAC_W0_427)	32	RW	0000_0000h
1D5Ch	Peripheral Domain Access Control (PDAC_W1_427)	32	RW	0000_0000h
1D60h	Peripheral Domain Access Control (PDAC_W0_428)	32	RW	0000_0000h
1D64h	Peripheral Domain Access Control (PDAC_W1_428)	32	RW	0000_0000h
1D68h	Peripheral Domain Access Control (PDAC_W0_429)	32	RW	0000_0000h
1D6Ch	Peripheral Domain Access Control (PDAC_W1_429)	32	RW	0000_0000h
1D70h	Peripheral Domain Access Control (PDAC_W0_430)	32	RW	0000_0000h
1D74h	Peripheral Domain Access Control (PDAC_W1_430)	32	RW	0000_0000h
1D78h	Peripheral Domain Access Control (PDAC_W0_431)	32	RW	0000_0000h
1D7Ch	Peripheral Domain Access Control (PDAC_W1_431)	32	RW	0000_0000h
1D80h	Peripheral Domain Access Control (PDAC_W0_432)	32	RW	0000_0000h
1D84h	Peripheral Domain Access Control (PDAC_W1_432)	32	RW	0000_0000h
1D88h	Peripheral Domain Access Control (PDAC_W0_433)	32	RW	0000_0000h
1D8Ch	Peripheral Domain Access Control (PDAC_W1_433)	32	RW	0000_0000h
1D98h	Peripheral Domain Access Control (PDAC_W0_435)	32	RW	0000_0000h
1D9Ch	Peripheral Domain Access Control (PDAC_W1_435)	32	RW	0000_0000h
1DA8h	Peripheral Domain Access Control (PDAC_W0_437)	32	RW	0000_0000h
1DACh	Peripheral Domain Access Control (PDAC_W1_437)	32	RW	0000_0000h
2000h	Memory Region Descriptor (MRGD_W0_0)	32	RW	0000_0000h
2004h	Memory Region Descriptor (MRGD_W1_0)	32	RW	0000_0000h
2008h	Memory Region Descriptor (MRGD_W2_0)	32	RW	0000_0000h
200Ch	Memory Region Descriptor (MRGD_W3_0)	32	RW	0000_0000h
2020h	Memory Region Descriptor (MRGD_W0_1)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2024h	Memory Region Descriptor (MRGD_W1_1)	32	RW	0000_0000h
2028h	Memory Region Descriptor (MRGD_W2_1)	32	RW	0000_0000h
202Ch	Memory Region Descriptor (MRGD_W3_1)	32	RW	0000_0000h
2040h	Memory Region Descriptor (MRGD_W0_2)	32	RW	0000_0000h
2044h	Memory Region Descriptor (MRGD_W1_2)	32	RW	0000_0000h
2048h	Memory Region Descriptor (MRGD_W2_2)	32	RW	0000_0000h
204Ch	Memory Region Descriptor (MRGD_W3_2)	32	RW	0000_0000h
2060h	Memory Region Descriptor (MRGD_W0_3)	32	RW	0000_0000h
2064h	Memory Region Descriptor (MRGD_W1_3)	32	RW	0000_0000h
2068h	Memory Region Descriptor (MRGD_W2_3)	32	RW	0000_0000h
206Ch	Memory Region Descriptor (MRGD_W3_3)	32	RW	0000_0000h
2200h	Memory Region Descriptor (MRGD_W0_16)	32	RW	0000_0000h
2204h	Memory Region Descriptor (MRGD_W1_16)	32	RW	0000_0000h
2208h	Memory Region Descriptor (MRGD_W2_16)	32	RW	0000_0000h
220Ch	Memory Region Descriptor (MRGD_W3_16)	32	RW	0000_0000h
2220h	Memory Region Descriptor (MRGD_W0_17)	32	RW	0000_0000h
2224h	Memory Region Descriptor (MRGD_W1_17)	32	RW	0000_0000h
2228h	Memory Region Descriptor (MRGD_W2_17)	32	RW	0000_0000h
222Ch	Memory Region Descriptor (MRGD_W3_17)	32	RW	0000_0000h
2240h	Memory Region Descriptor (MRGD_W0_18)	32	RW	0000_0000h
2244h	Memory Region Descriptor (MRGD_W1_18)	32	RW	0000_0000h
2248h	Memory Region Descriptor (MRGD_W2_18)	32	RW	0000_0000h
224Ch	Memory Region Descriptor (MRGD_W3_18)	32	RW	0000_0000h
2260h	Memory Region Descriptor (MRGD_W0_19)	32	RW	0000_0000h
2264h	Memory Region Descriptor (MRGD_W1_19)	32	RW	0000_0000h
2268h	Memory Region Descriptor (MRGD_W2_19)	32	RW	0000_0000h
226Ch	Memory Region Descriptor (MRGD_W3_19)	32	RW	0000_0000h
2280h	Memory Region Descriptor (MRGD_W0_20)	32	RW	0000_0000h
2284h	Memory Region Descriptor (MRGD_W1_20)	32	RW	0000_0000h
2288h	Memory Region Descriptor (MRGD_W2_20)	32	RW	0000_0000h
228Ch	Memory Region Descriptor (MRGD_W3_20)	32	RW	0000_0000h
22A0h	Memory Region Descriptor (MRGD_W0_21)	32	RW	0000_0000h
22A4h	Memory Region Descriptor (MRGD_W1_21)	32	RW	0000_0000h
22A8h	Memory Region Descriptor (MRGD_W2_21)	32	RW	0000_0000h
22ACh	Memory Region Descriptor (MRGD_W3_21)	32	RW	0000_0000h
22C0h	Memory Region Descriptor (MRGD_W0_22)	32	RW	0000_0000h
22C4h	Memory Region Descriptor (MRGD_W1_22)	32	RW	0000_0000h
22C8h	Memory Region Descriptor (MRGD_W2_22)	32	RW	0000_0000h
22CCh	Memory Region Descriptor (MRGD_W3_22)	32	RW	0000_0000h
22E0h	Memory Region Descriptor (MRGD_W0_23)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
22E4h	Memory Region Descriptor (MRGD_W1_23)	32	RW	0000_0000h
22E8h	Memory Region Descriptor (MRGD_W2_23)	32	RW	0000_0000h
22ECh	Memory Region Descriptor (MRGD_W3_23)	32	RW	0000_0000h
2400h	Memory Region Descriptor (MRGD_W0_32)	32	RW	0000_0000h
2404h	Memory Region Descriptor (MRGD_W1_32)	32	RW	0000_0000h
2408h	Memory Region Descriptor (MRGD_W2_32)	32	RW	0000_0000h
240Ch	Memory Region Descriptor (MRGD_W3_32)	32	RW	0000_0000h
2420h	Memory Region Descriptor (MRGD_W0_33)	32	RW	0000_0000h
2424h	Memory Region Descriptor (MRGD_W1_33)	32	RW	0000_0000h
2428h	Memory Region Descriptor (MRGD_W2_33)	32	RW	0000_0000h
242Ch	Memory Region Descriptor (MRGD_W3_33)	32	RW	0000_0000h
2440h	Memory Region Descriptor (MRGD_W0_34)	32	RW	0000_0000h
2444h	Memory Region Descriptor (MRGD_W1_34)	32	RW	0000_0000h
2448h	Memory Region Descriptor (MRGD_W2_34)	32	RW	0000_0000h
244Ch	Memory Region Descriptor (MRGD_W3_34)	32	RW	0000_0000h
2460h	Memory Region Descriptor (MRGD_W0_35)	32	RW	0000_0000h
2464h	Memory Region Descriptor (MRGD_W1_35)	32	RW	0000_0000h
2468h	Memory Region Descriptor (MRGD_W2_35)	32	RW	0000_0000h
246Ch	Memory Region Descriptor (MRGD_W3_35)	32	RW	0000_0000h
2600h	Memory Region Descriptor (MRGD_W0_48)	32	RW	0000_0000h
2604h	Memory Region Descriptor (MRGD_W1_48)	32	RW	0000_0000h
2608h	Memory Region Descriptor (MRGD_W2_48)	32	RW	0000_0000h
260Ch	Memory Region Descriptor (MRGD_W3_48)	32	RW	0000_0000h
2620h	Memory Region Descriptor (MRGD_W0_49)	32	RW	0000_0000h
2624h	Memory Region Descriptor (MRGD_W1_49)	32	RW	0000_0000h
2628h	Memory Region Descriptor (MRGD_W2_49)	32	RW	0000_0000h
262Ch	Memory Region Descriptor (MRGD_W3_49)	32	RW	0000_0000h
2640h	Memory Region Descriptor (MRGD_W0_50)	32	RW	0000_0000h
2644h	Memory Region Descriptor (MRGD_W1_50)	32	RW	0000_0000h
2648h	Memory Region Descriptor (MRGD_W2_50)	32	RW	0000_0000h
264Ch	Memory Region Descriptor (MRGD_W3_50)	32	RW	0000_0000h
2660h	Memory Region Descriptor (MRGD_W0_51)	32	RW	0000_0000h
2664h	Memory Region Descriptor (MRGD_W1_51)	32	RW	0000_0000h
2668h	Memory Region Descriptor (MRGD_W2_51)	32	RW	0000_0000h
266Ch	Memory Region Descriptor (MRGD_W3_51)	32	RW	0000_0000h
2800h	Memory Region Descriptor (MRGD_W0_64)	32	RW	0000_0000h
2804h	Memory Region Descriptor (MRGD_W1_64)	32	RW	0000_0000h
2808h	Memory Region Descriptor (MRGD_W2_64)	32	RW	0000_0000h
280Ch	Memory Region Descriptor (MRGD_W3_64)	32	RW	0000_0000h
2820h	Memory Region Descriptor (MRGD_W0_65)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2824h	Memory Region Descriptor (MRGD_W1_65)	32	RW	0000_0000h
2828h	Memory Region Descriptor (MRGD_W2_65)	32	RW	0000_0000h
282Ch	Memory Region Descriptor (MRGD_W3_65)	32	RW	0000_0000h
2840h	Memory Region Descriptor (MRGD_W0_66)	32	RW	0000_0000h
2844h	Memory Region Descriptor (MRGD_W1_66)	32	RW	0000_0000h
2848h	Memory Region Descriptor (MRGD_W2_66)	32	RW	0000_0000h
284Ch	Memory Region Descriptor (MRGD_W3_66)	32	RW	0000_0000h
2860h	Memory Region Descriptor (MRGD_W0_67)	32	RW	0000_0000h
2864h	Memory Region Descriptor (MRGD_W1_67)	32	RW	0000_0000h
2868h	Memory Region Descriptor (MRGD_W2_67)	32	RW	0000_0000h
286Ch	Memory Region Descriptor (MRGD_W3_67)	32	RW	0000_0000h
2A00h	Memory Region Descriptor (MRGD_W0_80)	32	RW	0000_0000h
2A04h	Memory Region Descriptor (MRGD_W1_80)	32	RW	0000_0000h
2A08h	Memory Region Descriptor (MRGD_W2_80)	32	RW	0000_0000h
2A0Ch	Memory Region Descriptor (MRGD_W3_80)	32	RW	0000_0000h
2A20h	Memory Region Descriptor (MRGD_W0_81)	32	RW	0000_0000h
2A24h	Memory Region Descriptor (MRGD_W1_81)	32	RW	0000_0000h
2A28h	Memory Region Descriptor (MRGD_W2_81)	32	RW	0000_0000h
2A2Ch	Memory Region Descriptor (MRGD_W3_81)	32	RW	0000_0000h
2A40h	Memory Region Descriptor (MRGD_W0_82)	32	RW	0000_0000h
2A44h	Memory Region Descriptor (MRGD_W1_82)	32	RW	0000_0000h
2A48h	Memory Region Descriptor (MRGD_W2_82)	32	RW	0000_0000h
2A4Ch	Memory Region Descriptor (MRGD_W3_82)	32	RW	0000_0000h
2A60h	Memory Region Descriptor (MRGD_W0_83)	32	RW	0000_0000h
2A64h	Memory Region Descriptor (MRGD_W1_83)	32	RW	0000_0000h
2A68h	Memory Region Descriptor (MRGD_W2_83)	32	RW	0000_0000h
2A6Ch	Memory Region Descriptor (MRGD_W3_83)	32	RW	0000_0000h
2C00h	Memory Region Descriptor (MRGD_W0_96)	32	RW	0000_0000h
2C04h	Memory Region Descriptor (MRGD_W1_96)	32	RW	0000_0000h
2C08h	Memory Region Descriptor (MRGD_W2_96)	32	RW	0000_0000h
2C0Ch	Memory Region Descriptor (MRGD_W3_96)	32	RW	0000_0000h
2C20h	Memory Region Descriptor (MRGD_W0_97)	32	RW	0000_0000h
2C24h	Memory Region Descriptor (MRGD_W1_97)	32	RW	0000_0000h
2C28h	Memory Region Descriptor (MRGD_W2_97)	32	RW	0000_0000h
2C2Ch	Memory Region Descriptor (MRGD_W3_97)	32	RW	0000_0000h
2C40h	Memory Region Descriptor (MRGD_W0_98)	32	RW	0000_0000h
2C44h	Memory Region Descriptor (MRGD_W1_98)	32	RW	0000_0000h
2C48h	Memory Region Descriptor (MRGD_W2_98)	32	RW	0000_0000h
2C4Ch	Memory Region Descriptor (MRGD_W3_98)	32	RW	0000_0000h
2C60h	Memory Region Descriptor (MRGD_W0_99)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
2C64h	Memory Region Descriptor (MRGD_W1_99)	32	RW	0000_0000h
2C68h	Memory Region Descriptor (MRGD_W2_99)	32	RW	0000_0000h
2C6Ch	Memory Region Descriptor (MRGD_W3_99)	32	RW	0000_0000h
2C80h	Memory Region Descriptor (MRGD_W0_100)	32	RW	0000_0000h
2C84h	Memory Region Descriptor (MRGD_W1_100)	32	RW	0000_0000h
2C88h	Memory Region Descriptor (MRGD_W2_100)	32	RW	0000_0000h
2C8Ch	Memory Region Descriptor (MRGD_W3_100)	32	RW	0000_0000h
2CA0h	Memory Region Descriptor (MRGD_W0_101)	32	RW	0000_0000h
2CA4h	Memory Region Descriptor (MRGD_W1_101)	32	RW	0000_0000h
2CA8h	Memory Region Descriptor (MRGD_W2_101)	32	RW	0000_0000h
2CACh	Memory Region Descriptor (MRGD_W3_101)	32	RW	0000_0000h
2CC0h	Memory Region Descriptor (MRGD_W0_102)	32	RW	0000_0000h
2CC4h	Memory Region Descriptor (MRGD_W1_102)	32	RW	0000_0000h
2CC8h	Memory Region Descriptor (MRGD_W2_102)	32	RW	0000_0000h
2CCCh	Memory Region Descriptor (MRGD_W3_102)	32	RW	0000_0000h
2CE0h	Memory Region Descriptor (MRGD_W0_103)	32	RW	0000_0000h
2CE4h	Memory Region Descriptor (MRGD_W1_103)	32	RW	0000_0000h
2CE8h	Memory Region Descriptor (MRGD_W2_103)	32	RW	0000_0000h
2CECh	Memory Region Descriptor (MRGD_W3_103)	32	RW	0000_0000h

11.4.1.2 Control (CR)

11.4.1.2.1 Offset

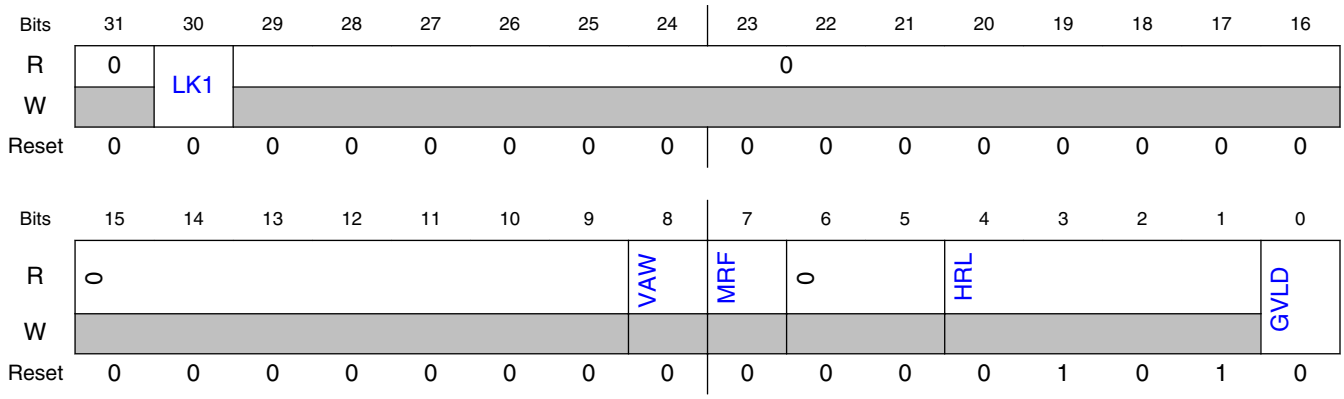
Register	Offset
CR	0h

11.4.1.2.2 Function

This register provides read-only status about the XRDC and a global enable bit for the entire module's operation.

Access: Secure Privileged Read/write

11.4.1.2.3 Diagram



11.4.1.2.4 Fields

Field	Function
31 —	Reserved
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to prohibit the ability to write this register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29-9 —	Reserved
8 VAW	Virtualization aware This read-only field signals that the domain assignments support the optional inclusion of a logical partition identifier (also known as an operating system number). 0b - Implementation is not virtualization aware. 1b - Implementation is virtualization aware.
7 MRF	Memory Region Format This read-only bit signals the format of the memory region descriptors and is defined by SoC requirements. 0b - Kinetis format based on Arm Cortex-M processor core definition. 1b - SMPU family format.
6-5 —	Reserved
4-1 HRL	Hardware Revision Level This read-only field specifies the XRDC's hardware and definition revision level. It can be read by software to determine the functional definition of the module. It is hardwired to 5 as it represents an architectural descendent from previous {system} memory protection units ({S}MPU).
0 GVLD	Global Valid (XRDC global enable/disable). 0b - XRDC is disabled. All accesses from all bus masters to all slaves are allowed. 1b - XRDC is enabled.

11.4.1.3 Hardware Configuration 0 (HWCFG0)

11.4.1.3.1 Offset

Register	Offset
HWCFG0	F0h

11.4.1.3.2 Function

This read-only register contains information on the XRDC's hardware configuration. Specifically, it defines the number of implemented domains and bus masters along with the number of instances of memory region controllers (MRCs) and peripheral access controllers (PACs). The register value at reset is device-specific. Attempted writes are error terminated.

Access: Secure Privileged Read

11.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NPAC				NMRC							
W																
Reset	0	0	0	1	0	0	1	1	0	0	0	0	0	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								NDID							
W																
Reset	0	0	0	0	1	1	0	1	0	0	0	0	0	1	1	1

11.4.1.3.4 Fields

Field	Function
31-28 MID	Module ID
27-24 NPAC	Number of PACs This field defines the number of PACs minus 1 in the device. Add one to the field value to get the actual number of peripheral access controllers [1-4].

Table continues on the next page...

Field	Function
23-16 NMRC	Number of MRCs This field defines the number of MRCs minus 1 in the device. Add one to the field value to get the actual number of memory region controllers [1-16].
15-8 NMSTR	Number of bus masters This read-only field defines the number of bus masters minus 1 in the device. Add one to the field value to get the actual number of bus masters [1-64].
7-0 NDID	Number of domains This read-only field defines the number of domains minus 1 in the device. Add one to the field value to get the actual number of domains [1-16].

11.4.1.4 Hardware Configuration 1 (HWCFG1)

11.4.1.4.1 Offset

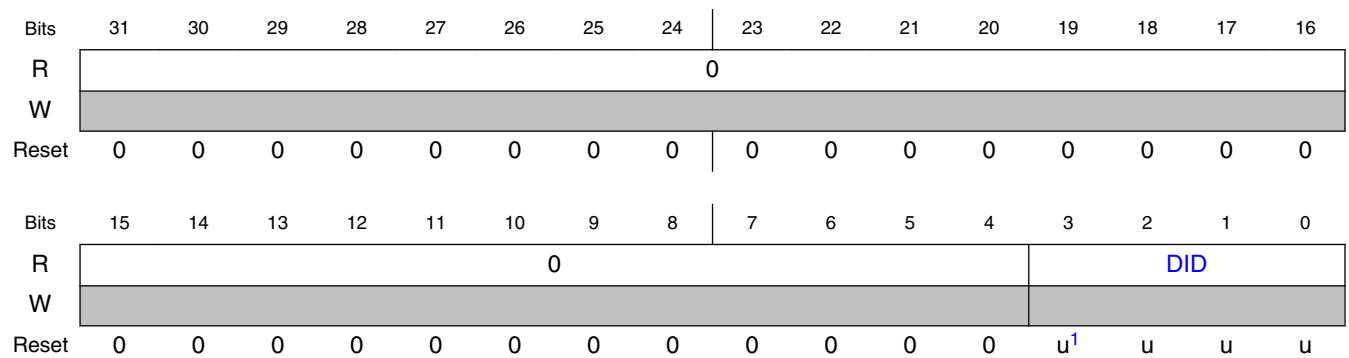
Register	Offset
HWCFG1	F4h

11.4.1.4.2 Function

This register contains information on the XRDC's hardware configuration. It provides a mechanism for software to determine its domain number by simply reading the register. See Domain error capture management for more details on typical usage. Attempted writes are error terminated.

Access: Secure Privileged Read

11.4.1.4.3 Diagram



1. Reset value is determined by current configuration of master accessing.

11.4.1.4.4 Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain identifier number This field provides the domain number [0-15] of the requesting bus master.

11.4.1.5 Hardware Configuration 2 (HWCFG2)

11.4.1.5.1 Offset

Register	Offset
HWCFG2	F8h

11.4.1.5.2 Function

This register contains information on the XRDC's hardware configuration. It provides a bitmap signaling the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG2 register is associated with bus masters [31-0]. Attempted writes are error terminated.

Access: Secure Privileged Read

11.4.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PIDP31	PIDP30	PIDP29	PIDP28	PIDP27	PIDP26	PIDP25	PIDP24	PIDP23	PIDP22	PIDP21	PIDP20	PIDP19	PIDP18	PIDP17	PIDP16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIDP15	PIDP14	PIDP13	PIDP12	PIDP11	PIDP10	PIDP9	PIDP8	PIDP7	PIDP6	PIDP5	PIDP4	PIDP3	PIDP2	PIDP1	PIDP0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.4.1.5.4 Fields

Field	Function
31 PIDP31	Process identifier Process identifier present from bus master 31. This field provides a bitmap to signal that bus master 31 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 31 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 31 sources a process identifier register to the XRDC_MDAC logic.
30 PIDP30	Process identifier Process identifier present from bus master 30. This field provides a bitmap to signal that bus master 30 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 30 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 30 sources a process identifier register to the XRDC_MDAC logic.
29 PIDP29	Process identifier Process identifier present from bus master 29. This field provides a bitmap to signal that bus master 29 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 29 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 29 sources a process identifier register to the XRDC_MDAC logic.
28 PIDP28	Process identifier Process identifier present from bus master 28. This field provides a bitmap to signal that bus master 28 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 28 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 28 sources a process identifier register to the XRDC_MDAC logic.
27 PIDP27	Process identifier Process identifier present from bus master 27. This field provides a bitmap to signal that bus master 27 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 27 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 27 sources a process identifier register to the XRDC_MDAC logic.
26 PIDP26	Process identifier Process identifier present from bus master 26. This field provides a bitmap to signal that bus master 26 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 26 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 26 sources a process identifier register to the XRDC_MDAC logic.
25 PIDP25	Process identifier Process identifier present from bus master 25. This field provides a bitmap to signal that bus master 25 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 25 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 25 sources a process identifier register to the XRDC_MDAC logic.

Table continues on the next page...

Field	Function
24	Process identifier
PIDP24	Process identifier present from bus master 24. This field provides a bitmap to signal that bus master 24 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 24 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 24 sources a process identifier register to the XRDC_MDAC logic.
23	Process identifier
PIDP23	Process identifier present from bus master 23. This field provides a bitmap to signal that bus master 23 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 23 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 23 sources a process identifier register to the XRDC_MDAC logic.
22	Process identifier
PIDP22	Process identifier present from bus master 22. This field provides a bitmap to signal that bus master 22 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 22 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 22 sources a process identifier register to the XRDC_MDAC logic.
21	Process identifier
PIDP21	Process identifier present from bus master 21. This field provides a bitmap to signal that bus master 21 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 21 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 21 sources a process identifier register to the XRDC_MDAC logic.
20	Process identifier
PIDP20	Process identifier present from bus master 20. This field provides a bitmap to signal that bus master 20 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 20 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 20 sources a process identifier register to the XRDC_MDAC logic.
19	Process identifier
PIDP19	Process identifier present from bus master 19. This field provides a bitmap to signal that bus master 19 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 19 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 19 sources a process identifier register to the XRDC_MDAC logic.
18	Process identifier
PIDP18	Process identifier present from bus master 18. This field provides a bitmap to signal that bus master 18 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 18 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 18 sources a process identifier register to the XRDC_MDAC logic.
17	Process identifier
PIDP17	Process identifier present from bus master 17. This field provides a bitmap to signal that bus master 17 sources a process identifier register to the XRDC_MDAC logic.

Table continues on the next page...

Field	Function
	0b - Bus master 17 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 17 sources a process identifier register to the XRDC_MDAC logic.
16 PIDP16	Process identifier Process identifier present from bus master 16. This field provides a bitmap to signal that bus master 16 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 16 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 16 sources a process identifier register to the XRDC_MDAC logic.
15 PIDP15	Process identifier Process identifier present from bus master 15. This field provides a bitmap to signal that bus master 15 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 15 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 15 sources a process identifier register to the XRDC_MDAC logic.
14 PIDP14	Process identifier Process identifier present from bus master 14. This field provides a bitmap to signal that bus master 14 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 14 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 14 sources a process identifier register to the XRDC_MDAC logic.
13 PIDP13	Process identifier Process identifier present from bus master 13. This field provides a bitmap to signal that bus master 13 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 13 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 13 sources a process identifier register to the XRDC_MDAC logic.
12 PIDP12	Process identifier Process identifier present from bus master 12. This field provides a bitmap to signal that bus master 12 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 12 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 12 sources a process identifier register to the XRDC_MDAC logic.
11 PIDP11	Process identifier Process identifier present from bus master 11. This field provides a bitmap to signal that bus master 11 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 11 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 11 sources a process identifier register to the XRDC_MDAC logic.
10 PIDP10	Process identifier Process identifier present from bus master 10. This field provides a bitmap to signal that bus master 10 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 10 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 10 sources a process identifier register to the XRDC_MDAC logic.
9 PIDP9	Process identifier

Table continues on the next page...

Field	Function
	Process identifier present from bus master 9. This field provides a bitmap to signal that bus master 9 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 9 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 9 sources a process identifier register to the XRDC_MDAC logic.
8 PIDP8	Process identifier Process identifier present from bus master 8. This field provides a bitmap to signal that bus master 8 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 8 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 8 sources a process identifier register to the XRDC_MDAC logic.
7 PIDP7	Process identifier Process identifier present from bus master 7. This field provides a bitmap to signal that bus master 7 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 7 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 7 sources a process identifier register to the XRDC_MDAC logic.
6 PIDP6	Process identifier Process identifier present from bus master 6. This field provides a bitmap to signal that bus master 6 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 6 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 6 sources a process identifier register to the XRDC_MDAC logic.
5 PIDP5	Process identifier Process identifier present from bus master 5. This field provides a bitmap to signal that bus master 5 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 5 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 5 sources a process identifier register to the XRDC_MDAC logic.
4 PIDP4	Process identifier Process identifier present from bus master 4. This field provides a bitmap to signal that bus master 4 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 4 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 4 sources a process identifier register to the XRDC_MDAC logic.
3 PIDP3	Process identifier Process identifier present from bus master 3. This field provides a bitmap to signal that bus master 3 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 3 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 3 sources a process identifier register to the XRDC_MDAC logic.
2 PIDP2	Process identifier Process identifier present from bus master 2. This field provides a bitmap to signal that bus master 2 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 2 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 2 sources a process identifier register to the XRDC_MDAC logic.

Table continues on the next page...

Field	Function
1 PIDP1	Process identifier Process identifier present from bus master 1. This field provides a bitmap to signal that bus master 1 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 1 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 1 sources a process identifier register to the XRDC_MDAC logic.
0 PIDP0	Process identifier Process identifier present from bus master 0. This field provides a bitmap to signal that bus master 0 sources a process identifier register to the XRDC_MDAC logic. 0b - Bus master 0 does not source a process identifier register. The XRDC_MDAC logic provides the needed PID for processor cores. 1b - Bus master 0 sources a process identifier register to the XRDC_MDAC logic.

11.4.1.6 Master Domain Assignment Configuration (MDACFG0 - MDACFG13)

11.4.1.6.1 Offset

For m = 0 to 13:

Register	Offset
MDACFGm	100h + (m × 1h)

11.4.1.6.2 Function

This register defines the number of implemented domain assignment registers for bus master m, where m+1 can specify from 1 to 64 bus masters. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. An all-zero value (NCM = 0, NMDAR = 0) indicates a non-existent bus master. Attempted writes are error terminated.

Typically, processor bus masters have one or more domain assignment registers, while non-processor masters have a single domain assignment register.

Access: Secure Privileged Read

11.4.1.6.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	See Register reset values.							

11.4.1.6.4 Register reset values

Register	Reset value
MDACFG0–MDACFG1	02h
MDACFG2	81h
MDACFG3	02h
MDACFG4–MDACFG13	81h
MDACFG14–MDACFG63	Register not supported

11.4.1.6.5 Fields

Field	Function
7 NCM	Non-CPU Master This read-only field signals that bus master m is a non-CPU master. It specifies that the format of the associated MDA_Wr_m register defines a non-processor domain assignment. This field is zero for a non-existent bus master. 0b - Bus master is a processor. 1b - Bus master is a non-processor.
6-4 —	Reserved
3-0 NMDAR	Number of master domain assignment registers for bus master m This read-only field specifies the number of registers associated with the master domain assignment register for a given bus master. The value is limited to the range [0-8], where zero indicates a non-existent bus master and non-zero values indicate the number of implemented registers associated with this MDAm.

11.4.1.7 Memory Region Configuration (MRCFG0 - MRCFG6)

11.4.1.7.1 Offset

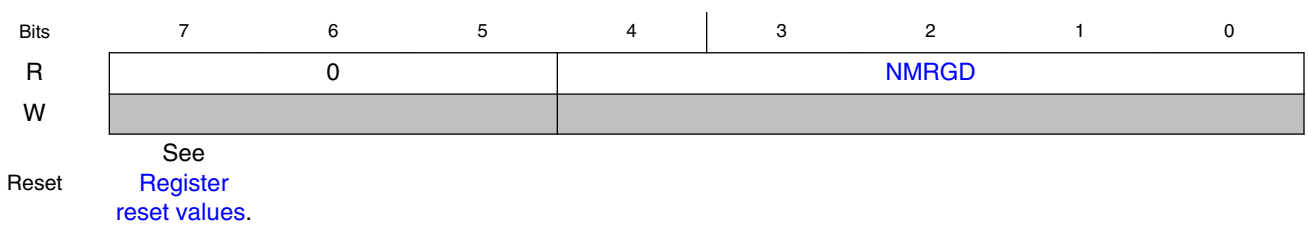
Register	Offset
MRCFG0	140h
MRCFG1	141h
MRCFG2	142h
MRCFG3	143h
MRCFG4	144h
MRCFG5	145h
MRCFG6	146h

11.4.1.7.2 Function

This read-only register defines the number of implemented memory region descriptors for each MRCr, where r+1 can specify up to 16 instances. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. A zero value indicates a non-existent memory region controller instance. Attempted writes are error terminated.

Access: Secure Privileged Read

11.4.1.7.3 Diagram



11.4.1.7.4 Register reset values

Register	Reset value
MRCFG0	04h
MRCFG1	08h
MRCFG2–MRCFG5	04h
MRCFG6	08h
MRCFG7–MRCFG15	Register not supported

11.4.1.7.5 Fields

Field	Function
7-5 —	Reserved
4-0 NMRGD	Number of memory region descriptors for memory region controller n Number of memory region descriptors for MRCr. This field specifies the number of memory region descriptors associated with a given memory region controller instance. The value is limited to the range [0-16], where zero indicates a non-existent MRC instance and non-zero values indicate the number of implemented memory region descriptors [4, 8,12,16] associated with the XRDC_MRCr submodule.

11.4.1.8 Domain Error Location (DERRLOC0 - DERRLOC7)

11.4.1.8.1 Offset

For d = 0 to 7:

Register	Offset
DERRLOCd	200h + (d × 4h)

11.4.1.8.2 Function

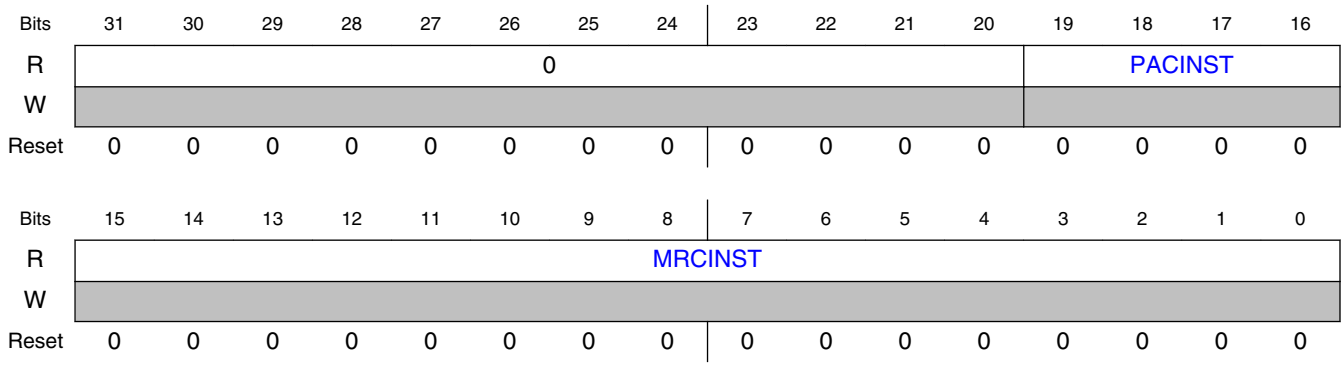
This array of read-only registers provide the instance number of the submodule where (an) access violation(s) occurred. These registers are organized as a word array, which is indexed by the faulting domain number, d. The two fields of this register provide a bitmap of instances associated with all submodules containing captured error information for that domain. These instance numbers are then used as indices into the DERR_W0_*i*, DERR_W1_*i*, DERR_W2_*i* and DERR_W3_*i* register arrays. See Domain error capture management for more details.

When an access violation is detected by either a memory region controller (MRC) or a peripheral access controller (PAC), address and attribute information of the offending access is captured. Using the faulting domainID number as the index, d, this array of read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*i* and DERR_W1_*i*, these registers provide the instance number details.

Attempted writes are error terminated.

Access: Secure Privileged Read

11.4.1.8.3 Diagram



11.4.1.8.4 Fields

Field	Function
31-20 —	Reserved
19-16 PACINST	PAC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the PAC. The least-significant bit of this field (bit 16) corresponds to PAC instance 0. The most-significant bit of this field (bit 19) corresponds to PAC instance 3, and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the PACs. **NOTE:** When a memory slot controller (MSC) is present, it will appear here in the same way as a PAC instance. For each bit in this field: • 0 - The peripheral access controller has not detected an access violation or is not physically present. • 1 - The peripheral access controller has detected one or more access violations for this domain.
15-0 MRCINST	MRC instance This field is a bitmap indicating the presence of a detected access violation for domain d in the implemented instances of the MRC. The least-significant bit of this field (bit 0) corresponds to MRC instance 0. The most-significant bit of this field (bit 3) corresponds to MRC instance 3, and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the MRCs. For each bit in this field: • 0 - The memory region controller has not detected an access violation or is not physically present. • 1 - The memory region controller has detected one or more access violations for this domain.

11.4.1.9 Domain Error Word0 (DERR_W0_0 - DERR_W0_19)

11.4.1.9.1 Offset

Register	Offset
DERR_W0_0	400h
DERR_W0_1	410h
DERR_W0_2	420h
DERR_W0_3	430h
DERR_W0_4	440h
DERR_W0_5	450h
DERR_W0_6	460h
DERR_W0_16	500h
DERR_W0_17	510h
DERR_W0_18	520h
DERR_W0_19	530h

11.4.1.9.2 Function

This read-only register array provides the address of an access violation detected by either a memory region controller (MRC) or a peripheral access controller (PAC). These registers are organized as a word array, which is indexed by the violating submodule instance number. That is, the index, i, of this array is the instance number of the submodule that detected the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory-mapped error capture detail registers are organized as 20 sequential 16-byte entries. The first 16 register sets associated with the MRCs. The following 4 registers sets associated with the PACs. Each 16-byte structure contains DERR_W0_i, DERR_W1_i, DERR_W2_i, and DERR_W3_i.

The error capture registers in the MRC and PAC submodules contain physical registers for each domain, but are organized in the DERR_W0_i and DERR_W1_i registers to provide the information for the requesting domain only. Thus, the access violation exception handler for each domain only has visibility into the captured error information for its own domain. See Domain error capture management for more details.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_i register. At that time, this register is cleared and re-enabled to capture the next access violation.

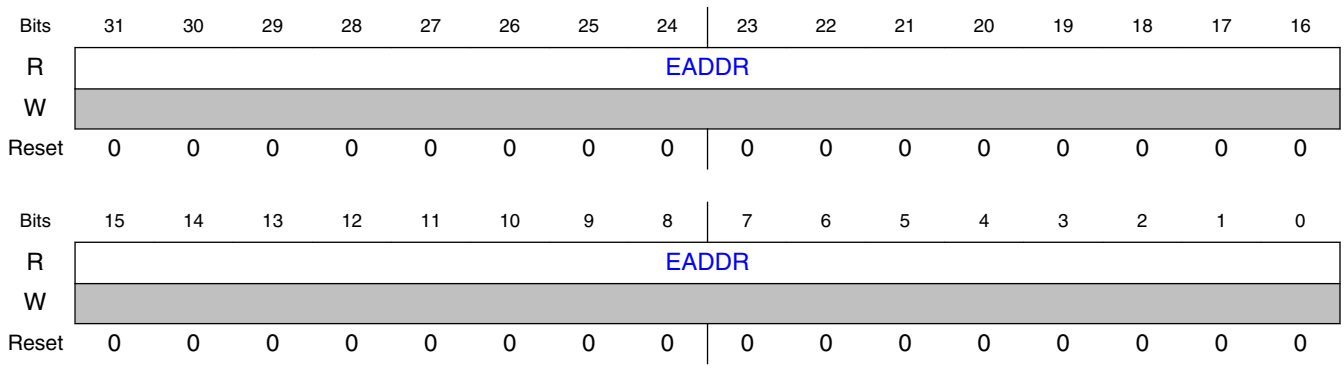
Attempted writes are error terminated as are attempted reads of an MRC or PAC instance that is not physically present.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W0_i will record only the error of the lowest slave index.

Access: Secure Privileged Read

11.4.1.9.3 Diagram



11.4.1.9.4 Fields

Field	Function
31-0 EADDR	Error address This is the access address that generated an access violation.

11.4.1.10 Domain Error Word1 (DERR_W1_0 - DERR_W1_19)

11.4.1.10.1 Offset

Register	Offset
DERR_W1_0	404h
DERR_W1_1	414h
DERR_W1_2	424h
DERR_W1_3	434h
DERR_W1_4	444h
DERR_W1_5	454h
DERR_W1_6	464h
DERR_W1_16	504h

Table continues on the next page...

Register	Offset
DERR_W1_17	514h
DERR_W1_18	524h
DERR_W1_19	534h

11.4.1.10.2 Function

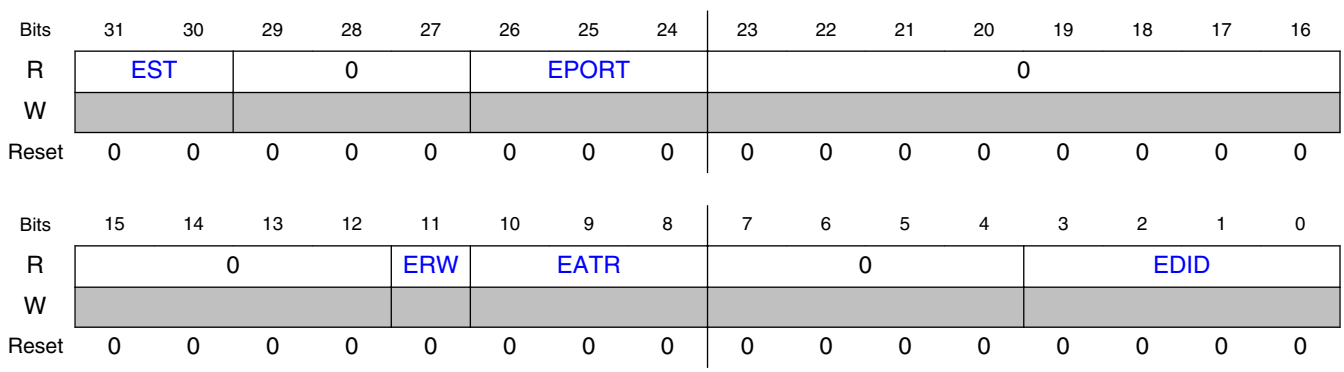
This read-only register array provides the attributes of an access violation detected by either a memory region controller (MRC) or a peripheral access controller (PAC). These registers are organized as a word array, which is indexed by the violating submodule instance number. See Domain Error Location (DERRLOC0 - DERRLOC7), Domain Error Word0 (DERR_W0_0 - DERR_W0_19), and Domain error capture management for more information.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W1_i will record only the error of the lowest slave index.

Access: Secure Privileged Read

11.4.1.10.3 Diagram



11.4.1.10.4 Fields

Field	Function
31-30 EST	Error state This field signals the state of access violations for this domain in this instance of the memory region controller or peripheral access controller. Once an access violation has been detected and the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured.

Table continues on the next page...

Field	Function
	After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_i. 00b - No access violation has been detected. 01b - No access violation has been detected. 10b - A single access violation has been detected. 11b - Multiple access violations for this domain have been detected by this submodule instance. Only the address and attribute information for the first error have been captured in DERR_W0_i and DERR_W1_i.
29-27 —	Reserved
26-24 EPORT	Error port This field identifies the encoded port number of the MRC that detected the access violation. The MRC port number connection is device-specific. See the chip configuration details for more information. For access violations detected by the PAC, this field is cleared.
23-12 —	Reserved
11 ERW	Error read/write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8 EATR	Error attributes This field captures certain attributes of the access violation. 000b - Secure user mode, instruction fetch access. 001b - Secure user mode, data access. 010b - Secure privileged mode, instruction fetch access. 011b - Secure privileged mode, data access. 100b - Nonsecure user mode, instruction fetch access. 101b - Nonsecure user mode, data access. 110b - Nonsecure privileged mode, instruction fetch access. 111b - Nonsecure privileged mode, data access.
7-4 —	Reserved
3-0 EDID	Error domain identifier This field captures the domain identifier of the access violation.

11.4.1.11 Domain Error Word3 (DERR_W3_0 - DERR_W3_19)

11.4.1.11.1 Offset

Register	Offset
DERR_W3_0	40Ch
DERR_W3_1	41Ch

Table continues on the next page...

Register	Offset
DERR_W3_2	42Ch
DERR_W3_3	43Ch
DERR_W3_4	44Ch
DERR_W3_5	45Ch
DERR_W3_6	46Ch
DERR_W3_16	50Ch
DERR_W3_17	51Ch
DERR_W3_18	52Ch
DERR_W3_19	53Ch

11.4.1.11.2 Function

This register is used to rearm the error capture logic and clear the DERR_W0_i and DERR_W1_i registers. After the domain access violation error details have been read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

A read of this location returns zeroes. Attempted reads of a memory region controller (MRC) or peripheral access controller (PAC) instance that is not physically present are error terminated.

See Domain error capture management for more details.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W3_i will record only the error of the lowest slave index.

11.4.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0													
W	RECR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.4.1.11.4 Fields

Field	Function
31-30 RECR	Rearm Error Capture registers This 2-bit, write-only field controls the rearming of the domain error capture registers. Once an access violation has been detected with the faulting address and attribute information stored, subsequent errors are simply recorded as an overrun condition without any data captured. Writing 01b to this field rearms the error capture mechanism and clears the DERR_W0_i and DERR_W1_i registers. A write of any value other than 01b has no effect.
29-0 —	Reserved

11.4.1.12 Process Identifier (PID0 - PID3)

11.4.1.12.1 Offset

Register	Offset
PID0	700h
PID1	704h
PID3	70Ch

11.4.1.12.2 Function

In the XRDC's access control definition, each processor has a corresponding process identifier (PID) which performs two important functions: the PID[5] bit defines the secure/nonsecure attribute associated with an executing task, and the entire value can be used to group tasks into different domains.

While certain processors include this register in their programming model definitions, others do not. This register is provided for those processors that do *not* include a PID register in their programming models. Secure privileged software saves and restores the PID as part of any context switch.

This data structure defines an array of 32-bit values, one per MDA module, that define the PID. Since this register resource is only applicable to processor cores, the data structure is typically sparsely populated. The HWCFG2 register provides a bitmap of the implemented PID*n* registers. This data structure is indexed using the corresponding MDA instance number.

For processors only supporting the 3-state access control model (SecurePriv, SecureUser, NonsecureUser), the nonsecure[n] output signal from the MDAC submodule is forced to zero while in privileged mode to enable precise state transitions between user and privileged modes. Specifically, the MDAC logic for bus master n forms the nonsecure attribute output signal as a function of 2 configuration bits: PIDm[TSM] and HWCFG{2,3}[n]. See Table 11-7.

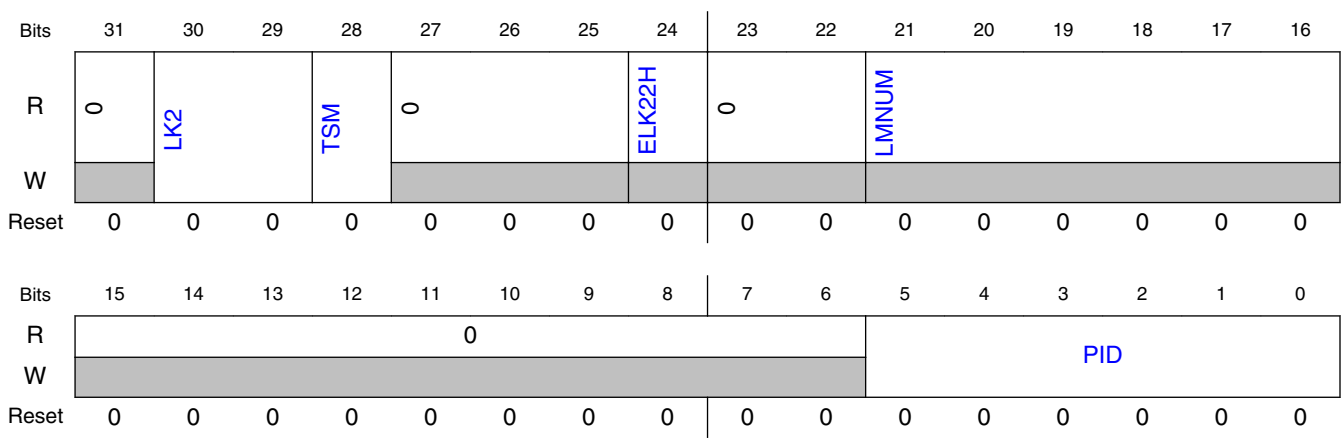
Depending on the operating clock domain of each MDAC instance, there may be optional information stored in the corresponding PIDm register to properly implement the LK2 = 2 functionality.

Reads of the PID*n* register return the contents of this register, or the PID*n* value directly sourced from a processor. For non-processor bus masters, this register does not exist and any attempted read is error terminated.

Table 11-7. Generation of the nonsecure_out[n] Attribute

Configuration {PID*n*[TSM], HWCFG{2,3}[n]}	local_nonsecure[n]	nonsecure_out[n]
00b	nonsecure_in[n]	local_nonsecure[n]
01b	nonsecure_in[n]	local_nonsecure[n] & ~priv_in[n]
10b	PID*n*[5]	local_nonsecure[n] & ~priv_in[n]
11b	pid_in[n][5]	local_nonsecure[n] & ~priv_in[n]

11.4.1.12.3 Diagram



11.4.1.12.4 Fields

Field	Function
31	Reserved

Table continues on the next page...

Field	Function
—	
30-29 LK2	Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, these bits individually remain asserted until the next reset. If the processor sources its PIDm directly, that is, HWCFG{2-3}[PIDPm = 1], then secure privileged reads of this memory location return zero for this field. 00b - Register can be written by any secure privileged write. 01b - Register can be written by any secure privileged write. 10b - Register can only be written by a secure privileged write from bus master m. 11b - Register is locked (read-only) until the next reset.
28 TSM	Three-state model If asserted, this bit indicates the associated processor core only supports the 3-state access control model. This indicator is “sticky”, that is, once set, it remains set until the next reset. For processors only supporting the 3-state access control model, this field must be set before loading any nonsecure value into the PID. See Table 11-7 for the specification of the privileged and nonsecure attributes generated by MDAC.
27-25 —	Reserved
24 ELK22H	Enable (LK2 = 2) special handling LK2 is in its standard form and LMNUM will be reserved and always read as 0.
23-22 —	Reserved
21-16 LMNUM	Locked Master NUMber This read-only field is only enabled when ELK22K = 1. It is written by the MDAC instance with the bus master number of the locking processor that set LK2 = 2, and used to qualify subsequent attempted writes of the register. The field is cleared when LK2 = [0,1,3] or ELK22H = 0.
15-6 —	Reserved
5-0 PID	Process identifier The PIDm[5] bit determines the secure (0) or nonsecure (1) attribute for transactions associated with the corresponding processor. If the processor sources its PID*n* directly, then secure privileged reads of this memory location return the processor register for this field.

11.4.1.13 Master Domain Assignment (MDA_W0_0_DFMT0 - MDA_W1_3_DFMT0)

11.4.1.13.1 Offset

Register	Offset
MDA_W0_0_DFMT0	800h
MDA_W1_0_DFMT0	804h
MDA_W0_1_DFMT0	820h
MDA_W1_1_DFMT0	824h
MDA_W0_3_DFMT0	860h
MDA_W1_3_DFMT0	864h

11.4.1.13.2 Function

The MDA_Wr_m registers provide a 2-dimensional data structure for assigning bus masters to domains. The number of implemented registers is defined by MDACFGm[NMDAR]. This per-master domain assignment is then repeated for each bus master (MDAm). Thus, m specifies the master number and r refers to the specific MDA register for a given bus master.

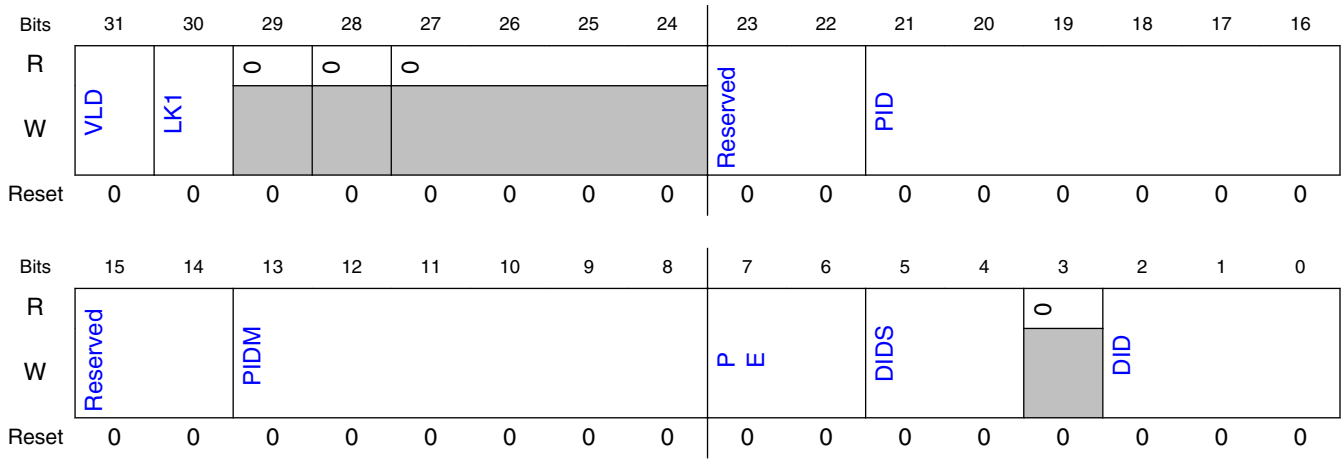
Each Wr within the MDAm structure is a word-sized definition; there are two formats supported, one for processor cores and another for non-processors. Processor masters typically support one or more Wr domain definitions, while non-processor masters support a single Wr.

The MDAC submodule is responsible for the generation of domain identifiers for every transaction from every bus master. If there is a single Wr for a given master, then the specified domain identifier is used directly. If there are multiple Wr values for a given master, then the MDAC evaluates the conditional terms to determine a “hit”. For all Wr hits, their corresponding domain identifiers are simply logically summed together (boolean OR). Use cases are typically expected to *hit in a single Wr* for a processor master. Special care is needed if *none* of the conditional terms hit in any Wr evaluation; for this case, the generated DID = 0 and software needs to be aware of any potential access rights granted for this DID.

Each MDA_Wr_m register has one of two programming models depending on the state of the domain format field, DFMT. The model described in this section is for DFMT = 0. This definition allows three different specifications of the DID for processors. The DFMT = 1 model is described in Master Domain Assignment (MDA_Wr_m_DFMT1).

Access: Secure Privileged Read/write

11.4.1.13.3 Diagram



11.4.1.13.4 Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLD] = 1. If CR[GVLD] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the XRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLD] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write. 1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment. 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved.
27-24 —	Reserved.
23-22 —	Reserved Always write 00b to this field.
21-16 PID	Process Identifier This field specifies that the process identifier is to be combined with the PIDM field and included in the domain hit determination. The optional inclusion of the PID and PIDM is controlled by the MDA_Wr_m[PE] field.

Table continues on the next page...

Field	Function
15-14 —	Reserved Always write 00b to this field.
13-8 PIDM	Process Identifier Mask This field provides a masking capability so that multiple process identifiers can be included as part of the domain hit determination. If a bit in the PIDM is set, then the corresponding bit of the PID is ignored in the comparison. The optional inclusion of the PID and PIDM is controlled by the MDA_Wr_m[PE] field.
7-6 PE	Process identifier enable This field controls the optional inclusion of the PID, qualified by PIDM, into the domain hit evaluation. It provides the ability to include inclusive or exclusive sets of masked PID values. 00b - No process identifier is included in the domain hit evaluation. 01b - No process identifier is included in the domain hit evaluation. 10b - The process identifier is included in the domain hit evaluation as defined by the expression: `partial_domain_hit = (PE == 10b) && ((PID & ~PIDM) == (XRDC_PIDn[PID] & ~PIDM))` 11b - The process identifier is included in the domain hit evaluation as defined by the expression: `partial_domain_hit = (PE == 11b) && ~((PID & ~PIDM) == (XRDC_PIDn[PID] & ~PIDM))`
5-4 DIDS	DID Select This field selects the source of the domain identifier. 00b - Use DID field of this register as the domain identifier. 01b - Use the input DID as the domain identifier. 10b - Use bits [3:2] of this register concatenated with the low-order 2 bits of the input DID (DID_in[1:0]) as the domain identifier. 11b - Reserved for future use.
3 —	Reserved
2-0 DID	Domain identifier

11.4.1.14 Master Domain Assignment (MDA_W0_2_DFMT1 - MDA_W0_13_DFMT1)

11.4.1.14.1 Offset

Register	Offset
MDA_W0_2_DFMT1	840h
MDA_W0_4_DFMT1	880h
MDA_W0_5_DFMT1	8A0h
MDA_W0_6_DFMT1	8C0h
MDA_W0_7_DFMT1	8E0h
MDA_W0_8_DFMT1	900h

Table continues on the next page...

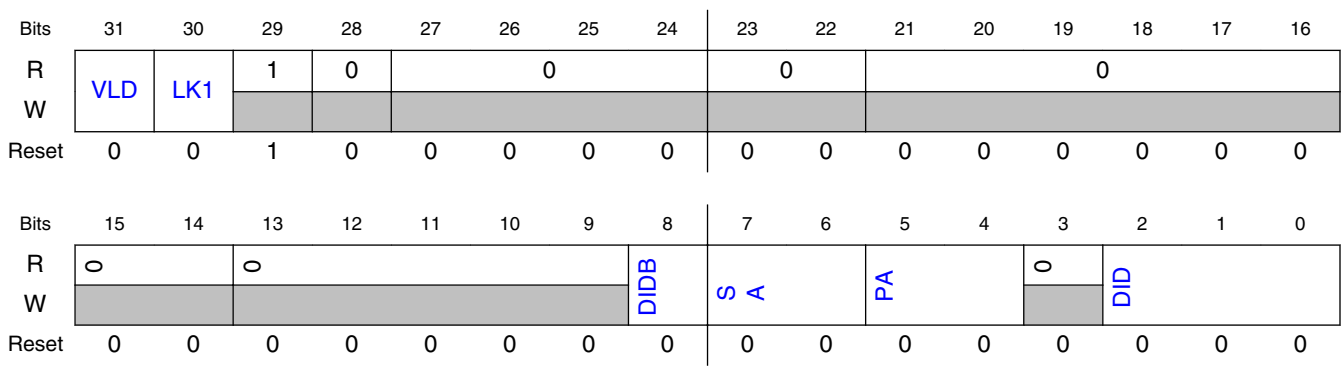
Register	Offset
MDA_W0_9_DFMT1	920h
MDA_W0_10_DFMT1	940h
MDA_W0_11_DFMT1	960h
MDA_W0_12_DFMT1	980h
MDA_W0_13_DFMT1	9A0h

11.4.1.14.2 Function

This register is identical to Master Domain Assignment (MDA_Wr_m_DFMT0) except that the domain format field, DFMT, is 1. This format supports two different specifications of the DID for non-core bus masters.

Access: Secure Privileged Read/write

11.4.1.14.3 Diagram



11.4.1.14.4 Fields

Field	Function
31 VLD	Valid This field indicates the domain assignment is valid. It is further qualified by CR[GVLD] = 1. If CR[GVLD] is cleared, the DID output is defined by the SoC-specific *default DID* value. See the XRDC chip configuration section for more details. If both MDA_Wr_m[VLD] and CR[GVLD] are asserted, the DID output is defined by the remaining contents of this register. 0b - The Wr domain assignment is invalid. 1b - The Wr domain assignment is valid.
30 LK1	1-bit Lock This field provides a locking mechanism that can be used to limit the ability to write the register. Once set, this bit remains asserted until the next reset. 0b - Register can be written by any secure privileged write.

Table continues on the next page...

Field	Function
	1b - Register is locked (read-only) until the next reset.
29 DFMT	Domain format Identifies this register's domain assignment. 0b - Processor-core domain assignment 1b - Non-processor domain assignment
28 —	Reserved.
27-24 —	Reserved.
23-22 —	Reserved.
21-16 —	Reserved
15-14 —	Reserved.
13-9 —	Reserved
8 DIDB	DID Bypass If asserted, this bit enables the bypassing of an input DID value as the domain identifier for this non-processor bus master. This capability allows non-processor bus masters, for example, a DMA to masquerade as a processor. Once set, this field is “sticky” and remains set until the next reset. 0b - Use MDAn[3:0] as the domain identifier. 1b - Use the DID input as the domain identifier.
7-6 SA	Secure attribute This field defines the secure/nonsecure attribute for non-processor cores. **NOTE:** The bus master’s input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to secure. 01b - Force the bus attribute for this master to nonsecure. 10b - Use the bus master’s secure/nonsecure attribute directly. 11b - Use the bus master’s secure/nonsecure attribute directly.
5-4 PA	Privileged attribute This field defines the privileged/user attribute for non-processor cores. **NOTE:** The bus master’s input privileged/user attribute is used if PA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to user. 01b - Force the bus attribute for this master to privileged. 10b - Use the bus master’s privileged/user attribute directly. 11b - Use the bus master’s privileged/user attribute directly.
3 —	Reserved
2-0 DID	Domain identifier

11.4.1.15 Peripheral Domain Access Control (PDAC_W0_8 - PDAC_W0_437)

11.4.1.15.1 Offset

Register	Offset
PDAC_W0_(c * 128) + s	1000h + (c × 400h) + (s × 8h)

11.4.1.15.2 Function

This register array, PDAC_W*r*_*s*, provides a two-dimensional data structure for defining access control policies per domain for each implemented slave peripheral. In the offset equation above, c is the instance number and s is the slot number. See the Access control policy table below. There are two word-sized registers assigned for each peripheral "address slot" (W*r*).

Each W*r* within the PDAC*s* structure is a word-sized definition; a total of two words (64 bits) is needed to define the access control policies for all the domains.

These registers also allow a hardware semaphore to be optionally included in the access control evaluation for peripherals shared by multiple domains. If enabled, the state of the semaphore dynamically modifies the access control policies so that only the domain "owning" it has write permission into the peripheral. The write permissions for all other domains are revoked based on the semaphore state. If the semaphore is not owned by any domain, the DxACP fields are evaluated normally.

The PAC submodule uses the peripheral access address to select the appropriate PDAC*s* register, and then uses the domain identifier associated with the access to select the appropriate DxACP field. The selected DxACP field is then evaluated against the attributes associated with the memory transaction to determine the validity of the access. See DxACP evaluation for more details. If allowed, the access proceeds, else an access violation is signaled, the transfer error terminated and the appropriate address and attribute information captured in the corresponding error detail registers.

Access: Secure Privileged Read/write

Table 11-8. Access control policy

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	r, w	r, w	r, w	r, w

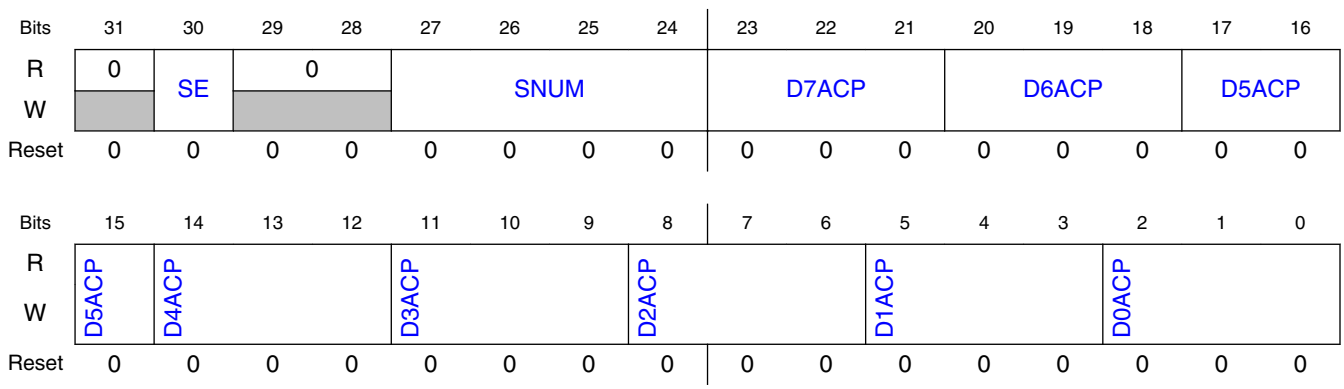
Table continues on the next page...

Table 11-8. Access control policy (continued)

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
110	r, w	r, w	r, w	none
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none
010	r, w	none	none	none
001	r	r	none	none
000	none	none	none	none

Access: Secure Privileged Read/write

11.4.1.15.3 Diagram



11.4.1.15.4 Fields

Field	Function
31 —	Reserved
30 SE	Semaphore enable This field indicates if a semaphore, SNUM, is to be included in the DxACP access evaluation. 0b - Do not include a semaphore in the DxACP evaluation. 1b - Include the semaphore defined by SNUM in the DxACP evaluation.
29-28 —	Reserved
27-24 SNUM	Semaphore number Include this hardware semaphore in the DxACP access evaluation.
23-21 D7ACP	Domain 7 access control policy

Table continues on the next page...

Field	Function
	This field defines the access control rights for the selected domain. The encodings specify read and write access capabilities based on the 4 operating states. These fields are only implemented for supported domainIDs; unsupported domainIDs are not implemented and default to a zero value providing no access rights. See the Access control policy table above.
20-18 D6ACP	Domain 6 access control policy See D7ACP above.
17-15 D5ACP	Domain 5 access control policy See D7ACP above.
14-12 D4ACP	Domain 4 access control policy See D7ACP above.
11-9 D3ACP	Domain 3 access control policy See D7ACP above.
8-6 D2ACP	Domain 2 access control policy See D7ACP above.
5-3 D1ACP	Domain 1 access control policy See D7ACP above.
2-0 D0ACP	Domain 0 access control policy See D7ACP above.

11.4.1.16 Peripheral Domain Access Control (PDAC_W1_8 - PDAC_W1_437)

11.4.1.16.1 Offset

Register	Offset
PDAC_W1_(c * 128) + s	1004h + (c × 400h) + (s × 8h)

11.4.1.16.2 Function

See Peripheral Domain Access Control (PDAC_W0_8 - PDAC_W0_437) for description.

Access: Secure Privileged Read/write

11.4.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	LK2		0					0			0			0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0			0				0			0			0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.4.1.16.4 Fields

Field	Function
31 VLD	Valid Valid. This field indicates the peripheral domain access control definition is valid. It is further qualified by CR[GVLD] = 1. If either CR[GVLD] or this VLD flag is cleared, all accesses to the peripheral are allowed. Any write to PDAC_W0_s clears the PDAC_W1_s[VLD] indicator so a coherent register state can be supported. 0b - The PDACs assignment is invalid. 1b - The PDACs assignment is valid.
30-29 LK2	Lock Lock. This 2-bit field provides a mechanism to limit writes to the PDACs register to protect its contents. Once set, these bits individually remain asserted until the next reset. 00b - Entire PDACs can be written. 01b - Entire PDACs can be written. 10b - Domain x can only update the DxACP field; no other PDACs fields can be written. 11b - PDACs is locked (read-only) until the next reset.
28-24 —	Reserved
23-21 —	Reserved.
20-18 —	Reserved.
17-15 —	Reserved.
14-12 —	Reserved.
11-9 —	Reserved.
8-6 —	Reserved.

Table continues on the next page...

Field	Function
5-3 —	Reserved.
2-0 —	Reserved.

11.4.1.17 Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_103)

11.4.1.17.1 Offset

Register	Offset
MRGD_W0_n * 16 + m	2000h + (n × 200h) + (m × 20h)

11.4.1.17.2 Function

The MRGD_W*o*_*n*_*m* registers provide a 2-dimensional data structure for defining access control policies (Table 11-9) per domain for each supported memory region. There are four word-size registers (W*o*) defined in each memory region. There are up to 16 memory region controllers (MRCs), *n*, each supporting up to 16 memory region descriptors (MRGD), *m*. The number of MRCs is defined by HWCFG0[NMRC].

Each instance of a MRC submodule can monitor up to 8 system buses concurrently.

NOTE

An AXI port counts as two system buses since the read address and write address channels operate independently.

Each W*o* within the MRGD*m* structure is a word-sized definition; a total of four words (128 bits, W0-W3) are needed to completely define the memory region descriptor.

XRDC uses a base address plus a size (0-modulo-size) to specify memory regions. This format provides addressing capabilities that are equivalent to those provided in the Arm Cortex-M processor core memory protection unit.

The region descriptor also allows a hardware semaphore to be optionally included in the access control evaluation for memory regions shared by multiple domains. If enabled, the state of the semaphore dynamically modifies the access control policies so that only the domain "owning" it has write permission into the region. The write permissions for all other domains are revoked based on the semaphore state. If the semaphore is not owned by any domain, the D*x*ACP fields are evaluated normally.

The MRC submodule evaluates multiple system bus access addresses versus the programmed address ranges to determine a region hit. Once a region hit has been determined, the MRC then uses the domain identifier associated with the access to select the appropriate D*x*ACP field from the corresponding MRGDn. The selected D*x*ACP field is then evaluated against the attributes associated with the memory transaction to determine the validity of the access. See Memory region descriptor hit determination and Memory region DxACP evaluation for more details. If allowed, the access proceeds, else an access violation is signaled, the transfer error terminated and the appropriate address and attribute information captured in the corresponding error detail registers.

11.4.1.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BASEADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BASEADDR											0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.4.1.17.4 Fields

Field	Function
31-5 BASEADDR	Base Address This field defines the most significant bits of the base address of the memory region (minimum size = 32 bytes). The Base Address is aligned to a 0-modulo-2(SZ+1) byte address.
4-0 —	Reserved

11.4.1.18 Memory Region Descriptor (MRGD_W1_0 - MRGD_W1_103)

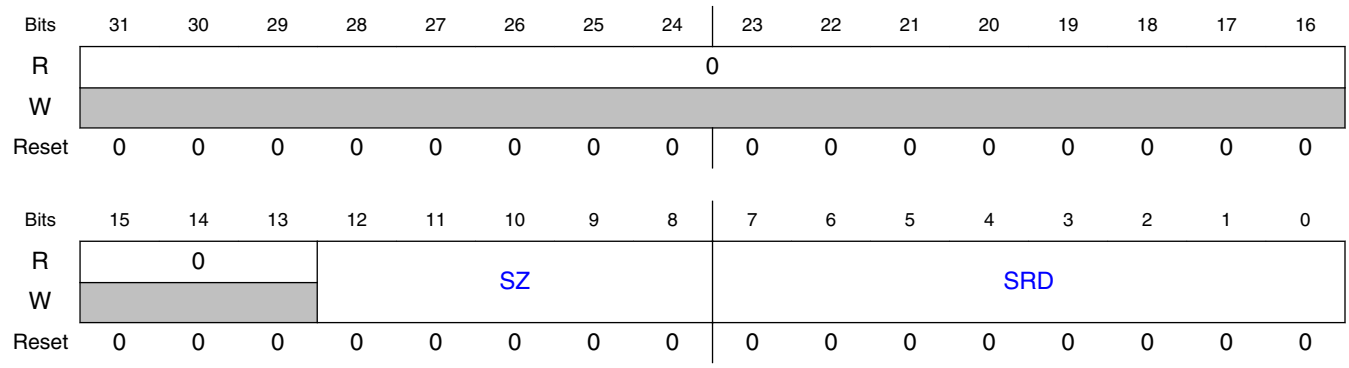
11.4.1.18.1 Offset

Register	Offset
MRGD_W1_n * 16 + m	2004h + (n × 200h) + (m × 20h)

11.4.1.18.2 Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_103) for description.

11.4.1.18.3 Diagram



11.4.1.18.4 Fields

Field	Function
31-13 —	Reserved
12-8 SZ	Region Size The region size is defined as 2(SZ+1) bytes.This field must be >= 4 as the smallest region size is 32 bytes, else the Arm definition of UNPREDICTABLE behavior applies.
7-0 SRD	Subregion disable For regions of 256 bytes or larger, this field is a bitmap enabling or disabling each of the eight equal-size subregions. SRD[0] disables the subregion with the lowest address range, while SRD[7] disables the subregion with the highest address range. If any SRD bits are asserted to disable a subregion and the total region size is less than 256 bytes, the Arm definition of UNPREDICTABLE behavior applies. 0 Subregion is enabled. 1 Subregion is disabled.

11.4.1.19 Memory Region Descriptor (MRGD_W2_0 - MRGD_W2_103)

11.4.1.19.1 Offset

Register	Offset
MRGD_W2_n * 16 + m	2008h + (n × 200h) + (m × 20h)

11.4.1.19.2 Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_103) for description.

Table 11-9. Access control policy

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	r, w	r, w	r, w	r, w
110	r, w	r, w	r, w	none
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none
010	r, w	none	none	none
001	r	r	none	none
000	none	none	none	none

11.4.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SE	0		SNUM				D7ACP			D6ACP			D5ACP	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	D5ACP	D4ACP			D3ACP			D2ACP			D1ACP			D0ACP		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.4.1.19.4 Fields

Field	Function
31 —	Reserved
30 SE	Semaphore enable This field indicates if a semaphore, SNUM, is to be included in the DxACP access evaluation. 0b - Do not include a semaphore in the DxACP evaluation. 1b - Include the semaphore defined by SNUM in the DxACP evaluation.
29-28 —	Reserved

Table continues on the next page...

Field	Function
27-24 SNUM	Semaphore number Include this hardware semaphore in the DxACP access evaluation.
23-21 D7ACP	Domain 7 access control policy This field defines the access control rights for the selected domain. The encodings specify read and write access capabilities based on the 4 operating states. These fields are only implemented for supported domainIDs; unsupported domainIDs are not implemented and default to a zero value providing no access rights. See Table 11-9.
20-18 D6ACP	Domain 6 access control policy See Domain 7 access control policy.
17-15 D5ACP	Domain 5 access control policy See Domain 7 access control policy.
14-12 D4ACP	Domain 4 access control policy See Domain 7 access control policy.
11-9 D3ACP	Domain 3 access control policy See Domain 7 access control policy.
8-6 D2ACP	Domain 2 access control policy See Domain 7 access control policy.
5-3 D1ACP	Domain 1 access control policy See Domain 7 access control policy.
2-0 D0ACP	Domain 0 access control policy See Domain 7 access control policy.

11.4.1.20 Memory Region Descriptor (MRGD_W3_0 - MRGD_W3_103)

11.4.1.20.1 Offset

Register	Offset
MRGD_W3_n * 16 + m	200Ch + (n × 200h) + (m × 20h)

11.4.1.20.2 Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_103) for description.

11.4.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	LK2		0					0			0			0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0			0				0			0			0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11.4.1.20.4 Fields

Field	Function
31 VLD	Valid This field indicates the memory region descriptor definition is valid. It is further qualified by CR[GVLD] = 1. If CR[GVLD] is cleared, then all accesses to the memory region are allowed. Any write to MRGD_W[0-2]_*n* clears the MRGD_W3_*n*[VLD] indicator so a coherent register state can be supported. 0b - The MRGDn assignment is invalid. 1b - The MRGDn assignment is valid.
30-29 LK2	Lock This 2-bit field provides a mechanism to limit writes to the MRGDn register to protect its contents. Once set, these bits individually remain asserted until the next reset. 00b - Entire MRGDn can be written. 01b - Reserved 10b - Domain *x* can only update the D*x*ACP field for domain *x*; no other ACP fields can be written. 11b - MRGDn is locked (read-only) until the next reset.
28-24 —	Reserved
23-21 —	Reserved.
20-18 —	Reserved.
17-15 —	Reserved.
14-12 —	Reserved.
11-9 —	Reserved.
8-6	Reserved.

Table continues on the next page...

Field	Function
—	
5-3 —	Reserved.
2-0 —	Reserved.

11.5 Functional description

This section provides more details on the operation and implementation of the XRDC submodules.

11.5.1 Manager (XRDC_MGR)

The MGR submodule is responsible for coordinating XRDC's programming model reads and writes.

The complete programming model is large, requiring four 4-Kbyte slave peripheral address space slots, although most devices define a sparsely populated subset. Additionally, the programming model is distributed across the various submodule instances (MDAC, MRC, PAC). The MGR submodule provides the required hardware management for all programming model accesses, generating and distributing the appropriate address decodes and write data to the MDAC, MRC and PAC submodule instances, collecting and combining all the register read data buses and responses.

MGR implements the Control Register, all the read-only hardware configuration registers (HWCFG[0-3], MDACFGn, MRCFG) and the program-visible versions of the domain error capture registers. To support the domain error reporting functionality, the MGR submodule collects the individual captured error indicator output signals from all the submodule instances and remaps them so they are organized into a instance bitmap by domain as specified by the DERRLOCn register array.

11.5.2 Master Domain Assignment Controller (XRDC_MDAC)

The MDAC submodule is responsible for the generation of the domainID on every memory transaction for every bus master in the device. The resulting domainID and {nonsecure, privileged} signals are generated and then treated as address attributes and associated with each transaction as it moves through the system.

MDAC modules that drive the domainIDs (DIDs) for processor cores include an option of processor-identifier (PID) matching in order to select between DIDs. This is accomplished by first enabling PID checking by setting XRDC_MDA_Wm_n[PE] = [2,3]. With PID matching enabled, a domain hit is determined by matching XRDC_MDA_Wm_n[PID] with the PID input or PID in the XRDC_PID register. The PIDM field can be used to match specific groups of PIDs.

As a simple example, assume a system with two domainIDs.

- domainID 1 is for critical tasks ("critical" is system-specific and may refer to timing, safety, and so on.).
- domainID 2 is for non-critical tasks.
- Two domainIDs require two MDA_Wm_n registers, MDA_W0 and MDA_W1.

Further assume the processor's task identifier defines whether a task is critical or non-critical as defined by system software.

- PIDs [0-15] are assigned to critical tasks.
- PIDs != [0-15] are assigned to noncritical tasks.

During startup, software initializes the MDA_W[0-1] registers. MDA_W0 defines DID = 1 for PIDs [0-15] and MDA_W1 defines DID = 2 for PIDs != [0-15].

1. MDA_W0 = 0x8000_0F81
    - VLD = 1
    - PID = 0x0
    - PIDM = 0xF
    - PE = 2
    - DIDS = 0
    - DID = 1
2. MDA_W1 = 0x8000_0FC2
    - VLD = 1
    - PID = 0x0
    - PIDM = 0xF
    - PE = 3
    - DIDS = 0
    - DID = 2

As the processor executes, it loads the appropriate task ID into its PID register as the task is started. The processor's PID register value is input to the corresponding MDAC module and used by the MDA_Wm comparison logic. The system then dynamically generates two domainIDs and the downstream XRDC access check logic can distinguish and enforce different access control rights based on the different domainIDs.

A special mode is provided for driving the DID of non-processor masters. This is enabled with the DID bypass (DIDB) bit. The MDAC has a DID input that can be driven by a master and used directly as the DID output. This allows variability in the DID as dictated by the master and is intended to be used by DMA masters that masquerade as, (that is, drive the DID of) the master that programmed it.

11.5.3 DxACP evaluation

Fundamental to the XRDC's operation is the actual access violation check performed by the memory region (MRC) and peripheral access (PAC) controllers.

Previous descriptions have detailed the domain assignment mechanisms and the tracking of the domain identifier as an address attribute through the system bus switching fabric(s). As transactions reach the slave memory and peripheral controllers, the address is used to select the appropriate memory region descriptor or peripheral access control register. Once the appropriate register has been selected, the next function is the actual access evaluation.

For this step, the domainID of the current transaction selects the appropriate 3-bit DxACP (domain "x" access control policy) from the source register. The DxACP field defines the access rights for the domain based on the 4-level model combined with the transaction's access attributes (read = 0/write = 1, secure = 0/nonsecure = 1, user = 0/privileged = 1).

The 4-level model is defined by the concatenation of two address attributes: secure/nonsecure and user/privileged attributes. These two attribute signals (nonsecure, privileged) and the resulting access levels are defined as:

```
if {nonsecure, privileged} == 0b01, then level = SecurePriv(ileged)
if {nonsecure, privileged} == 0b00, then level = SecureUser
if {nonsecure, privileged} == 0b11, then level = NonsecurePriv(ileged)
if {nonsecure, privileged} == 0b10, then level = NonsecureUser
```

Recall the resulting hierarchical access control model specifies:

```
SecurePriv(ileged) > SecureUser > NonsecurePriv(ileged) > NonsecureUser
```

The XRDC's evaluation of the access control policy is defined by Table 11-10.

Table 11-10. Domain x Access Control Policy Definition

	Allowable Accesses			
Policy	Secure Priv	Secure User	Nonsecure Priv	Nonsecure User
111	r, w	r, w	r, w	r, w
110	r, w	r, w	r, w	none

Table continues on the next page...

Table 11-10. Domain x Access Control Policy Definition (continued)

Policy	Allowable Accesses			
	Secure Priv	Secure User	Nonsecure Priv	Nonsecure User
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none
010	r, w	none	none	none
001	r	r	none	none
000	none	none	none	none

Unimplemented domains default to an all-zero DxACP field providing no access rights.

After the DxACP evaluation is complete and the access error signal generated, the transaction is allowed to continue if it has sufficient access rights, else the access is aborted with the address and attribute information captured in the appropriate error registers.

Recall XRDC optionally supports the inclusion of a hardware semaphore to *dynamically alter* the access control policy of a memory region or peripheral access. See Hardware semaphores and dynamic access rights for details.

11.5.4 Hardware semaphores and dynamic access rights

Both memory region descriptors and peripheral access control registers support the ability to optionally include a hardware semaphore in their access evaluations. This capability is provided to allow hardware enforcement of dynamic access rights based on the state of the semaphore for shared memory regions and/or shared peripherals.

If enabled (MRGD_W2_n[SE] = 1 or PDAC_W0_n[SE] = 1), the state of the specified hardware semaphore, as defined by MRGD_W2_n[SNUM] or PDAC_W0_n[SNUM]), is checked *before* the normal DxACP evaluation. On a write transaction, if the semaphore is not idle, that is, the semaphore state is non-zero and not owned by the requesting domain, the access is aborted and error terminated. Stated differently, writes into a semaphore-enabled address space are only allowed if the semaphore is idle, or is owned by the requesting domain. See "Semaphores2 (SEMA42)" chapter for more information.

The semaphore check is detailed in lines 84-92 of the C code description contained in MRC C code description. The comparable function in the PAC C code in Peripheral Access Controller (XRDC_PAC) is found in lines 38-45.

One additional note on this capability: since the semaphore module "only" supports up to 15 domains, that is, domainIDs [0-14], domainID 15 *cannot* be used for this function.

11.5.5 Memory Region Controller (XRDC_MRC)

The XRDC's Memory Region Controller provides domain-based, hardware access control for all system bus references targeted at non-peripheral memory spaces. Using pre-programmed *region descriptors* which define memory spaces and their associated access rights per domain identifier, the MRC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer. Memory references that have sufficient access control rights are allowed to complete, while references that are not mapped to any region descriptor or have insufficient rights are terminated with an access error response.

The XRDC architectural framework supports up to 16 MRC instances, where each MRC instance can support {4, 8, 12, 16} region descriptors, concurrently monitoring up to 8 slave memory ports. Note, monitoring an AXI protocol port counts as two ports since there are independent read address and write address channels. Conversely, an AHB connection counts as a single port.

A partial, simplified one-dimensional block diagram of an instance of the MRC module is shown in Figure 11-2. In this figure, a single slave bus port is shown. When the remaining slave ports are considered, the MRC hardware's two-dimensional connection matrix, broadcasting each system bus slave port access across the implemented memory region descriptors, is the resulting structure with the basic access evaluation macro shown as the replicated submodule block. In Figure 11-2, the system bus fabric slave port is shown on the left, the memory region descriptor registers are in the middle, and the peripheral bus interface on the right side. The evaluation macro contains two magnitude comparators connected to the start and end address registers from each region descriptor as well as the combinational logic blocks to determine the region hit and, based on the domainID associated with the reference, access violations.

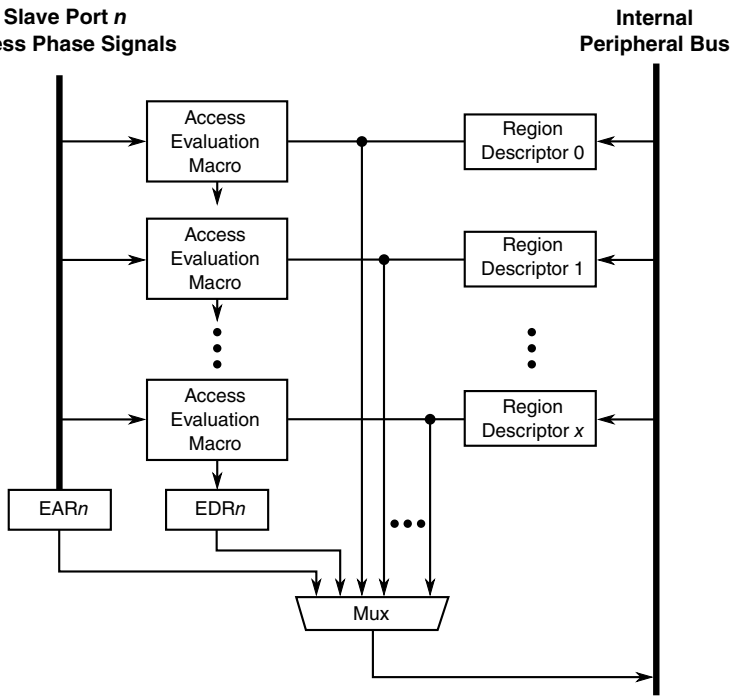


Figure 11-2. MRC access evaluation macro hardware structure

For each access, the MRC hardware performs a two-step evaluation process. First, the access is compared to all memory region descriptors to determine hit/miss status, and then, using the domainID associated with the reference plus the address attributes {write, nonsecure, privileged}, the access rights are examined using the method detailed in DxACP evaluation.

11.5.5.1 Memory region descriptor hit determination

To determine if the current reference hits in a given region, an equality comparator with a bit mask is used in conjunction with the region's BASEADDR and SZ. The boolean equation for this portion of the hit determination is defined as:

```
    region_hit_n = 0;
    if (mrgd_w1_n[sz] >= 4) {
      sz_mask  = (1 << (mrgd_w1_n[sz] + 1)) - 1;
      region_hit_n = (addr[31:5] | sz_mask) == (mrgd_w0_n[baseaddr] | sz_mask) |
(mrgdn_w1_n[sz] == 31);
```

The region_hit evaluation is then further qualified by the subregion disables as required.

Also, the region hit determination is based only on an address comparison of the first byte being accessed; that is, the MRC does *not* check the size of the access to make sure it entirely fits within the region.

11.5.5.2 Memory region DxACP evaluation

For each region descriptor hit, the MRC logic then evaluates the access rights defined by the MRGD_Wm_n registers. Specifically, the domainID attribute selects the appropriate DxACP field from the MRGD_W{2,3}_n registers and then the {write, nonsecure, privileged} address attributes are compared with the access policy defined by the selected DxACP. See DxACP evaluation for additional details on this function.

While CR[GVLD] = 1, and for each bus slave port being monitored, the MRC performs a **reduction-AND** of all the individual *(no_hit | error)* terms from each access evaluation macro. Recall as specified in Memory region descriptor hit determination, an *invalid* MRGD_Wm_n (MRGD_W3_n[VLD] = 0) forces the region_hit_n evaluation to be negated.

This expression then terminates the bus cycle with an error and reports an access error for three conditions:

1. If the access does not hit in *any* region descriptor, an access error is reported.
2. If the access hits in a single region descriptor and that region signals a domain violation, then an access error is reported.
3. If the access hits in multiple (overlapping) regions and *all* regions signal violations, then an access error is reported.

The third condition reflects that priority is given to *permission granting over access denying* for overlapping regions as this approach provides more flexibility to system software in the region descriptor assignments.

Unimplemented domain identifiers default to all-zero DxACP fields providing no access rights. Additionally, these fields are RAZ/WI.

11.5.5.3 MRC C code description

The functionality of the MRC evaluation is described in the C code of Listing 11-1 on page 239. This code begins with the description of the data structures corresponding to the MRC-associated registers (lines 1-12). Note the memory region descriptors are described as a 3-dimensional unsigned 32-bit integer array indexed by the MRC instance number, the RGD number within the instance and then the word offset within the region descriptor.

The access evaluation consists of four basic functions: a check of the XRDC's global valid bit (lines 43-45), the address hit/miss determination (lines 54-72), the selection of the DxACP field and its evaluation (lines 80-96) and finally, the combination of all the individual MRGD_Wm_n hit and error indicators to determine the final state of the memory reference (lines 99-134).

Listing 11-1. MRC Evaluation C Code Description

```
 1  //  MRC  registers
 2 unsigned int      xrdc_hwcfg0;          // contains NMRC -> # of MRC's
 3 unsigned char     xrdc_mrcfg[16];       // defines NMRGD -> # of RGD's
 4
 5 #define RGD_ADDR_MASK   0xfffffe0      // force sysAddr to modulo-32
 6 //
 7 // the memory region descriptors are modeled as a 3-dimensional array
 8 //  xrdc_mrgd[x][y][z] where:
 9 //       [x] = MRC instance number defined by xrdc_hwcfg0[nmrc]
10 //       [y] = RGDn within a given instance, defined by xrdc_mrcfgn[nmrgd]
11 //       [z] = Wordm within a given region descriptor
12 unsigned int      xrdc_mrgd[16][16][4]; // memory region descriptors
13
14 evaluate_MRC (MRCn, NMRGD, sysAddr, write, nonsecure, privileged, did)
15   unsigned int    MRCn;                 // MRCn instance number
16   unsigned int    NMRGD;                // number of RGDs for this MRC
17   unsigned int    sysAddr;              // system address
18   unsigned int    write;                // read   (=0), write      (=1)
19   unsigned int    nonsecure;            // secure (=0), nonsecure  (=1)
20   unsigned int    privileged;           // user   (=0), privileged (=1)
21   unsigned int    did;                  // domain identifier
22 {
23   unsigned int    mrgd_vld;             // local RGDn valid
24   unsigned int    mrgd_w2;              // local RGDn Word2
25   unsigned int    mrgd_w3;              // local RGDn Word3
26   unsigned int    mrgd_hit;             // calculated RGD "hit" signal
27   unsigned int    mrgd_error;           // calculated RGD "error" signal
28   unsigned int    mrgd_sema4;           // local copy of RGDn semaphore
29   unsigned int    mrc_hit;              // bitmask of RGDn hit signals
30   unsigned int    mrc_error;            // bitmask of RGDn access errors
31   unsigned int    final_error_state;    // final error state
32   unsigned int    memory_access_error;  // final access error indicator
33   unsigned int    mrgd_sz;              // local RGDn size
34   unsigned int    eff_srd;              // local RGDn subregion disable
35   unsigned int    sz_mask;              // local RGDn size mask
36   unsigned int    srd_mask;             // local RGDn subregion mask
37   unsigned int    j;                    // temp variable for subregion addr
38   unsigned int    i;                    // loop counter
39
40   mrc_hit   = 0;
41   mrc_error = 0;
42
43   if (xrdc_cr_gvld == 0) {              // check global valid
44       memory_access_error = 0;
45   } else {
46
47       for (i = 0; i < xrdc_mrcfg[MRCn]; i++) {
48           mrgd_hit    = 0;
49           mrgd_error  = 0;
50
51           // extract the region descriptor valid indicator
52           mrgd_vld = (xrdc_mrgd[MRCn][i][3] & 0x80000000) >> 31;
53
54           // first, determine if system address "hits" in the given MRDGn
55           // define the "effective SRD" - valid only if SZ >= 7
56           mrgd_sz = (xrdc_mrgd[MRCn][i][1] & 0x1F00) >> 8;
57           if (mrgd_sz < 7)
58               eff_srd = 0;                              // disable SRD
```

```
 59           else
 60               eff_srd = xrdc_mrgd[MRCn][i][2] & 0xFF;
 61
 62           mrgd_hit = 0;
 63           if ((mrgd_sz >= 4) {
 64               sz_mask  = (1 << (mrgd_sz + 1)) - 1;
 65               srd_mask =  7 << (mrgd_sz - 2);
 66               mrgd_hit = (sysAddr | sz_mask)
 67                        ==(xrdc_mrgd[MRCn][i][0] | sz_mask)
 68                        | (rgd_sz == 31);
 69
 70               j = (sysAddr & srd_mask) >> (mrgd_sz - 2);
 71               mrgd_hit = mrgd_hit & ((eff_srd >> j) ^ 1) & 1;
 72           }
 79
 80           // second, if there was a hit, evaluate DxACP including any sema4
 81           if (mrgd_hit == 1) {
 82               mrgd_w2 = xrdc_mrgd[MRCn][i][2];
 83               mrgd_w3 = xrdc_mrgd[MRCn][i][3];
 84               // semaphore check using a local copy
 85               mrgd_sema4 = sema4[(mrgd_w2 & 0x3f000000) >> 24];
 86               // check if it is enabled on write and
 87              //       if not idle, nor equal to (did+1)
 88               if (((mrgd_w2 & 0x40000000) == 0x40000000) &&  // se
 89                    (write == 1)                          &&  // wt
 90                    (mrgd_sema4 != 0)                     &&  // sema4 !=idle
 91                    (mrgd_sema4 != (did+1)))              {   // sema4 !=did
 92                   mrgd_error = 1;
 93               } else {
 94                   // perform standard DxACP check
 95                   mrgd_error = evaluate_DxACP(mrgd_w2, mrgd_w3, did,
\                                  nonsecure, privileged, write);
 96               }
 97           }
 98
 99           // accumulate the hit and error indicators
100           mrc_hit   = (mrc_hit   | (mrgd_hit   << i)) & 0xffff;
101           mrc_error = (mrc_error | (mrgd_error << i)) & 0xffff;
102       }
103
104
105       For each slave port monitored, MRC performs a reduction-AND of all
106       the individual (~hit | error) terms from each access evaluation.
107       This expression then terminates the bus cycle with an error & reports
108       an access error for three conditions:
109
110       1) If the access does not hit in *any* region descriptor,
111             then an access error is reported.
112       2) If the access hits in one region descriptor and
113          that region signals a violation,
114             then an access error is reported.
115       3) If the access hits in multiple (overlapping) regions and *all*
116          regions signal violations,
117             then an access error is reported.
118
119       The third condition reflects that priority is given to "permission
120       granting over access denying" for overlapping regions as this
121       approach provides more flexibility to system software in region
122       descriptor assignments.
123       */
124
125       final_error_state   = ((mrc_hit   ^ 0xffff)  // form ~mrc_hit
126                           |  mrc_error) & 0xffff;  // factor in error flags
127
128       if (final_error_state == 0xffff)
129           memory_access_error = 1;
130       else
131           memory_access_error = 0;
132   }
```

```
133   return memory_access_error;
134 }
```

11.5.6 Peripheral Access Controller (XRDC_PAC)

The peripheral access controller performs an access control evaluation similar to that of the memory region controller (Memory Region Controller (XRDC_MRC), but using dedicated "memory region descriptors" with one unique PDAC_W[0-1]_n register per slave peripheral slot. The presence of one region descriptor per slave peripheral address space slot in the PAC simplifies the required functionality compared to the MRC.

The functionality of the PAC evaluation is described in the C code of Listing 11-2 on page 241. This code begins with the description of the data structures corresponding to the PAC-associated registers (lines 1-3). It then performs a check of the global valid bit (CR[GVLD]) (lines 25-27), followed by the extraction of the peripheral slot number from the system address (line 32), then followed by the access of the selected PDAC_W[0-1]_n registers (lines 33-34). Next, a check of the optional inclusion of a hardware semaphore and its current state is performed (lines 37-46). Finally, if the access is still allowed, an evaluation of the appropriate DxACP permission field is made (line 48) as detailed in DxACP evaluation and the state of the peripheral access error control signal returned (line 52).

Listing 11-2. PAC Evaluation C Code Description

```
 1   // PAC registers
 2   unsigned int    xrdc_cr_gvld;
 3   unsigned int    xrdc_pdac[512][2];
 4
 5   evaluate_PAC (sysAddr, write, privileged, nonsecure, did)
 6       unsigned int    sysAddr;                    // system address
 7       unsigned int    write;                      // read/write attribute
 8       unsigned int    privileged;                 // user/privileged attribute
 9       unsigned int    nonsecure;                  // secure/nonsecure attribute
10       unsigned int    did;                        // domain identifier
11   {
12       unsigned int    ips_slot_number;            // f(system address)
13       unsigned int    pdacw0;                     // selected PAC_W0_n register
14       unsigned int    pdacw1;                     // selected PAC_W1_n register
15       unsigned int    pac_sema4;                 // local copy of referenced sema4
16       unsigned int    peripheral_access_error;
17
18       // the evaluation of the PACn involves a 4-step process:
19       //   1) select the appropriate PDACn using the system address
20       //   2) using the DID for the access, select the appropriate DxACP field
21       //   3) evaluate the state of the (optional) SEMA4 for this access
22       //   4) using the {secure/nonsecure, user/priviledged} attributes, eval DxACP
23
24       // first, check the global XRDC valid bit
25       if (xrdc_cr_gvld == 0) {
26           peripheral_access_error = 0;
27           }
28
29       else {
30           // Extract the peripheral slot number from the system memory map.
31           // Example for a standard 4k slot assignment:
```

```
32          // sysAddr[20:12] = slot select [0-511]
33          ips_slot_number = (sysAddr & 0x001ff000) >> 12;
34          pdacw0 = xrdc_pdac[ips_slot_number][0];
35          pdacw1 = xrdc_pdac[ips_slot_number][1];
36          peripheral_access_error = 0;
37          if ((pdacw1 & 0x80000000) == 0x80000000) {          // PDACn is valid
38              // semaphore check
39              pac_sema4 = sema4[(pdacw0 & 0x3f000000) >> 24]; // local copy
40              // check if it is enabled on a write and if not idle, nor == (did+1)
41              if (((pdacw0 & 0x40000000) == 0x40000000)   &&  // se
42                   (write == 1)                           &&  // wt
43                   (pac_sema4 != 0)                        &&  // sema4 != idle
44                   (pac_sema4 != (did+1)))                {     // sema4 != did
45                  peripheral_access_error = 1;
46              } else {
47                  // perform the standard DxACP check
48                  peripheral_access_error = evaluate_DxACP(pdacw0, pdacw1, did,
\                                                    nonsecure, privileged, write);
49              }
50          }
51      }
52      return peripheral_access_error;
53  }
```

11.6 Initialization information

Out of reset, CR[GVLD] is cleared and XRDC disabled allowing secure privileged startup code to configure the entire programming model. The initialization process typically is performed in 3 steps:

1. Read the various hardware configuration registers (HWCFG{0-3}, MDACFGn, MRCFGn) to obtain the implemented hardware capabilities for the device.
2. Using the information retrieved in Step 1 coupled with the desired domain architecture, program all the register data structures for domain assignment (MDA_Wm_n), memory region descriptors (MRGD_Wm_n) and peripheral access control register (PDAC_Wm_n). There are individual valid bits included in all these registers which typically would be asserted. Additionally, there are also lock mechanisms available if register settings need to be configured and marked as read-only.
3. Once the XRDC programming model is loaded, the CR is written with GVLD asserted. At that point, the XRDC is fully operational.

In this context, XRDC being fully operational means the XRDC_MDAC modules are generating the domainIDs and the slave memory and peripheral responders (XRDC_MRC, XRDC_PAC) are checking the access rights. Due to the distributed design hierarchy and the pipelined nature of the hardware system bus fabric, it can take multiple cycles for a generated DID to propagate through the entire design. Any previous DID is based on the master's default DID, and depending on the programmed access policies, could generate an access error response.

If incorrect error responses are being generated, the following approaches can be used to minimize or eliminate these extraneous access errors:

1. Minimize the amount of system bus traffic in the system while XRDC_CR[GVLD] is being set.
2. Guarantee that all addresses being accessed provide sufficient access rights for the default DID(s) as well as the newly-programmed DID(s). Once XRDC is fully operational, as confirmed by a read of XRDC_HWCFG1, the permissions for the default DID(s) can be removed.
3. In defining the DID assignments for the system, try to have the bus master that is programming and configuring the XRDC registers use the same DID, that is, its default DID, for both initialization and configuration as well as normal system operation.

11.7 Application information

As described in Introduction, the XRDC architecture is intended to provide a broad, highly-capable framework for access control, system memory protection and peripheral isolation. The resulting microarchitecture implementation is distributed throughout the core platform, highly configurable via hardware design parameters and software programmability. The considerable number of hardware design parameters allows the XRDC architecture to be a very scalable solution, able to service a wide spectrum of SoC devices from ultra-low end MCUs to multi-core high performance embedded microprocessors.

11.7.1 Master domain assignments

The typical use case related to master domain assignments is to include one (or more) processor core(s) in a single domain, possibly coupled with other bus master devices like DMA, etc. The definition of a domain may be static, based simply on the combination of a processor and other optional bus masters. Alternatively, the processor's operation may be configured to dynamically select between a very small number of domains. The maximum number of possible CPU domain definitions is limited by MDACFGn value. In particular, the optional inclusion of PID value(s) to be used to create multiple classes of CPU with different domain values.

For example, "critical" tasks, whether safety-critical, performance-critical, etc. can be grouped together in one domain and all other tasks into a second domain. Non-processor bus masters typically have a single MDA_Wm_n configuration register associated with

them. The domainID and {nonsecure, privileged} attributes are usually statically assigned, unless the module can "inherit" attributes from a processor programming it (such as certain modules like DMAs).

The MDA_Wm_n registers, along with the memory region descriptors and the peripheral domain access control registers all have lock features that allow the register resources to be converted into read-only resources to protect the written configuration.

11.7.2 Memory region descriptor management

There are two important concepts to consider in managing the memory region descriptors.

First, recall that the association between each MRCn instance and the slave memory regions it monitors is chip-specific. The device-specific configuration specifies the number of implemented memory region descriptors in a given instance and the specific port numbers associated with the slave memories being monitored.

Second, as detailed in Memory region DxACP evaluation, recall that, once the XRDC is enabled, *every memory reference must hit in one or more regions*, else an access violation is reported. These are also two other conditions were access errors are reported:

- If the access hits in a single region descriptor and that region signals a domain violation, then an access error is reported.
- If the access hits in multiple (overlapping) regions and *all* regions signal violations, then an access error is reported.

11.7.3 Domain error capture management

Recall when a domain access violation is detected by either a memory region controller or a peripheral access controller, address and attribute information of the offending access is captured. The DERRLOCn read-only registers provide additional information signaling the instance number of the submodule where the access violation(s) occurred. Since the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_n and DERR_W1_n, the DERRLOCn registers provide the instance number details.

These registers are organized as a word array, *indexed by the faulting domain number*, with the contents of each register providing a bitmap signaling the instance number(s) associated with all submodules containing captured error information for that domain.

It is recommended that the exception handler begin by reading the HWCFG1 register to determine its domainID. Next, it uses the just-retrieved domainID to index into the DERRLOCn array. The resulting DERRLOCn value is then examined to determine the instance number of the reporting MRC and/or PAC submodule.

There may be multiple access violations, across multiple instances, pending for a given domain. It is suggested that exception handler use a "find first one" instruction (alternatively known as "count leading zeroes") to quickly and efficiently find the instance number containing access violation details. Once the instance number has been determined, the captured error address (DERR_W0_n) and attribute information (DERR_W1_n) can easily be retrieved from the domain error registers. The 2-bit DERR_W1_n[EST] field provides information signaling whether one or more errors have been detected by the submodule instance. A special write to DERR_W3_n is required to reset (and rearm) the EST error capture state machine.

This process is repeated until all errors associated with a given domainID have been processed.

A graphical representation of this 2-step retrieval of the domain error address and attributes is shown in Figure 11-3.

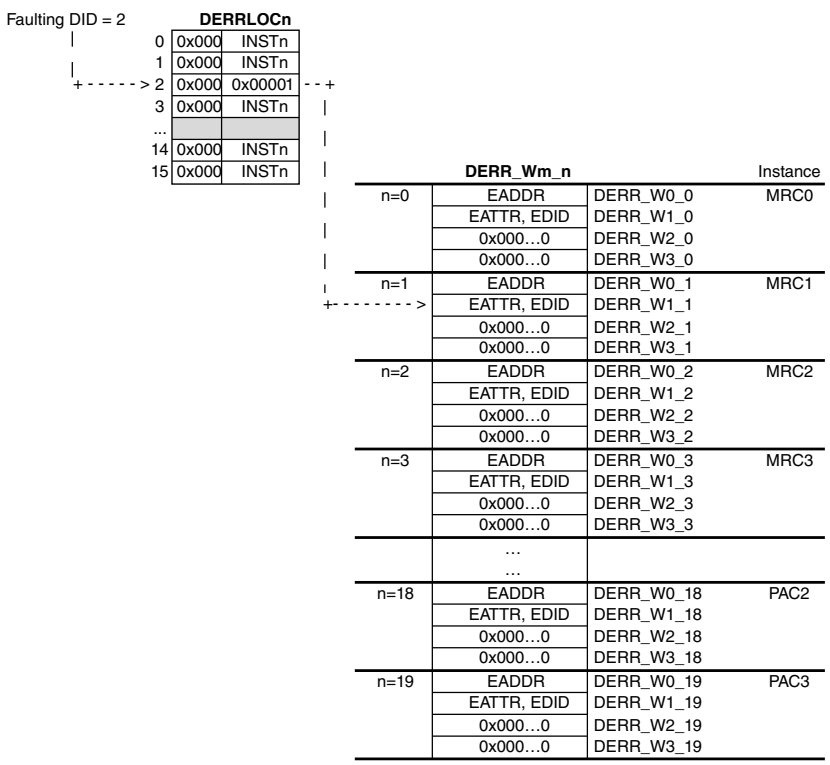


Figure 11-3. Two-step retrieval of domain error address and attributes

Chapter 12
Bus systems

12.1 Bus system

12.1.1 Main crossbar switch

i.MX 7ULP implements two instances of the AMBA Network Interconnect (NIC) for the bus fabric on the A7 domain. The NIC is a highly configurable and high performance AMBA-compliant network infrastructure that arbitrates between multiple AXI or AHB masters, to grant access to internal or external memories or other slave devices.

Figure 13-1 shows the configuration details of NIC0 and NIC1. The bus clock frequency ratios between NIC0 and NIC1 are 2:1 and 1:1.

NOTE

Register slice (RS) are not currently shown in the figure, but it (RS) may be used on critical time paths to meet the timing for the target frequency

12.1.1.1 Arbitration

In the NIC, each slave port (or master interface port) is configured separately in hardware to have an arbitration scheme which is either:

- a non-programmable round-robin (RR) scheme
- a programmable least recently granted (LRG) scheme

For i.MX 7ULP, all slave ports on NIC0 and NIC1 are configured for round-robin arbitration scheme, except the MMDC slave port on NIC0 is configured for LRG scheme.

12.1.1.2 Sparse interconnect

The NIC can be configured as a sparse interconnect, where some slave interface ports are not connected to all master interface ports. For this device, the following tables define the required interconnections for NIC0 and NIC1 crossbar switches.

Table 12-1. NIC0 sparse interconnect

	SRAM0	MMDC/ DDR	NIC1_M0
A7 ACE	Yes	Yes	Yes
NIC1_S1	Yes	Yes	No
DSI/LCDIF	Yes	Yes	Yes
GPU-3D	Yes	Yes	Yes
GPU-2D	Yes	Yes	Yes

Table 12-2. NIC1 sparse interconnect

				S0	S1	S2	S3	S4	S5	S6
Master	**Description**	**Width**	**Protocol**	**SRAM1**	**nic0**	**SecRAM**	**IPS0**	**IPS1**	**FB/ROM/ GPUR**	**AXBS**
M0	From nic0	64	AXI	Yes	No	Yes	Yes	Yes	Yes	Yes
M1	DMA1	64	AHB	Yes	Yes	Yes	Yes	Yes	Yes	Yes
M2	AXBS_IN	64	AHB	Yes	Yes	Yes	Yes	Yes	Yes	No
M3	Security(CAAM)[1]	32	AXI	Yes	Yes	Yes	No	No	Yes	Yes
M4	USB_OTG0/1	32	AXI	Yes	Yes	Yes	No	No	Yes	Yes
M5	VIU	64	AHB	Yes	Yes	Yes	No	No	Yes	Yes
M6	uSDHC0	32	AXI	Yes	Yes	Yes	No	No	Yes	Yes
M7	uSDHC1	32	AXI	Yes	Yes	Yes	No	No	Yes	Yes

1. For complete chapter, see the i.MX7ULP Security Reference Manual

12.1.1.3 Latency through NIC and Related Bus Gaskets

Expected latency associated with the NIC arbiter and related bus gaskets is illustrated below. Table 12-3 shows the Read address, Write address, and Write data channels which are initiated by the bus master. Table 12-4 shows the Read data and Write response channels which are initiated from the bus slaves in response to read or write requests.

Table 12-3. Latencies for Address and Write Data Channels

Component	Master Port Clock Cycles	NIC Clock Cycles	Slave Port Clock Cycles
AHB2AXI Gasket on master port	1	--	--
Synchronizer on master port	1	2	--
Downsizer or Expander on master port	--	1 (Downsizer)	--
Arbiter: address decode	--	1	--
Arbiter: arbitration	--	1	--
Downsizer on slave port	--	1	--
Synchronizer on slave port	--	1	2
AXI2AHB Gasket on slave port	--	--	2 (write data channel)
Register Slice	--	--	1

Table 12-4. Latencies for Read Data and Write Response Channels

Component	Master Port Clock Cycles	NIC Clock Cycles	Slave Port Clock Cycles
Register Slice	--	--	1
AXI2AHB Gasket on slave port	--	--	2 (write response)
Synchronizer on slave port	--	2	1
Downsizer on slave port	--	1 (read data path)	--
Arbiter	--	--	--
Downsizer or Expander on master port	--	1 (read data path)	--
Synchronizer on master port	2	1	--
AHB2AXI on master port	1 (write response)	--	--

12.1.2 AHB crossbar switch

i.MX 7ULP implements a 4x5 AHB XBar Switch (AXBS) in the M4 domain. The AXBS uses a 64-bit internal data path. If the slave port is simultaneously requested by more than one master port, arbitration logic exists in the crossbar to allow the higher priority master port to be granted access to the slave, while stalling the other requestor(s) until that transaction has completed parking.

See the following figure.

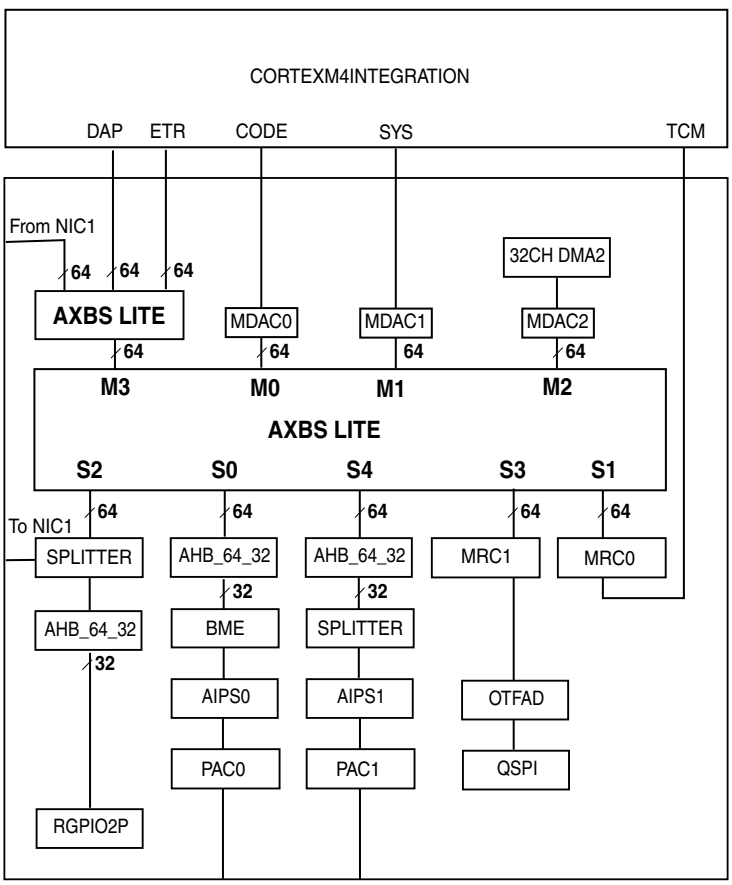


Figure 12-1. Crossbar switch bus diagram

12.1.2.1 Arbitration

The AXBS supports a fixed arbitration scheme. The M4 master ports have lower priority than other bus masters connected to this crossbar switch.

12.1.2.2 Access latency

The AXBS has low latency throughput when slave port is not busy:

- Single clock (zero-wait-state) accesses through AXBS when slave port is not busy or is parked on the requesting master port.
- 1 cycle penalty through AXBS when slave port is not busy, but parked on another master port.

Chapter 13
Network Interconnect Bus System (NIC-301)

13.1 Chip-specific NIC-301 information

Table 13-1. Reference links to related information

Topic	Related module	Reference
Full description	NIC-301	NIC-301
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

13.1.1 AMBA Network Interconnect (NIC)

The NIC is a highly configurable and high performance AMBA-compliant network infrastructure which arbitrates between multiple AXI or AHB masters to grant access to internal or external memories or other slave devices. It supports connectivity between several slave and master ports for parallel processing. It uses a hybrid round-robin arbitration scheme and contains frequency converters, data width converters, bus protocol converter, and AXI channel buffers. In order to support multicore enhancements, additional side-band signals are needed to communicate the domain attributes. The following table shows the configuration of the NIC.

Table 13-2. NIC configuration

Parameter	Description
Name	AMBA Network Interconnect (NIC)
Instances	2
Configurable features	Refer to Main crossbar switch and AHB crossbar switch

Table continues on the next page...

Table 13-2. NIC configuration (continued)

Parameter	Description
Interface speed	NA
External I/O pins	NA

13.1.2 Write buffer mechanism for NIC1

To achieve higher performance, there are write buffers in the master port of NIC1 for those bufferable write transactions from the real time domain. All those AHB bufferable write data beats receive an automatic OKAY response from the NIC1 irrespective of the slave's response. This means that if the NIC1 receives an error response, it does not feed it back to the master. User can disable the early write response feature of the AHB master port of NIC1 by programming the fn_mod_ahb register bit fields of NIC ASIB to 0x03. The address of this register is 0x41D44028 (Only AHB ASIB port has this early response feature). See Table 13-8

The CA7 L2 memory system has a synchronous abort mechanism and an asynchronous abort mechanism; see External aborts handling in Arm Cortex-A7 MPCore Technical Reference Manual, Revision r0p5. The abort exception is masked by CPSR.A bit by default; see Program Status Register in Arm Architecture v7AR Reference Manual.

13.2 Overview

This section provides an overview of the NIC-301 (Network Inter-Connect) AMBA bus arbiter IP.

The NIC-301 (by Arm Ltd.) is a configurable AMBA bus arbiter between several masters and slaves. The NIC-301 IP is designed so that many configuration options are selected at the hardware design stage, determined by SoC characteristics and needs, while several other configuration options are software-controlled.

This chapter covers in brief the NIC-301 functionality, while providing configuration details on the NIC-301 instances used in the chip. For complete details on the NIC-301 design, see the Arm specification, *AMBA® Network Interconnect (NIC-301) Technical Reference Manual, version r2p3*.

NOTE

The NIC-301 default settings are configured by NXP's board support package (BSP), and in most cases should not be modified by the customer. The default settings have gone

through exhaustive testing during the validation of the part, and have proven to work well for the part's intended target applications. Changes to the default settings may result in a degradation in system performance.

13.2.1 Block diagram

The NIC-301 AMBA arbiter (or "CoreLink Network Interconnect") by Arm, provides configurable AMBA-based interconnect logic, for connecting a number of masters (initiators) to several slaves (targets), via configurable bus switches and bridging components.

The bus system is composed of two instances: NIC0 and NIC1.

Each instance can include one or more bus switches, with additional logic. This block provides details of the various instances and the selected configuration parameters. The top level diagram of the bus system is shown in the following figure.

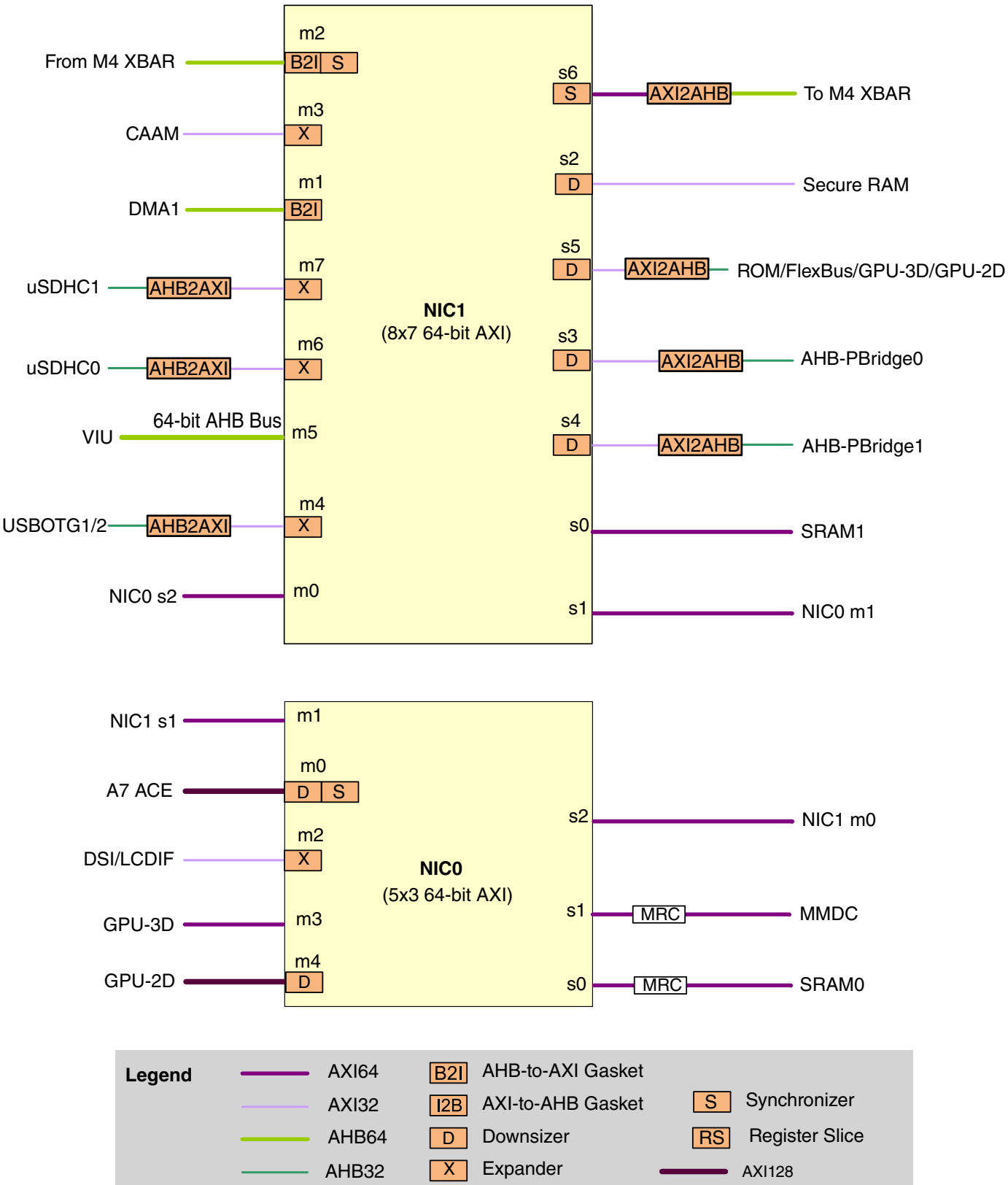


Figure 13-1. NIC-301 Bus System

13.2.2 NIC-301 Main Features

Key features of the NIC-301 module include the following:

- Address space memory mapping.
- Programmer's view, for software-configured parameters, via "GPV" ports.
- Support for cross-clock domain synchronization.

13.2.3 Modes and Operations

The NIC-301 supports a normal functional mode only.

13.3 External Signals

There are no external I/O interfaces for NIC-301.

13.4 Memory Map and Register Definition

This section includes the block memory map and detailed register descriptions.

In this device, no external GPV port is provided. GPV is internally located within the NIC-301. CPU can directly program the NIC0 GPV and NIC1 GPV, using the same memory map.

The base address of NIC0 GPV is 0x41C0_0000. The base address of NIC1 GPV is 0x41D0_0000. Their base address are applicable to both the CA7 and CM4 core

Table 13-3. NIC0/NIC1 GPV memory allocations

GPV	Allocation	Size	Chip Address	
			Start	End
NIC1 GPV	NIC1 internal global programmer's view	1MB	41D0_0000	41DF_FFFF
NIC0 GPV	NIC0 internal global programmer's view	1MB	41C0_0000	41CF_FFFF

13.4.1 Memory Map

The NIC-301 memory map, is dependent on the selected configuration option at time of creation.

A "template" map is provided in Table 13-4 below.

13.4.2 Configuration programmers model

The GPV's contain configuration registers, partitioned into a number of individual 4KB blocks.

The general structure of the registers is provided by the following tables:

- Address map of the programmers model, Table 13-4
- AMIB Registers, Table 13-5 and Table 13-6
- ASIB Registers, Table 13-7 and Table 13-8

Table 13-4. Address map of the programmers model

Address Offset from Base Address	Registers	Notes
0x000F_F000	Internal interface p registers	Maximum p = 61
	...	Note [1]
0x000C_4000	Internal interface 2 registers	
0x000C_3000	Internal interface 1 registers	
0x000C_2000	Internal interface 0 registers	
0x000C_1000	Slave interface m registers	Maximum m = 127
	...	Note [2]
0x0004_4000	Slave interface 2 registers	
0x0004_3000	Slave interface 1 registers	
0x0004_2000	Slave interface 0 registers	
0x0004_1000	Master interface n registers	Maximum n = 63
	...	Note [3]
0x0000_4000	Master interface 2 registers	
0x0000_3000	Master interface 1 registers	
0x0000_2000	Master interface 0 registers	
0x0000_1000	ID registers	
0x0000_0000	Address control registers	Configurable base address [4]

1. Index refers to BI registers index
2. Index refer to ASIB registers index
3. Index refer to AMIB registers index
4. Reserved for internal use

13.4.2.1 Address control and ID registers

Registers at offsets 0x0–0xFFC are reserved for internal use.

13.4.2.2 AMBA master interface block (AMIB) configuration registers

The table below lists only the registers that affect the user. All other addresses are treated as "reserved". Non-implemented or reserved addresses inside NIC domains are read as ZEROS and writes operations are ignored.

The NIC0 GPV base address is GPV_BASE = 0x41C00000. The NIC1 GPV base address is GPV_BASE = 0x41D00000.

Table 13-5. NIC0 AMIB configuration registers

Register Name	AMIB number	Absolute Address	Reset Value	Notes
fn_mod_bm_iss	0	GPV0_BASE + 0x2000 + 0x008	0	
fn_mod_bm_iss	1	GPV0_BASE + 0x3000 + 0x008	0	
fn_mod_bm_iss	2	GPV0_BASE + 0x4000 + 0x008	0	
wr_tidemark	1	GPV0_BASE + 0x3000 + 0x040	4	1
fn_mod	0	GPV0_BASE + 0x2000 + 0x108	0	
fn_mod	1	GPV0_BASE + 0x3000 + 0x108	0	

1. The WFIFO depth is 4.

Table 13-6. NIC1 AMIB configuration registers

Register Name	AMIB number	Absolute Address	Reset Value	Comment
fn_mod_bm_iss	0	GPV1_BASE + 0x2000 + 0x008	0	
fn_mod_bm_iss	1	GPV1_BASE + 0x3000 + 0x008	0	
fn_mod_bm_iss	2	GPV1_BASE + 0x4000 + 0x008	0	
fn_mod_bm_iss	3	GPV1_BASE + 0x5000 + 0x008	0	
fn_mod_bm_iss	4	GPV1_BASE + 0x6000 + 0x008	0	
fn_mod_bm_iss	5	GPV1_BASE + 0x7000 + 0x008	0	
fn_mod_bm_iss	6	GPV1_BASE + 0x8000 + 0x008	0	
fn_mod2	2	GPV1_BASE + 0x4000 + 0x024	0	

Table continues on the next page...

Table 13-6. NIC1 AMIB configuration registers (continued)

Register Name	AMIB number	Absolute Address	Reset Value	Comment
fn_mod2	3	GPV1_BASE + 0x5000 + 0x024	0	
fn_mod2	4	GPV1_BASE + 0x6000 + 0x024	0	
fn_mod2	5	GPV1_BASE + 0x7000 + 0x024	0	
wr_tidemark	0	GPV1_BASE + 0x2000 + 0x040	4	1
wr_tidemark	2	GPV1_BASE + 0x4000 + 0x040	4	1
wr_tidemark	5	GPV1_BASE + 0x7000 + 0x040	4	1
wr_tidemark	6	GPV1_BASE + 0x8000 + 0x040	4	1
fn_mod	0	GPV1_BASE + 0x2000 + 0x108	0	
fn_mod	1	GPV1_BASE + 0x3000 + 0x108	0	
fn_mod	2	GPV1_BASE + 0x4000 + 0x108	0	
fn_mod	3	GPV1_BASE + 0x5000 + 0x108	0	
fn_mod	4	GPV1_BASE + 0x6000 + 0x108	0	
fn_mod	5	GPV1_BASE + 0x7000 + 0x108	0	
fn_mod	6	GPV1_BASE + 0x8000 + 0x108	0	

1. The WFIFO depth is 4

13.4.2.3 ASIB (AMBA slave interface block) configuration registers

The table below lists only the registers that affect the user. All other addresses are treated as "reserved". Non-implemented or reserved addresses inside NIC domains are read as ZEROS and writes operations are ignored.

Table 13-7. NIC0 ASIB registers

Register Name	ASIB number	Absolute Address	Reset Value	Comment
fn_mod2	0	GPV0_BASE + 0x42000 + 0x024	0	
fn_mod2	4	GPV0_BASE + 0x46000 + 0x024	0	
wr_tidemark	0	GPV0_BASE + 0x42000 + 0x040	4	1
wr_tidemark	1	GPV0_BASE + 0x43000 + 0x040	4	2

Table continues on the next page...

Table 13-7. NIC0 ASIB registers (continued)

Register Name	ASIB number	Absolute Address	Reset Value	Comment
wr_tidemark	3	GPV0_BASE + 0x45000 + 0x040	4	1
wr_tidemark	4	GPV0_BASE + 0x46000 + 0x040	4	1
read_qos	0	GPV0_BASE + 0x42000 + 0x100	0	
read_qos	2	GPV0_BASE + 0x44000 + 0x100	6	
read_qos	3	GPV0_BASE + 0x45000 + 0x100	5	
read_qos	4	GPV0_BASE + 0x46000 + 0x100	5	
write_qos	0	GPV0_BASE + 0x42000 + 0x104	0	
write_qos	2	GPV0_BASE + 0x44000 + 0x104	6	
write_qos	3	GPV0_BASE + 0x45000 + 0x104	5	
write_qos	4	GPV0_BASE + 0x46000 + 0x104	5	
fn_mod	0	GPV0_BASE + 0x42000 + 0x108	0	
fn_mod	1	GPV0_BASE + 0x43000 + 0x108	0	
fn_mod	2	GPV0_BASE + 0x44000 + 0x108	0	
fn_mod	3	GPV0_BASE + 0x45000 + 0x108	0	
fn_mod	4	GPV0_BASE + 0x46000 + 0x108	0	

1. The WFIFO depth is 12.
2. The WFIFO depth is 4.

Table 13-8. NIC1 ASIB configuration registers

Register Name	ASIB number	Absolute Address	Reset Value	Notes
fn_mod2	3	GPV1_BASE + 0x45000 + 0x024	0	
fn_mod2	4	GPV1_BASE + 0x46000 + 0x024	0	
fn_mod2	6	GPV1_BASE + 0x48000 + 0x024	0	
fn_mod2	7	GPV1_BASE + 0x49000 + 0x024	0	
fn_mod_ahb	1	GPV1_BASE + 0x43000 + 0x028	0	1
fn_mod_ahb	2	GPV1_BASE + 0x44000 + 0x028	0	
fn_mod_ahb	5	GPV1_BASE + 0x47000 + 0x028	0	1
wr_tidemark	1	GPV1_BASE + 0x43000 + 0x040	4	2
wr_tidemark	2	GPV_BASE + 0x44000 + 0x040	4	2
wr_tidemark	5	GPV1_BASE + 0x47000 + 0x040	4	2
read_qos	1	GPV1_BASE + 0x43000 + 0x100	1	
read_qos	2	GPV1_BASE + 0x44000 + 0x100	1	
read_qos	3	GPV1_BASE + 0x45000 + 0x100	2	
read_qos	4	GPV1_BASE + 0x46000 + 0x100	4	
read_qos	5	GPV1_BASE + 0x47000 + 0x100	4	
read_qos	6	GPV1_BASE + 0x48000 + 0x100	4	
read_qos	7	GPV1_BASE + 0x49000 + 0x100	4	
write_qos	1	GPV1_BASE + 0x43000 + 0x104	1	

Table continues on the next page...

Table 13-8. NIC1 ASIB configuration registers (continued)

Register Name	ASIB number	Absolute Address	Reset Value	Notes
write_qos	2	GPV1_BASE + 0x44000 + 0x104	1	
write_qos	3	GPV1_BASE + 0x45000 + 0x104	2	
write_qos	4	GPV1_BASE + 0x46000 + 0x104	4	
write_qos	5	GPV1_BASE + 0x47000 + 0x104	4	
write_qos	6	GPV1_BASE + 0x48000 + 0x104	4	
write_qos	7	GPV1_BASE + 0x49000 + 0x104	4	
fn_mod	0	GPV1_BASE + 0x42000 + 0x108	0	
fn_mod	1	GPV_BASE + 0x43000 + 0x108	0	
fn_mod	2	GPV1_BASE + 0x44000 + 0x108	0	
fn_mod	3	GPV1_BASE + 0x45000 + 0x108	0	
fn_mod	4	GPV1_BASE + 0x46000 + 0x108	0	
fn_mod	5	GPV1_BASE + 0x47000 + 0x108	0	
fn_mod	6	GPV1_BASE + 0x48000 + 0x108	0	
fn_mod	7	GPV1_BASE + 0x49000 + 0x108	0	

1. The *lock_override* bit is not implemented
2. The WFIFO depth is 4.

13.4.3 Register Descriptions

The NIC-301 registers are dependent upon the selected configuration, the type of ports, hardware-selected features and whether they have a GPV view. The addressing is associated to a specific port, by looking at the port's index number, under "apb_slave" column.

The memory map template is provided in the Configuration programmers model above.

Chapter 14
Direct Memory Access Controller (DMA)

14.1 Chip-specific DMA information

Table 14-1. Reference links to related information

Topic	Related module	Reference
Full description	DMA	DMA
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

14.1.1 DMA

DMA is a second-generation module of the enhanced direct memory access (eDMA2) controller, which is capable of performing complex data transfers with minimal intervention from a host processor. There are two DMA instances in i.MX 7ULP. Each DMA supports 32 DMA channels. The transfer control descriptors for each of the 32 channels are located in system memory. The following table shows the configuration of the DMA.

Table 14-2. DMA configuration

Parameter	Description
Name	Enhanced Direct Memory Access v2 (eDMA2)
Instances	2
Configurable features	• DMA0-1: 32-channels • DMA0-1: 64-bit AHB
Interface speed	NA
External I/O pins	NA

14.2 Introduction

The enhanced direct memory access (eDMA) controller is a second-generation module capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations
  - Data-movement operations
- Local memory containing transfer control descriptors for each of the 32 channels

14.2.1 eDMA system block diagram

Figure 14-1 illustrates the components of the eDMA system, including the eDMA module ("engine").

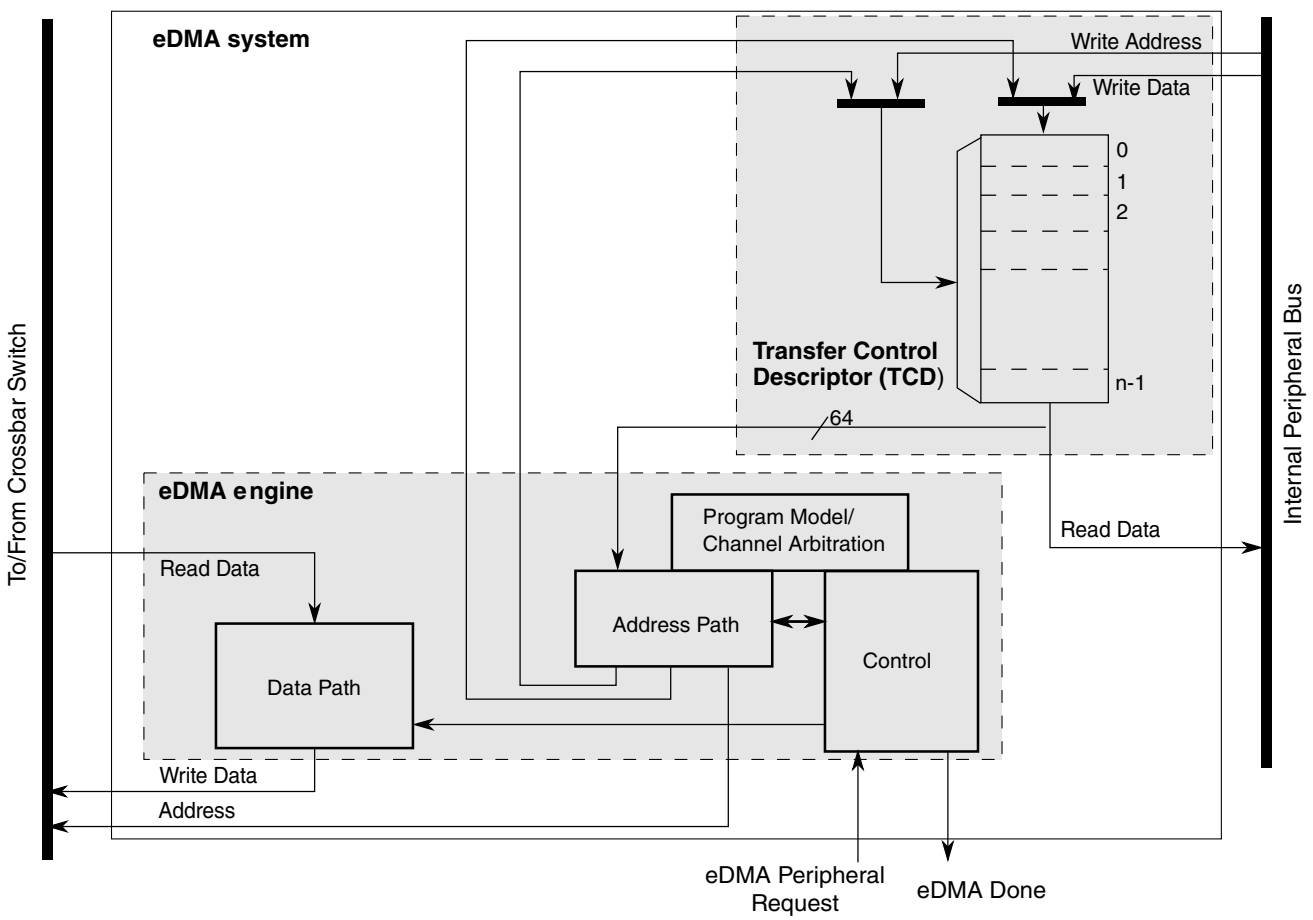


Figure 14-1. eDMA system block diagram

14.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer-control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 14-3. eDMA engine submodules

Submodule	Function
Address path	This block implements registered versions of two channel transfer control descriptors, channel x and channel y, and manages all master bus-address calculations. All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the first channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by DCHPRI*n*[ECP]) where a large data move operation can be preempted to minimize the time another channel is blocked from execution. When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of the next TCD*n* from memory as part of a scatter/gather operation.
Data path	This block implements the bus master read/write datapath. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the 1st stage of the bus pipeline (address phase), while the data path module implements the 2nd stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/ destination write operations until the number of bytes specified in the minor loop byte count has moved. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, two reads are performed, then one 32-bit write.

The transfer-control descriptor local memory is further partitioned into:

Table 14-4. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, managing accesses from the eDMA engine as well as references from the internal peripheral bus. As noted earlier, in the event of

Table continues on the next page...

Table 14-4. Transfer control descriptor memory (continued)

Submodule	Description
	simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

14.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for enhanced addressing modes
- 32-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage to support 16- and 32-byte transfers
  - Connections to the crossbar switch for bus mastering the data movement
- Transfer control descriptor (TCD) organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests

    - One interrupt per channel, which can be asserted at completion of major iteration count
    - Programmable error terminations per channel and logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

14.3 Modes of operation

The eDMA operates in the following modes:

Table 14-5. Modes of operation

Mode	Description
Normal	In Normal mode, the eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with the eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the transfer control descriptor (TCD). The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	DMA operation is configurable in Debug mode via the control register: • If CR[EDBG] is cleared, the DMA continues to operate. • If CR[EDBG] is set, the eDMA stops transferring data. If Debug mode is entered while a channel is active, the eDMA continues operation until the channel retires.
Wait	Before entering Wait mode, the DMA attempts to complete its current transfer. After the transfer completes, the device enters Wait mode.

14.4 Memory map/register definition

The eDMA's programming model is partitioned into two regions:

- The first region defines a number of registers providing control functions
- The second region corresponds to the local transfer control descriptor (TCD) memory

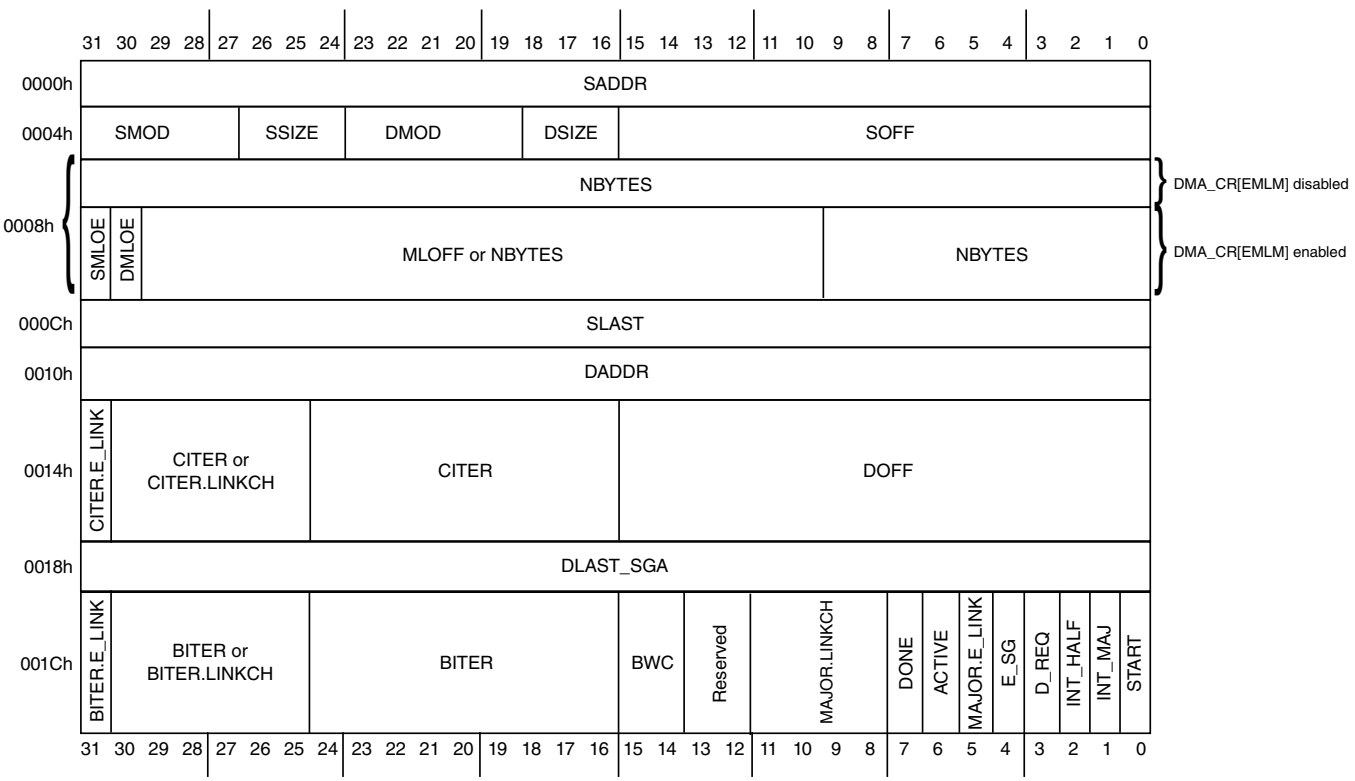
14.4.1 TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the desired data movement operation. The channel descriptors are stored in the local memory in sequential order: channel 0, channel 1, ... channel 31. Each TCD*n* definition is presented as 11 registers of 16 or 32 bits.

14.4.2 TCD initialization

Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

14.4.3 TCD structure



14.4.4 Reserved memory and bit fields

- Reading reserved bits in a register returns the value of zero.
- Writes to reserved bits in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

14.4.5 DMA register descriptions

14.4.5.1 DMA Memory map

DMA0 base address: 4100_8000h

DMA1 base address: 4008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Register (CR)	32	RW	See description.
4h	Error Status Register (ES)	32	RO	0000_0000h
Ch	Enable Request Register (ERQ)	32	RW	0000_0000h
14h	Enable Error Interrupt Register (EEI)	32	RW	0000_0000h
18h	Clear Enable Error Interrupt Register (CEEI)	8	WORZ	00h
19h	Set Enable Error Interrupt Register (SEEI)	8	WORZ	00h
1Ah	Clear Enable Request Register (CERQ)	8	WORZ	00h
1Bh	Set Enable Request Register (SERQ)	8	WORZ	00h
1Ch	Clear DONE Status Bit Register (CDNE)	8	WORZ	00h
1Dh	Set START Bit Register (SSRT)	8	WORZ	00h
1Eh	Clear Error Register (CERR)	8	WORZ	00h
1Fh	Clear Interrupt Request Register (CINT)	8	WORZ	00h
24h	Interrupt Request Register (INT)	32	W1C	0000_0000h
2Ch	Error Register (ERR)	32	W1C	0000_0000h
34h	Hardware Request Status Register (HRS)	32	RO	0000_0000h
44h	Enable Asynchronous Request in Stop Register (EARS)	32	RW	0000_0000h
100h	Channel Priority Register (DCHPRI3)	8	RW	03h
101h	Channel Priority Register (DCHPRI2)	8	RW	02h
102h	Channel Priority Register (DCHPRI1)	8	RW	01h
103h	Channel Priority Register (DCHPRI0)	8	RW	00h
104h	Channel Priority Register (DCHPRI7)	8	RW	07h
105h	Channel Priority Register (DCHPRI6)	8	RW	06h
106h	Channel Priority Register (DCHPRI5)	8	RW	05h
107h	Channel Priority Register (DCHPRI4)	8	RW	04h
108h	Channel Priority Register (DCHPRI11)	8	RW	0Bh
109h	Channel Priority Register (DCHPRI10)	8	RW	0Ah
10Ah	Channel Priority Register (DCHPRI9)	8	RW	09h
10Bh	Channel Priority Register (DCHPRI8)	8	RW	08h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
10Ch	Channel Priority Register (DCHPRI15)	8	RW	0Fh
10Dh	Channel Priority Register (DCHPRI14)	8	RW	0Eh
10Eh	Channel Priority Register (DCHPRI13)	8	RW	0Dh
10Fh	Channel Priority Register (DCHPRI12)	8	RW	0Ch
110h	Channel Priority Register (DCHPRI19)	8	RW	13h
111h	Channel Priority Register (DCHPRI18)	8	RW	12h
112h	Channel Priority Register (DCHPRI17)	8	RW	11h
113h	Channel Priority Register (DCHPRI16)	8	RW	10h
114h	Channel Priority Register (DCHPRI23)	8	RW	17h
115h	Channel Priority Register (DCHPRI22)	8	RW	16h
116h	Channel Priority Register (DCHPRI21)	8	RW	15h
117h	Channel Priority Register (DCHPRI20)	8	RW	14h
118h	Channel Priority Register (DCHPRI27)	8	RW	1Bh
119h	Channel Priority Register (DCHPRI26)	8	RW	1Ah
11Ah	Channel Priority Register (DCHPRI25)	8	RW	19h
11Bh	Channel Priority Register (DCHPRI24)	8	RW	18h
11Ch	Channel Priority Register (DCHPRI31)	8	RW	1Fh
11Dh	Channel Priority Register (DCHPRI30)	8	RW	1Eh
11Eh	Channel Priority Register (DCHPRI29)	8	RW	1Dh
11Fh	Channel Priority Register (DCHPRI28)	8	RW	1Ch
1000h - 13E0h	TCD Source Address (TCD0_SADDR - TCD31_SADDR)	32	RW	See description.
1004h - 13E4h	TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)	16	RW	See description.
1006h - 13E6h	TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)	16	RW	See description.
1008h - 13E8h	TCD Minor Byte Count (Minor Loop Mapping Disabled) (TCD0_NBYTES_MLNO - TCD31_NBYTES_MLNO)	32	RW	See description.
1008h - 13E8h	TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)	32	RW	See description.
1008h - 13E8h	TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)	32	RW	See description.
100Ch - 13ECh	TCD Last Source Address Adjustment (TCD0_SLAST - TCD31_SLAST)	32	RW	See description.
1010h - 13F0h	TCD Destination Address (TCD0_DADDR - TCD31_DADDR)	32	RW	See description.
1014h - 13F4h	TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)	16	RW	See description.
1016h - 13F6h	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)	16	RW	See description.

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1016h - 13F6h	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)	16	RW	See description.
1018h - 13F8h	TCD Last Destination Address Adjustment/Scatter Gather Address (TCD0_DLASTSGA - TCD31_DLASTSGA)	32	RW	See description.
101Ch - 13FCh	TCD Control and Status (TCD0_CSR - TCD31_CSR)	16	RW	See description.
101Eh - 13FEh	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)	16	RW	See description.
101Eh - 13FEh	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)	16	RW	See description.

14.4.5.2 Control Register (CR)

14.4.5.2.1 Offset

Register	Offset
CR	0h

14.4.5.2.2 Function

The CR defines the basic operating configuration of the DMA. The DMA arbitrates channel service requests in two groups of 16 channels each:

- Group 1 contains channels 31-16
- Group 0 contains channels 15-0

Arbitration within a group can be configured to use either a fixed-priority or a round-robin scheme. For fixed-priority arbitration, the highest priority channel requesting service is selected to execute. The channel priority registers assign the priorities; see the DCHPRIn registers. For round-robin arbitration, the channel priorities are ignored and channels within each group are cycled through (from high to low channel number) without regard to priority.

NOTE

For correct operation, writes to the CR register must be performed only when the DMA channels are inactive; that is, when TCDn_CSR[ACTIVE] bits are cleared.

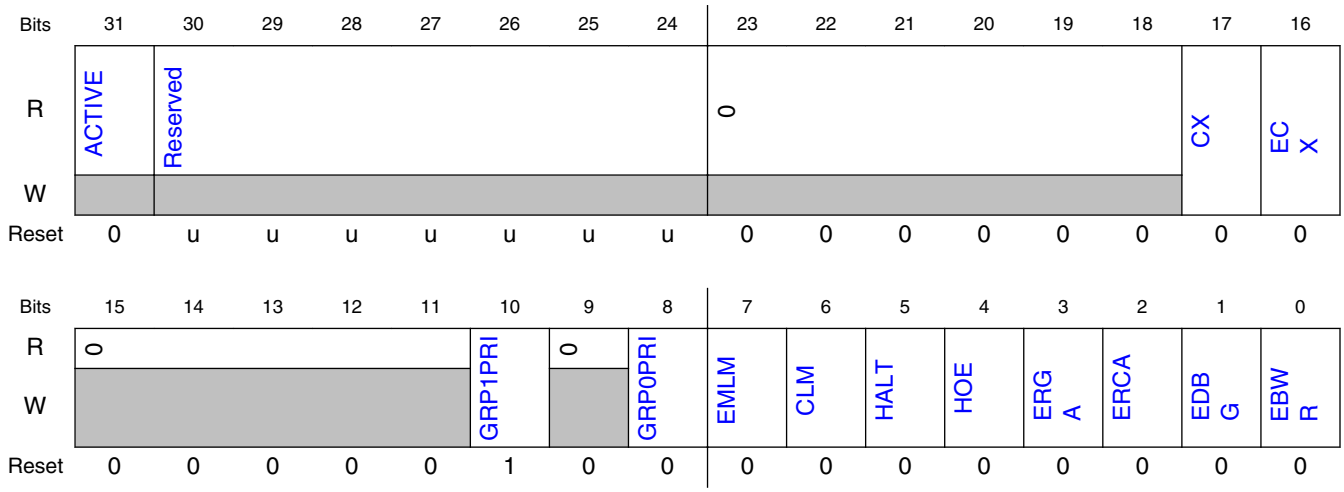
The group priorities operate in a similar fashion. In group fixed priority arbitration mode, channel service requests in the highest priority group are executed first, where priority level 1 is the highest and priority level 0 is the lowest. The group priorities are assigned in the GRPnPRI fields of the DMA Control Register (CR). All group priorities must have unique values prior to any channel service requests occurring; otherwise, a configuration error will be reported. For group round robin arbitration, the group priorities are ignored and the groups are cycled through (from high to low group number) without regard to priority.

Minor loop offsets are address offset values added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. When minor loop offsets are enabled, the minor loop offset (MLOFF) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both prior to the addresses being written back into the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (EMLM is 1), TCDn word2 is redefined. A portion of TCDn word2 is used to specify multiple fields: a source enable bit (SMLOE) to specify the minor loop offset should be applied to the source address (TCDn_SADDR) upon minor loop completion, a destination enable bit (DMLOE) to specify the minor loop offset should be applied to the destination address (TCDn_DADDR) upon minor loop completion, and the sign extended minor loop offset value (MLOFF). The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. When both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

When minor loop mapping is disabled (EMLM is 0), all 32 bits of TCDn word2 are assigned to the NBYTES field.

14.4.5.2.3 Diagram



14.4.5.2.4 Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle. 1b - eDMA is executing a channel.
30-24 —	eDMA version number Reserved
23-18 —	Reserved
17 CX	Cancel Transfer 0b - Normal operation 1b - Cancel the remaining data transfer. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The CX bit clears itself after the cancel has been honored. This cancel retires the channel normally as if the minor loop was completed.
16 ECX	Error Cancel Transfer 0b - Normal operation 1b - Cancel the remaining data transfer in the same fashion as the CX bit. Stop the executing channel and force the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. The ECX bit clears itself after the cancel is honored. In addition to cancelling the transfer, ECX treats the cancel as an error condition, thus updating the Error Status register (DMAx_ES) and generating an optional error interrupt.
15-11 —	Reserved
10 GRP1PRI	Channel Group 1 Priority Group 1 priority level when fixed priority group arbitration is enabled.
9 —	Reserved

Table continues on the next page...

Field	Function
8 GRP0PRI	Channel Group 0 Priority Group 0 priority level when fixed priority group arbitration is enabled.
7 EMLM	Enable Minor Loop Mapping 0b - Disabled. TCDn.word2 is defined as a 32-bit NBYTES field. 1b - Enabled. TCDn.word2 is redefined to include individual enable fields, an offset field, and the NBYTES field. The individual enable fields allow the minor loop offset to be applied to the source address, the destination address, or both. The NBYTES field is reduced when either offset is enabled.
6 CLM	Continuous Link Mode **NOTE:** Do not use continuous link mode with a channel linking to itself if there is only one minor loop iteration per service request, for example, if the channel's NBYTES value is the same as either the source or destination size. The same data transfer profile can be achieved by simply increasing the NBYTES value, which provides more efficient, faster processing. 0b - A minor loop channel link made to itself goes through channel arbitration before being activated again. 1b - A minor loop channel link made to itself does not go through channel arbitration before being activated again. Upon minor loop completion, the channel activates again if that channel has a minor loop channel link enabled and the link channel is itself. This effectively applies the minor loop offsets and restarts the next minor loop.
5 HALT	Halt DMA Operations 0b - Normal operation 1b - Stall the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this bit is cleared.
4 HOE	Halt On Error 0b - Normal operation 1b - Any error causes the HALT bit to set. Subsequently, all service requests are ignored until the HALT bit is cleared.
3 ERGA	Enable Round Robin Group Arbitration 0b - Fixed priority arbitration is used for selection among the groups. 1b - Round robin arbitration is used for selection among the groups.
2 ERCA	Enable Round Robin Channel Arbitration 0b - Fixed priority arbitration is used for channel selection within each group. 1b - Round robin arbitration is used for channel selection within each group.
1 EDBG	Enable Debug 0b - When in debug mode, the DMA continues to operate. 1b - When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. Channel execution resumes when the system exits debug mode or the EDBG bit is cleared.
0 EBWR	Enable Buffered Writes 0b - Buffered writes are disabled. 1b - Buffered writes are enabled.

14.4.5.3 Error Status Register (ES)

14.4.5.3.1 Offset

Register	Offset
ES	4h

14.4.5.3.2 Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- A configuration error, that is:
  - An illegal setting in the transfer-control descriptor, or
  - An illegal priority register setting in fixed-arbitration
- An error termination to a bus master read or write cycle
- A cancel transfer with error bit that will be set when a transfer is canceled via the corresponding cancel transfer control bit

See Fault reporting and handling for more details.

14.4.5.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	0														ECX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPE	CPE	0	ERRCHN					SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.4.5.3.4 Fields

Field	Function
31 VLD	VLD Logical OR of all ERR status bits 0b - No ERR bits are set. 1b - At least one ERR bit is set indicating a valid error exists that has not been cleared.
30-17 —	Reserved

Table continues on the next page...

Field	Function
16 ECX	Transfer Canceled 0b - No canceled transfers 1b - The last recorded entry was a canceled transfer by the error cancel transfer input
15 GPE	Group Priority Error 0b - No group priority error 1b - The last recorded error was a configuration error among the group priorities. All group priorities are not unique.
14 CPE	Channel Priority Error 0b - No channel priority error 1b - The last recorded error was a configuration error in the channel priorities within a group. Channel priorities within a group are not unique.
13 —	Reserved
12-8 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error, excluding GPE and CPE errors, or last recorded error canceled transfer.
7 SAE	Source Address Error 0b - No source address configuration error. 1b - The last recorded error was a configuration error detected in the TCDn_SADDR field. TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE].
6 SOE	Source Offset Error 0b - No source offset configuration error 1b - The last recorded error was a configuration error detected in the TCDn_SOFF field. TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE].
5 DAE	Destination Address Error 0b - No destination address configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DADDR field. TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE].
4 DOE	Destination Offset Error 0b - No destination offset configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DOFF field. TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE].
3 NCE	NBYTES/CITER Configuration Error 0b - No NBYTES/CITER configuration error 1b - The last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields. TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE], orTCDn_CITER[CITER] is equal to zero, orTCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]
2 SGE	Scatter/Gather Configuration Error 0b - No scatter/gather configuration error 1b - The last recorded error was a configuration error detected in the TCDn_DLASTSGA field. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. TCDn_DLASTSGA is not on a 32 byte boundary.
1 SBE	Source Bus Error 0b - No source bus error 1b - The last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - The last recorded error was a bus error on a destination write

14.4.5.4 Enable Request Register (ERQ)

14.4.5.4.1 Offset

Register	Offset
ERQ	Ch

14.4.5.4.2 Function

The ERQ register provides a bit map for the 32 channels to enable the request signal for each channel. The state of any given channel enable is directly affected by writes to this register; it is also affected by writes to the SERQ and CERQ registers. These registers are provided so the request enable for a single channel can easily be modified without needing to perform a read-modify-write sequence to this register.

DMA request input signals and this enable request flag must be asserted before a channel's hardware service request is accepted. The state of the DMA enable request flag does not affect a channel service request made explicitly through software or a linked channel request.

NOTE

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

14.4.5.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ERQ31	ERQ30	ERQ29	ERQ28	ERQ27	ERQ26	ERQ25	ERQ24	ERQ23	ERQ22	ERQ21	ERQ20	ERQ19	ERQ18	ERQ17	ERQ16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ERQ15	ERQ14	ERQ13	ERQ12	ERQ11	ERQ10	ERQ9	ERQ8	ERQ7	ERQ6	ERQ5	ERQ4	ERQ3	ERQ2	ERQ1	ERQ0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.4.5.4.4 Fields

Field	Function
31 ERQ31	Enable DMA Request 31 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
30 ERQ30	Enable DMA Request 30 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
29 ERQ29	Enable DMA Request 29 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
28 ERQ28	Enable DMA Request 28 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
27 ERQ27	Enable DMA Request 27 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
26 ERQ26	Enable DMA Request 26 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
25 ERQ25	Enable DMA Request 25 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
24 ERQ24	Enable DMA Request 24 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
23 ERQ23	Enable DMA Request 23 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
22 ERQ22	Enable DMA Request 22 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
21 ERQ21	Enable DMA Request 21 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
20 ERQ20	Enable DMA Request 20 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
19 ERQ19	Enable DMA Request 19 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
18 ERQ18	Enable DMA Request 18 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
17 ERQ17	Enable DMA Request 17 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
16 ERQ16	Enable DMA Request 16 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled

Table continues on the next page...

Field	Function
15 ERQ15	Enable DMA Request 15 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
14 ERQ14	Enable DMA Request 14 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
13 ERQ13	Enable DMA Request 13 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
12 ERQ12	Enable DMA Request 12 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
11 ERQ11	Enable DMA Request 11 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
10 ERQ10	Enable DMA Request 10 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
9 ERQ9	Enable DMA Request 9 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
8 ERQ8	Enable DMA Request 8 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
7 ERQ7	Enable DMA Request 7 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
6 ERQ6	Enable DMA Request 6 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
5 ERQ5	Enable DMA Request 5 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
4 ERQ4	Enable DMA Request 4 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
3 ERQ3	Enable DMA Request 3 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
2 ERQ2	Enable DMA Request 2 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
1 ERQ1	Enable DMA Request 1 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled
0 ERQ0	Enable DMA Request 0 0b - The DMA request signal for the corresponding channel is disabled 1b - The DMA request signal for the corresponding channel is enabled

14.4.5.5 Enable Error Interrupt Register (EEI)

14.4.5.5.1 Offset

Register	Offset
EEI	14h

14.4.5.5.2 Function

The EEI register provides a bit map for the 32 channels to enable the error interrupt signal for each channel. The state of any given channel's error interrupt enable is directly affected by writes to this register; it is also affected by writes to the SEEI and CEEI. These registers are provided so that the error interrupt enable for a single channel can easily be modified without the need to perform a read-modify-write sequence to the EEI register.

The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller.

14.4.5.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EEI31	EEI30	EEI29	EEI28	EEI27	EEI26	EEI25	EEI24	EEI23	EEI22	EEI21	EEI20	EEI19	EEI18	EEI17	EEI16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EEI15	EEI14	EEI13	EEI12	EEI11	EEI10	EEI 9	EEI 8	EEI 7	EEI 6	EEI 5	EEI 4	EEI 3	EEI 2	EEI 1	EEI 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.4.5.5.4 Fields

Field	Function
31 EEI31	Enable Error Interrupt 31 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
30	Enable Error Interrupt 30

Table continues on the next page...

Field	Function
EEI30	0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
29 EEI29	Enable Error Interrupt 29 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
28 EEI28	Enable Error Interrupt 28 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
27 EEI27	Enable Error Interrupt 27 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
26 EEI26	Enable Error Interrupt 26 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
25 EEI25	Enable Error Interrupt 25 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
24 EEI24	Enable Error Interrupt 24 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
23 EEI23	Enable Error Interrupt 23 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
22 EEI22	Enable Error Interrupt 22 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
21 EEI21	Enable Error Interrupt 21 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
20 EEI20	Enable Error Interrupt 20 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
19 EEI19	Enable Error Interrupt 19 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
18 EEI18	Enable Error Interrupt 18 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
17 EEI17	Enable Error Interrupt 17 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
16 EEI16	Enable Error Interrupt 16 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
15 EEI15	Enable Error Interrupt 15 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
14 EEI14	Enable Error Interrupt 14 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request

Table continues on the next page...

Field	Function
13 EEI13	Enable Error Interrupt 13 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
12 EEI12	Enable Error Interrupt 12 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
11 EEI11	Enable Error Interrupt 11 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
10 EEI10	Enable Error Interrupt 10 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
9 EEI9	Enable Error Interrupt 9 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
8 EEI8	Enable Error Interrupt 8 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
7 EEI7	Enable Error Interrupt 7 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
6 EEI6	Enable Error Interrupt 6 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
5 EEI5	Enable Error Interrupt 5 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
4 EEI4	Enable Error Interrupt 4 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
3 EEI3	Enable Error Interrupt 3 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
2 EEI2	Enable Error Interrupt 2 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
1 EEI1	Enable Error Interrupt 1 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request
0 EEI0	Enable Error Interrupt 0 0b - The error signal for corresponding channel does not generate an error interrupt 1b - The assertion of the error signal for corresponding channel generates an error interrupt request

14.4.5.6 Clear Enable Error Interrupt Register (CEEI)

14.4.5.6.1 Offset

Register	Offset
CEEI	18h

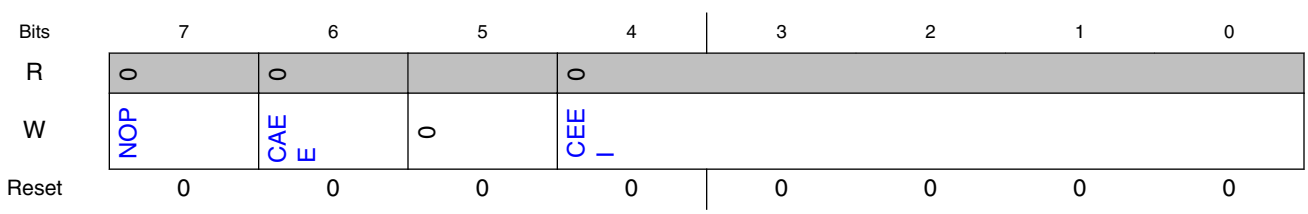
14.4.5.6.2 Function

The CEEI provides a simple memory-mapped mechanism to clear a given bit in the EEI to disable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be cleared. Setting the CAEE bit provides a global clear function, forcing the EEI contents to be cleared, disabling all DMA request inputs.

If the NOP bit is set, the command is ignored. This allows you to set a single, byte-wide register with a 32-bit write while not affecting the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP bit set so that that these register will not be affected by the write.

Reads of this register return all zeroes.

14.4.5.6.3 Diagram



14.4.5.6.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 CAEE	Clear All Enable Error Interrupts 0b - Clear only the EEI bit specified in the CEEI field 1b - Clear all bits in EEI
5 —	Reserved
4-0 CEEI	Clear Enable Error Interrupt Clears the corresponding bit in EEI

14.4.5.7 Set Enable Error Interrupt Register (SEEI)

14.4.5.7.1 Offset

Register	Offset
SEEI	19h

14.4.5.7.2 Function

The SEEI provides a simple memory-mapped mechanism to set a given bit in the EEI to enable the error interrupt for a given channel. The data value on a register write causes the corresponding bit in the EEI to be set. Setting the SAEE bit provides a global set function, forcing the entire EEI contents to be set.

If the NOP bit is set, the command is ignored. This allows you to set a single, byte-wide register with a 32-bit write while not affecting the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP bit set so that that these register will not be affected by the write.

Reads of this register return all zeroes.

14.4.5.7.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0		0				
W	NOP	SAEE	0	SEEI				
Reset	0	0	0	0	0	0	0	0

14.4.5.7.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 SAEE	Sets All Enable Error Interrupts 0b - Set only the EEI bit specified in the SEEI field. 1b - Sets all bits in EEI
5	Reserved

Table continues on the next page...

Field	Function
—	
4-0 SEEI	Set Enable Error Interrupt Sets the corresponding bit in EEI

14.4.5.8 Clear Enable Request Register (CERQ)

14.4.5.8.1 Offset

Register	Offset
CERQ	1Ah

14.4.5.8.2 Function

The CERQ provides a simple memory-mapped mechanism to clear a given bit in the ERQ to disable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be cleared. Setting the CAER bit provides a global clear function, forcing the entire contents of the ERQ to be cleared, disabling all DMA request inputs.

If the NOP bit is set, the command is ignored. This allows you to set a single, byte-wide register with a 32-bit write while not affecting the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP bit set so that that these register will not be affected by the write.

Reads of this register return all zeroes.

NOTE

Disable a channel's hardware service request at the source before clearing the channel's ERQ bit.

14.4.5.8.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0		0				
W	NOP	CAER	0	CERQ				
Reset	0	0	0	0	0	0	0	0

14.4.5.8.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 CAER	Clear All Enable Requests 0b - Clear only the ERQ bit specified in the CERQ field 1b - Clear all bits in ERQ
5 —	Reserved
4-0 CERQ	Clear Enable Request Clears the corresponding bit in ERQ.

14.4.5.9 Set Enable Request Register (SERQ)

14.4.5.9.1 Offset

Register	Offset
SERQ	1Bh

14.4.5.9.2 Function

The SERQ provides a simple memory-mapped mechanism to set a given bit in the ERQ to enable the DMA request for a given channel. The data value on a register write causes the corresponding bit in the ERQ to be set. Setting the SAER bit provides a global set function, forcing the entire contents of ERQ to be set.

If the NOP bit is set, the command is ignored. This allows you to set a single, byte-wide register with a 32-bit write while not affecting the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP bit set so that that these register will not be affected by the write.

Reads of this register return all zeroes.

14.4.5.9.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0		0				
W	NOP	SAER	0	SERQ				
Reset	0	0	0	0	0	0	0	0

14.4.5.9.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 SAER	Set All Enable Requests 0b - Set only the ERQ bit specified in the SERQ field 1b - Set all bits in ERQ
5 —	Reserved
4-0 SERQ	Set Enable Request Sets the corresponding bit in ERQ.

14.4.5.10 Clear DONE Status Bit Register (CDNE)

14.4.5.10.1 Offset

Register	Offset
CDNE	1Ch

14.4.5.10.2 Function

The CDNE provides a simple memory-mapped mechanism to clear the DONE bit in the TCD of the given channel. The data value on a register write causes the DONE bit in the corresponding transfer control descriptor to be cleared. Setting the CADN bit provides a global clear function, forcing all DONE bits to be cleared.

If the NOP bit is set, the command is ignored. This allows you to set a single, byte-wide register with a 32-bit write while not affecting the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP bit set so that that these register will not be affected by the write.

Reads of this register return all zeroes.

14.4.5.10.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0		0				
W	NOP	CADN	0	CDNE				
Reset	0	0	0	0	0	0	0	0

14.4.5.10.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 CADN	Clears All DONE Bits 0b - Clears only the TCDn_CSR[DONE] bit specified in the CDNE field 1b - Clears all bits in TCDn_CSR[DONE]
5 —	Reserved
4-0 CDNE	Clear DONE Bit Clears the corresponding bit in TCDn_CSR[DONE]

14.4.5.11 Set START Bit Register (SSRT)

14.4.5.11.1 Offset

Register	Offset
SSRT	1Dh

14.4.5.11.2 Function

The SSRT provides a simple memory-mapped mechanism to set the START bit in the TCD of the given channel. The data value on a register write causes the START bit in the corresponding transfer control descriptor to be set. Setting the SAST bit provides a global set function, forcing all START bits to be set.

If the NOP bit is set, the command is ignored. This allows you to set a single, byte-wide register with a 32-bit write while not affecting the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP bit set so that that these register will not be affected by the write.

Reads of this register return all zeroes.

14.4.5.11.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0		0				
W	NOP	SAST	0	SSRT				
Reset	0	0	0	0	0	0	0	0

14.4.5.11.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 SAST	Set All START Bits (activates all channels) 0b - Set only the TCDn_CSR[START] bit specified in the SSRT field 1b - Set all bits in TCDn_CSR[START]
5 —	Reserved
4-0 SSRT	Set START Bit Sets the corresponding bit in TCDn_CSR[START]

14.4.5.12 Clear Error Register (CERR)

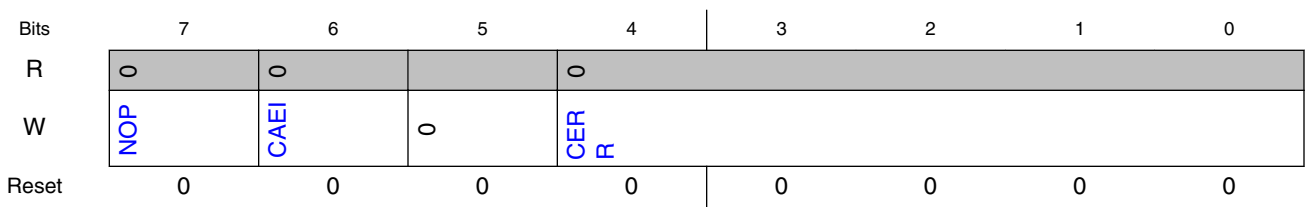
14.4.5.12.1 Offset

Register	Offset
CERR	1Eh

14.4.5.12.2 Function

The CERR provides a simple memory-mapped mechanism to clear a given bit in the ERR to disable the error condition flag for a given channel. The given value on a register write causes the corresponding bit in the ERR to be cleared. Setting the CAEI bit provides a global clear function, forcing the ERR contents to be cleared, clearing all channel error indicators. If the NOP bit is set, the command is ignored. This allows you to write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

14.4.5.12.3 Diagram



14.4.5.12.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 CAEI	Clear All Error Indicators 0b - Clear only the ERR bit specified in the CERR field 1b - Clear all bits in ERR
5 —	Reserved
4-0 CERR	Clear Error Indicator Clears the corresponding bit in ERR

14.4.5.13 Clear Interrupt Request Register (CINT)

14.4.5.13.1 Offset

Register	Offset
CINT	1Fh

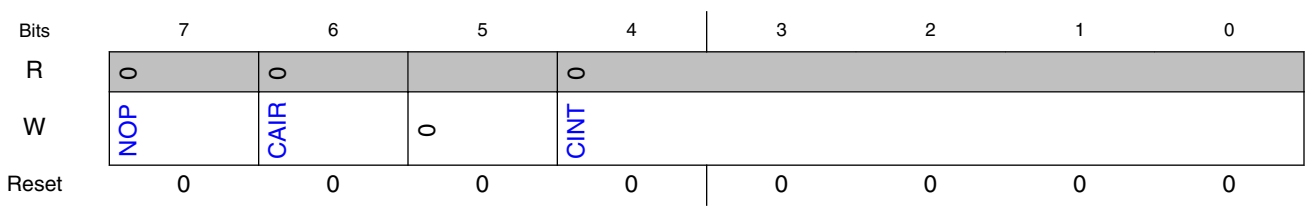
14.4.5.13.2 Function

The CINT provides a simple, memory-mapped mechanism to clear a given bit in the INT to disable the interrupt request for a given channel. The given value on a register write causes the corresponding bit in the INT to be cleared. Setting the CAIR bit provides a global clear function, forcing the entire contents of the INT to be cleared, disabling all DMA interrupt requests.

If the NOP bit is set, the command is ignored. This allows you to set a single, byte-wide register with a 32-bit write while not affecting the other registers addressed in the write. In such a case the other three bytes of the word would all have their NOP bit set so that that these register will not be affected by the write.

Reads of this register return all zeroes.

14.4.5.13.3 Diagram



14.4.5.13.4 Fields

Field	Function
7 NOP	No Op enable 0b - Normal operation 1b - No operation, ignore the other bits in this register
6 CAIR	Clear All Interrupt Requests 0b - Clear only the INT bit specified in the CINT field 1b - Clear all bits in INT
5 —	Reserved
4-0 CINT	Clear Interrupt Request Clears the corresponding bit in INT

14.4.5.14 Interrupt Request Register (INT)

14.4.5.14.1 Offset

Register	Offset
INT	24h

14.4.5.14.2 Function

The INT register provides a bit map for the 32 channels signaling the presence of an interrupt request for each channel. Depending on the appropriate bit setting in the transfer-control descriptors, the eDMA engine generates an interrupt on data transfer completion. The outputs of this register are directly routed to the interrupt controller. During the interrupt-service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. Typically, a write to the CINT register in the interrupt service routine is used for this purpose.

The state of any given channel's interrupt request is directly affected by writes to this register; it is also affected by writes to the CINT register. On writes to INT, a 1 in any bit position clears the corresponding channel's interrupt request. A zero in any bit position has no affect on the corresponding channel's current interrupt status. The CINT register is provided so the interrupt request for a single channel can easily be cleared without the need to perform a read-modify-write sequence to the INT register.

14.4.5.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INT31	INT30	INT29	INT28	INT27	INT26	INT25	INT24	INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.4.5.14.4 Fields

Field	Function
31 INT31	Interrupt Request 31 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
30 INT30	Interrupt Request 30 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
29 INT29	Interrupt Request 29 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
28 INT28	Interrupt Request 28 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
27 INT27	Interrupt Request 27 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
26 INT26	Interrupt Request 26 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
25 INT25	Interrupt Request 25 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
24 INT24	Interrupt Request 24 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
23 INT23	Interrupt Request 23 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
22 INT22	Interrupt Request 22 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
21 INT21	Interrupt Request 21 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
20 INT20	Interrupt Request 20 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
19 INT19	Interrupt Request 19 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
18 INT18	Interrupt Request 18 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
17 INT17	Interrupt Request 17 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
16	Interrupt Request 16 0b - The interrupt request for corresponding channel is cleared

Table continues on the next page...

Field	Function
INT16	1b - The interrupt request for corresponding channel is active
15 INT15	Interrupt Request 15 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
14 INT14	Interrupt Request 14 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
13 INT13	Interrupt Request 13 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
12 INT12	Interrupt Request 12 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
11 INT11	Interrupt Request 11 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
10 INT10	Interrupt Request 10 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
9 INT9	Interrupt Request 9 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
8 INT8	Interrupt Request 8 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
7 INT7	Interrupt Request 7 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
6 INT6	Interrupt Request 6 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
5 INT5	Interrupt Request 5 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
4 INT4	Interrupt Request 4 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
3 INT3	Interrupt Request 3 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
2 INT2	Interrupt Request 2 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
1 INT1	Interrupt Request 1 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active
0 INT0	Interrupt Request 0 0b - The interrupt request for corresponding channel is cleared 1b - The interrupt request for corresponding channel is active

14.4.5.15 Error Register (ERR)

14.4.5.15.1 Offset

Register	Offset
ERR	2Ch

14.4.5.15.2 Function

The ERR register provides a bit map for the 32 channels, signaling the presence of an error for each channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate bit in this register. The outputs of this register are enabled by the contents of the EEI register, then logically summed across groups of 16 and 32 channels to form several group error interrupt requests, which are then routed to the interrupt controller. During the execution of the interrupt-service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. Typically, a write to the CERR in the interrupt-service routine is used for this purpose. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when an error is detected.

The contents of this register can also be polled because a non-zero value indicates the presence of a channel error regardless of the state of the EEI fields. The state of any given channel's error indicators is affected by writes to this register; it is also affected by writes to the CERR. On writes to the ERR, a one in any bit position clears the corresponding channel's error status. A zero in any bit position has no affect on the corresponding channel's current error status. The CERR is provided so the error indicator for a single channel can easily be cleared.

14.4.5.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERR31	ERR30	ERR29	ERR28	ERR27	ERR26	ERR25	ERR24	ERR23	ERR22	ERR21	ERR20	ERR19	ERR18	ERR17	ERR16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR15	ERR14	ERR13	ERR12	ERR11	ERR10	ERR9	ERR8	ERR7	ERR6	ERR5	ERR4	ERR3	ERR2	ERR1	ERR0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.4.5.15.4 Fields

Field	Function
31 ERR31	Error In Channel 31 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
30 ERR30	Error In Channel 30 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
29 ERR29	Error In Channel 29 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
28 ERR28	Error In Channel 28 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
27 ERR27	Error In Channel 27 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
26 ERR26	Error In Channel 26 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
25 ERR25	Error In Channel 25 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
24 ERR24	Error In Channel 24 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
23 ERR23	Error In Channel 23 0b - An error in this channel has not occurred 1b - An error in this channel has occurred

Table continues on the next page...

Field	Function
22 ERR22	Error In Channel 22 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
21 ERR21	Error In Channel 21 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
20 ERR20	Error In Channel 20 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
19 ERR19	Error In Channel 19 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
18 ERR18	Error In Channel 18 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
17 ERR17	Error In Channel 17 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
16 ERR16	Error In Channel 16 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
15 ERR15	Error In Channel 15 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
14 ERR14	Error In Channel 14 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
13 ERR13	Error In Channel 13 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
12 ERR12	Error In Channel 12 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
11 ERR11	Error In Channel 11 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
10 ERR10	Error In Channel 10 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
9 ERR9	Error In Channel 9 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
8 ERR8	Error In Channel 8 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
7 ERR7	Error In Channel 7 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
6 ERR6	Error In Channel 6 0b - An error in this channel has not occurred

Table continues on the next page...

Field	Function
	1b - An error in this channel has occurred
5 ERR5	Error In Channel 5 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
4 ERR4	Error In Channel 4 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
3 ERR3	Error In Channel 3 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
2 ERR2	Error In Channel 2 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
1 ERR1	Error In Channel 1 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
0 ERR0	Error In Channel 0 0b - An error in this channel has not occurred 1b - An error in this channel has occurred

14.4.5.16 Hardware Request Status Register (HRS)

14.4.5.16.1 Offset

Register	Offset
HRS	34h

14.4.5.16.2 Function

The HRS register provides a bit map for the DMA channels, signaling the presence of a hardware request for each channel. The hardware request status bits reflect the current state of the register and qualified (via the ERQ fields) DMA request signals as seen by the DMA's arbitration logic. This view into the hardware request signals may be used for debug purposes.

NOTE

These bits reflect the state of the request as seen by the arbitration logic. Therefore, this status is affected by the ERQ bits.

14.4.5.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HRS31	HRS30	HRS29	HRS28	HRS27	HRS26	HRS25	HRS24	HRS23	HRS22	HRS21	HRS20	HRS19	HRS18	HRS17	HRS16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HRS15	HRS14	HRS13	HRS12	HRS11	HRS10	HRS 9	HRS 8	HRS 7	HRS 6	HRS 5	HRS 4	HRS 3	HRS 2	HRS 1	HRS 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.4.5.16.4 Fields

Field	Function
31 HRS31	Hardware Request Status Channel 31 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 31 is not present 1b - A hardware service request for channel 31 is present
30 HRS30	Hardware Request Status Channel 30 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 30 is not present 1b - A hardware service request for channel 30 is present
29 HRS29	Hardware Request Status Channel 29 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 29 is not preset 1b - A hardware service request for channel 29 is present
28 HRS28	Hardware Request Status Channel 28 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 28 is not present 1b - A hardware service request for channel 28 is present
27 HRS27	Hardware Request Status Channel 27 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 27 is not present

Table continues on the next page...

Field	Function
	1b - A hardware service request for channel 27 is present
26 HRS26	Hardware Request Status Channel 26 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 26 is not present 1b - A hardware service request for channel 26 is present
25 HRS25	Hardware Request Status Channel 25 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 25 is not present 1b - A hardware service request for channel 25 is present
24 HRS24	Hardware Request Status Channel 24 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 24 is not present 1b - A hardware service request for channel 24 is present
23 HRS23	Hardware Request Status Channel 23 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 23 is not present 1b - A hardware service request for channel 23 is present
22 HRS22	Hardware Request Status Channel 22 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 22 is not present 1b - A hardware service request for channel 22 is present
21 HRS21	Hardware Request Status Channel 21 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 21 is not present 1b - A hardware service request for channel 21 is present
20 HRS20	Hardware Request Status Channel 20 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 20 is not present 1b - A hardware service request for channel 20 is present
19 HRS19	Hardware Request Status Channel 19 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 19 is not present 1b - A hardware service request for channel 19 is present
18	Hardware Request Status Channel 18

Table continues on the next page...

Field	Function
HRS18	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 18 is not present 1b - A hardware service request for channel 18 is present
17	Hardware Request Status Channel 17
HRS17	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 17 is not present 1b - A hardware service request for channel 17 is present
16	Hardware Request Status Channel 16
HRS16	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 16 is not present 1b - A hardware service request for channel 16 is present
15	Hardware Request Status Channel 15
HRS15	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 15 is not present 1b - A hardware service request for channel 15 is present
14	Hardware Request Status Channel 14
HRS14	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 14 is not present 1b - A hardware service request for channel 14 is present
13	Hardware Request Status Channel 13
HRS13	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 13 is not present 1b - A hardware service request for channel 13 is present
12	Hardware Request Status Channel 12
HRS12	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 12 is not present 1b - A hardware service request for channel 12 is present
11	Hardware Request Status Channel 11
HRS11	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 11 is not present 1b - A hardware service request for channel 11 is present
10	Hardware Request Status Channel 10
HRS10	

Table continues on the next page...

Field	Function
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 10 is not present 1b - A hardware service request for channel 10 is present
9 HRS9	Hardware Request Status Channel 9 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 9 is not present 1b - A hardware service request for channel 9 is present
8 HRS8	Hardware Request Status Channel 8 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 8 is not present 1b - A hardware service request for channel 8 is present
7 HRS7	Hardware Request Status Channel 7 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 7 is not present 1b - A hardware service request for channel 7 is present
6 HRS6	Hardware Request Status Channel 6 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 6 is not present 1b - A hardware service request for channel 6 is present
5 HRS5	Hardware Request Status Channel 5 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 5 is not present 1b - A hardware service request for channel 5 is present
4 HRS4	Hardware Request Status Channel 4 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 4 is not present 1b - A hardware service request for channel 4 is present
3 HRS3	Hardware Request Status Channel 3 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 3 is not present 1b - A hardware service request for channel 3 is present
2 HRS2	Hardware Request Status Channel 2

Table continues on the next page...

Field	Function
	The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 2 is not present 1b - A hardware service request for channel 2 is present
1 HRS1	Hardware Request Status Channel 1 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 1 is not present 1b - A hardware service request for channel 1 is present
0 HRS0	Hardware Request Status Channel 0 The HRS bit for its respective channel remains asserted for the period when a Hardware Request is Present on the Channel. After the Request is completed and Channel is free, the HRS bit is automatically cleared by hardware. 0b - A hardware service request for channel 0 is not present 1b - A hardware service request for channel 0 is present

14.4.5.17 Enable Asynchronous Request in Stop Register (EARS)

14.4.5.17.1 Offset

Register	Offset
EARS	44h

14.4.5.17.2 Function

The EARS register is used to enable or disable the DMA requests in Enable Request Register (ERQ) by AND'ing the bits of these two registers.

14.4.5.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EDREQ_31	EDREQ_30	EDREQ_29	EDREQ_28	EDREQ_27	EDREQ_26	EDREQ_25	EDREQ_24	EDREQ_23	EDREQ_22	EDREQ_21	EDREQ_20	EDREQ_19	EDREQ_18	EDREQ_17	EDREQ_16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDREQ_15	EDREQ_14	EDREQ_13	EDREQ_12	EDREQ_11	EDREQ_10	EDREQ_9	EDREQ_8	EDREQ_7	EDREQ_6	EDREQ_5	EDREQ_4	EDREQ_3	EDREQ_2	EDREQ_1	EDREQ_0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14.4.5.17.4 Fields

Field	Function
31 EDREQ_31	Enable asynchronous DMA request in stop mode for channel 31 0b - Disable asynchronous DMA request for channel 31 1b - Enable asynchronous DMA request for channel 31
30 EDREQ_30	Enable asynchronous DMA request in stop mode for channel 30 0b - Disable asynchronous DMA request for channel 30 1b - Enable asynchronous DMA request for channel 30
29 EDREQ_29	Enable asynchronous DMA request in stop mode for channel 29 0b - Disable asynchronous DMA request for channel 29 1b - Enable asynchronous DMA request for channel 29
28 EDREQ_28	Enable asynchronous DMA request in stop mode for channel 28 0b - Disable asynchronous DMA request for channel 28 1b - Enable asynchronous DMA request for channel 28
27 EDREQ_27	Enable asynchronous DMA request in stop mode for channel 27 0b - Disable asynchronous DMA request for channel 27 1b - Enable asynchronous DMA request for channel 27
26 EDREQ_26	Enable asynchronous DMA request in stop mode for channel 26 0b - Disable asynchronous DMA request for channel 26 1b - Enable asynchronous DMA request for channel 26
25 EDREQ_25	Enable asynchronous DMA request in stop mode for channel 25 0b - Disable asynchronous DMA request for channel 25 1b - Enable asynchronous DMA request for channel 25
24 EDREQ_24	Enable asynchronous DMA request in stop mode for channel 24 0b - Disable asynchronous DMA request for channel 24 1b - Enable asynchronous DMA request for channel 24
23 EDREQ_23	Enable asynchronous DMA request in stop mode for channel 23 0b - Disable asynchronous DMA request for channel 23 1b - Enable asynchronous DMA request for channel 23
22	Enable asynchronous DMA request in stop mode for channel 22 0b - Disable asynchronous DMA request for channel 22

Table continues on the next page...

Field	Function
EDREQ_22	1b - Enable asynchronous DMA request for channel 22
21 EDREQ_21	Enable asynchronous DMA request in stop mode for channel 21 0b - Disable asynchronous DMA request for channel 21 1b - Enable asynchronous DMA request for channel 21
20 EDREQ_20	Enable asynchronous DMA request in stop mode for channel 20 0b - Disable asynchronous DMA request for channel 20 1b - Enable asynchronous DMA request for channel 20
19 EDREQ_19	Enable asynchronous DMA request in stop mode for channel 19 0b - Disable asynchronous DMA request for channel 19 1b - Enable asynchronous DMA request for channel 19
18 EDREQ_18	Enable asynchronous DMA request in stop mode for channel 18 0b - Disable asynchronous DMA request for channel 18 1b - Enable asynchronous DMA request for channel 18
17 EDREQ_17	Enable asynchronous DMA request in stop mode for channel 17 0b - Disable asynchronous DMA request for channel 17 1b - Enable asynchronous DMA request for channel 17
16 EDREQ_16	Enable asynchronous DMA request in stop mode for channel 16 0b - Disable asynchronous DMA request for channel 16 1b - Enable asynchronous DMA request for channel 16
15 EDREQ_15	Enable asynchronous DMA request in stop mode for channel 15 0b - Disable asynchronous DMA request for channel 15. 1b - Enable asynchronous DMA request for channel 15.
14 EDREQ_14	Enable asynchronous DMA request in stop mode for channel 14 0b - Disable asynchronous DMA request for channel 14. 1b - Enable asynchronous DMA request for channel 14.
13 EDREQ_13	Enable asynchronous DMA request in stop mode for channel 13 0b - Disable asynchronous DMA request for channel 13. 1b - Enable asynchronous DMA request for channel 13.
12 EDREQ_12	Enable asynchronous DMA request in stop mode for channel 12 0b - Disable asynchronous DMA request for channel 12. 1b - Enable asynchronous DMA request for channel 12.
11 EDREQ_11	Enable asynchronous DMA request in stop mode for channel 11 0b - Disable asynchronous DMA request for channel 11. 1b - Enable asynchronous DMA request for channel 11.
10 EDREQ_10	Enable asynchronous DMA request in stop mode for channel 10 0b - Disable asynchronous DMA request for channel 10. 1b - Enable asynchronous DMA request for channel 10.
9 EDREQ_9	Enable asynchronous DMA request in stop mode for channel 9 0b - Disable asynchronous DMA request for channel 9. 1b - Enable asynchronous DMA request for channel 9.
8 EDREQ_8	Enable asynchronous DMA request in stop mode for channel 8 0b - Disable asynchronous DMA request for channel 8. 1b - Enable asynchronous DMA request for channel 8.
7 EDREQ_7	Enable asynchronous DMA request in stop mode for channel 7 0b - Disable asynchronous DMA request for channel 7. 1b - Enable asynchronous DMA request for channel 7.
6 EDREQ_6	Enable asynchronous DMA request in stop mode for channel 6 0b - Disable asynchronous DMA request for channel 6. 1b - Enable asynchronous DMA request for channel 6.
5	Enable asynchronous DMA request in stop mode for channel 5

Table continues on the next page...

Field	Function
EDREQ_5	0b - Disable asynchronous DMA request for channel 5. 1b - Enable asynchronous DMA request for channel 5.
4 EDREQ_4	Enable asynchronous DMA request in stop mode for channel 4 0b - Disable asynchronous DMA request for channel 4. 1b - Enable asynchronous DMA request for channel 4.
3 EDREQ_3	Enable asynchronous DMA request in stop mode for channel 3. 0b - Disable asynchronous DMA request for channel 3. 1b - Enable asynchronous DMA request for channel 3.
2 EDREQ_2	Enable asynchronous DMA request in stop mode for channel 2. 0b - Disable asynchronous DMA request for channel 2. 1b - Enable asynchronous DMA request for channel 2.
1 EDREQ_1	Enable asynchronous DMA request in stop mode for channel 1. 0b - Disable asynchronous DMA request for channel 1 1b - Enable asynchronous DMA request for channel 1.
0 EDREQ_0	Enable asynchronous DMA request in stop mode for channel 0. 0b - Disable asynchronous DMA request for channel 0. 1b - Enable asynchronous DMA request for channel 0.

14.4.5.18 Channel Priority Register (DCHPRI0 - DCHPRI31)

14.4.5.18.1 Offset

Register	Offset
DCHPRI3	100h
DCHPRI2	101h
DCHPRI1	102h
DCHPRI0	103h
DCHPRI7	104h
DCHPRI6	105h
DCHPRI5	106h
DCHPRI4	107h
DCHPRI11	108h
DCHPRI10	109h
DCHPRI9	10Ah
DCHPRI8	10Bh
DCHPRI15	10Ch
DCHPRI14	10Dh
DCHPRI13	10Eh
DCHPRI12	10Fh
DCHPRI19	110h
DCHPRI18	111h

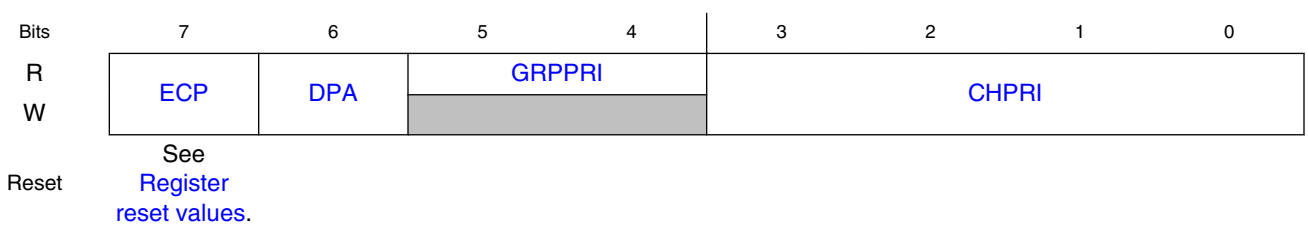
Table continues on the next page...

Register	Offset
DCHPRI17	112h
DCHPRI16	113h
DCHPRI23	114h
DCHPRI22	115h
DCHPRI21	116h
DCHPRI20	117h
DCHPRI27	118h
DCHPRI26	119h
DCHPRI25	11Ah
DCHPRI24	11Bh
DCHPRI31	11Ch
DCHPRI30	11Dh
DCHPRI29	11Eh
DCHPRI28	11Fh

14.4.5.18.2 Function

When fixed-priority channel arbitration is enabled (CR[ERCA] = 0), the contents of these registers define the unique priorities associated with each channel within a group. The channel priorities are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, etc. Software must program the channel priorities with unique values; otherwise, a configuration error is reported. The range of the priority value is limited to the values of 0 through 15. When read, the GRPPRI bits of the DCHPRIn register reflect the current priority level of the group of channels in which the corresponding channel resides. GRPPRI bits are not affected by writes to the DCHPRIn registers. The group priority is assigned in the DMA control register.

14.4.5.18.3 Diagram



14.4.5.18.4 Register reset values

Register	Reset value
DCHPRI0	DMA0,DMA1: 00h
DCHPRI1	DMA0,DMA1: 01h
DCHPRI2	DMA0,DMA1: 02h
DCHPRI3	DMA0,DMA1: 03h
DCHPRI4	DMA0,DMA1: 04h
DCHPRI5	DMA0,DMA1: 05h
DCHPRI6	DMA0,DMA1: 06h
DCHPRI7	DMA0,DMA1: 07h
DCHPRI8	DMA0,DMA1: 08h
DCHPRI9	DMA0,DMA1: 09h
DCHPRI10	DMA0,DMA1: 0Ah
DCHPRI11	DMA0,DMA1: 0Bh
DCHPRI12	DMA0,DMA1: 0Ch
DCHPRI13	DMA0,DMA1: 0Dh
DCHPRI14	DMA0,DMA1: 0Eh
DCHPRI15	DMA0,DMA1: 0Fh
DCHPRI16	DMA0,DMA1: 10h
DCHPRI17	DMA0,DMA1: 11h
DCHPRI18	DMA0,DMA1: 12h
DCHPRI19	DMA0,DMA1: 13h
DCHPRI20	DMA0,DMA1: 14h
DCHPRI21	DMA0,DMA1: 15h
DCHPRI22	DMA0,DMA1: 16h
DCHPRI23	DMA0,DMA1: 17h
DCHPRI24	DMA0,DMA1: 18h
DCHPRI25	DMA0,DMA1: 19h
DCHPRI26	DMA0,DMA1: 1Ah
DCHPRI27	DMA0,DMA1: 1Bh
DCHPRI28	DMA0,DMA1: 1Ch
DCHPRI29	DMA0,DMA1: 1Dh
DCHPRI30	DMA0,DMA1: 1Eh
DCHPRI31	DMA0,DMA1: 1Fh

14.4.5.18.5 Fields

Field	Function
7	Enable Channel Preemption. This field resets to 0. 0b - Channel n cannot be suspended by a higher priority channel's service request.

Table continues on the next page...

Field	Function
ECP	1b - Channel n can be temporarily suspended by the service request of a higher priority channel.
6 DPA	Disable Preempt Ability. This field resets to 0. 0b - Channel n can suspend a lower priority channel. 1b - Channel n cannot suspend any channel, regardless of channel priority.
5-4 GRPPRI	Channel n Current Group Priority Group priority assigned to this channel group when fixed-priority arbitration is enabled. This field is read-only; writes are ignored.
3-0 CHPRI	Channel n Arbitration Priority Channel priority when fixed-priority arbitration is enabled

14.4.5.19 TCD Source Address (TCD0_SADDR - TCD31_SADDR)

14.4.5.19.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_SADDR	1000h + (n × 20h)

14.4.5.19.2 Function

This register contains the source address of the transfer.

14.4.5.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

14.4.5.19.4 Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

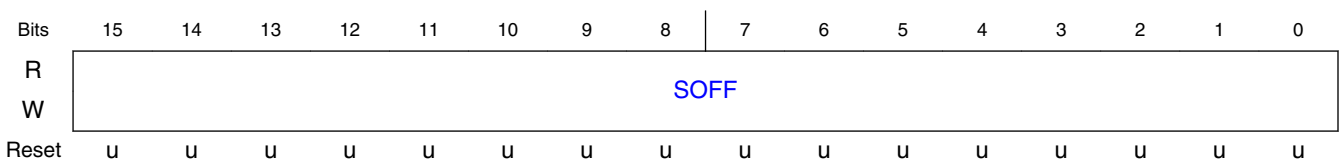
14.4.5.20 TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)

14.4.5.20.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_SOFF	1004h + (n × 20h)

14.4.5.20.2 Diagram



14.4.5.20.3 Fields

Field	Function
15-0 SOFF	Source address signed offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

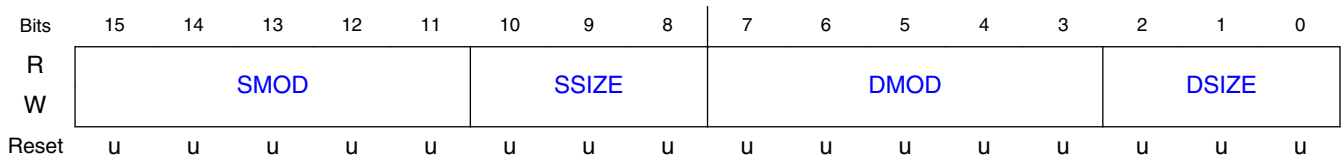
14.4.5.21 TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)

14.4.5.21.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_ATTR	1006h + (n × 20h)

14.4.5.21.2 Diagram



14.4.5.21.3 Fields

Field	Function
15-11 SMOD	Source Address Modulo 00000b - Source address modulo feature is disabled 00001-11111b - This value defines a specific address range specified to be the value after SADDR + SOFF calculation is performed on the original register value. Setting this field provides the ability to implement a circular data queue easily. For data queues requiring power-of-2 size bytes, the queue should start at a 0-modulo-size address and the SMOD field should be set to the appropriate value for the queue, freezing the desired number of upper address bits. The value programmed into this field specifies the number of lower address bits allowed to change. For a circular queue application, the SOFF is typically set to the transfer size to implement post-increment addressing with the SMOD function constraining the addresses to a 0-modulo-size range.
10-8 SSIZE	Source data transfer size **NOTE:** Using a Reserved value causes a configuration error. **NOTE:** The eDMA defaults to privileged data access for all transactions. 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - Reserved 101b - 32-byte burst (4 beats of 64 bits) 110b - Reserved 111b - Reserved
7-3 DMOD	Destination Address Modulo See the SMOD definition
2-0 DSIZE	Destination data transfer size See the SSIZE definition

14.4.5.22 TCD Minor Byte Count (Minor Loop Mapping Disabled) (TCD0_NBYTES_MLNO - TCD31_NBYTES_MLNO)

14.4.5.22.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_NBYTES_MLNO	1008h + (n × 20h)

14.4.5.22.2 Function

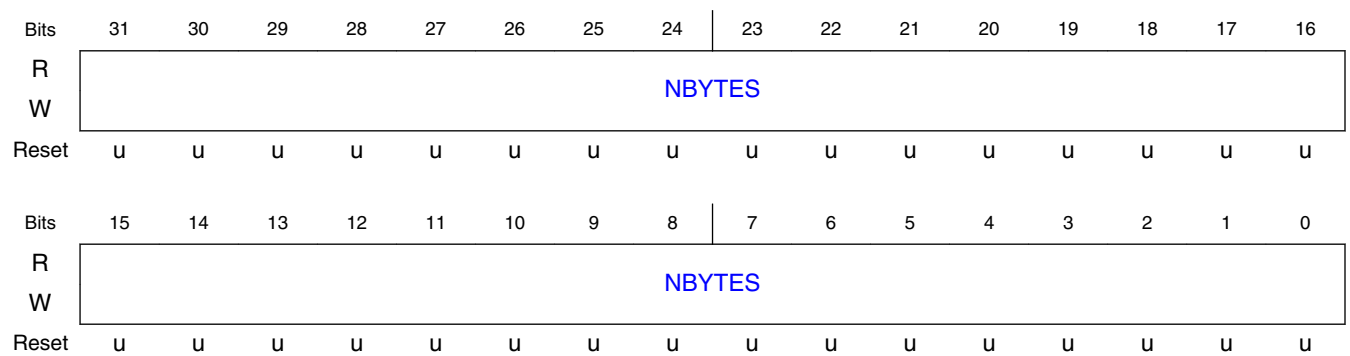
This register, or one of the next two registers (TCD_NBYTES_MLOFFNO, TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is disabled (CR[EMLM] = 0)

If minor loop mapping is enabled, see the TCD_NBYTES_MLOFFNO and TCD_NBYTES_MLOFFYES register descriptions for the definition of TCD word 2.

14.4.5.22.3 Diagram



14.4.5.22.4 Fields

Field	Function
31-0 NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

Field	Function
	NOTE: An NBYTES value of 0x0000_0000 is interpreted as a 4 GB transfer.

14.4.5.23 TCD Signed Minor Loop Offset (Minor Loop Mapping Enabled and Offset Disabled) (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)

14.4.5.23.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_NBYTES_MLOFFNO	1008h + (n × 20h)

14.4.5.23.2 Function

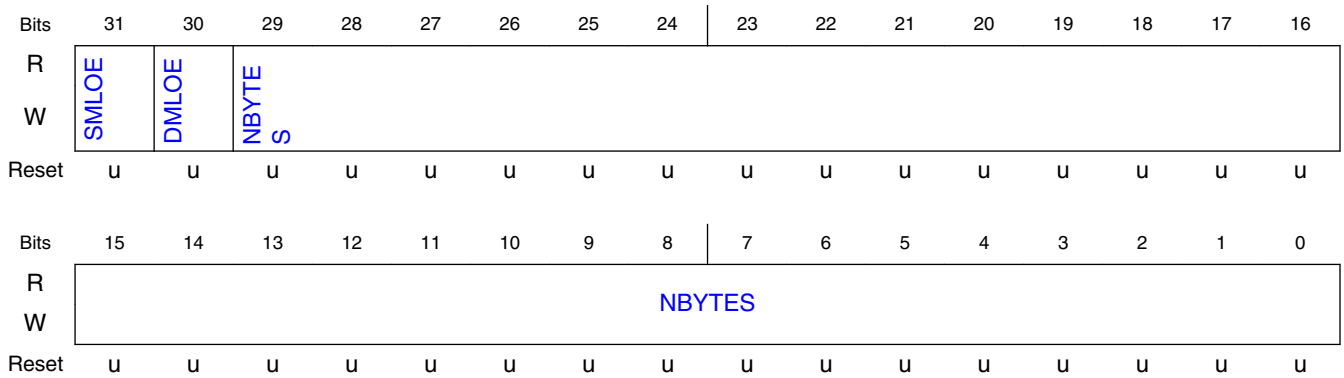
One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- SMLOE = 0 and DMLOE = 0

If minor loop mapping is enabled and SMLOE or DMLOE is set, then refer to the TCD_NBYTES_MLOFFYES register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

14.4.5.23.3 Diagram



14.4.5.23.4 Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - The minor loop offset is not applied to the SADDR 1b - The minor loop offset is applied to the SADDR
30 DMLOE	Destination Minor Loop Offset enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - The minor loop offset is not applied to the DADDR 1b - The minor loop offset is applied to the DADDR
29-0 NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

14.4.5.24 TCD Signed Minor Loop Offset (Minor Loop Mapping and Offset Enabled) (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)

14.4.5.24.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_NBYTES_MLOFFYES	1008h + (n × 20h)

14.4.5.24.2 Function

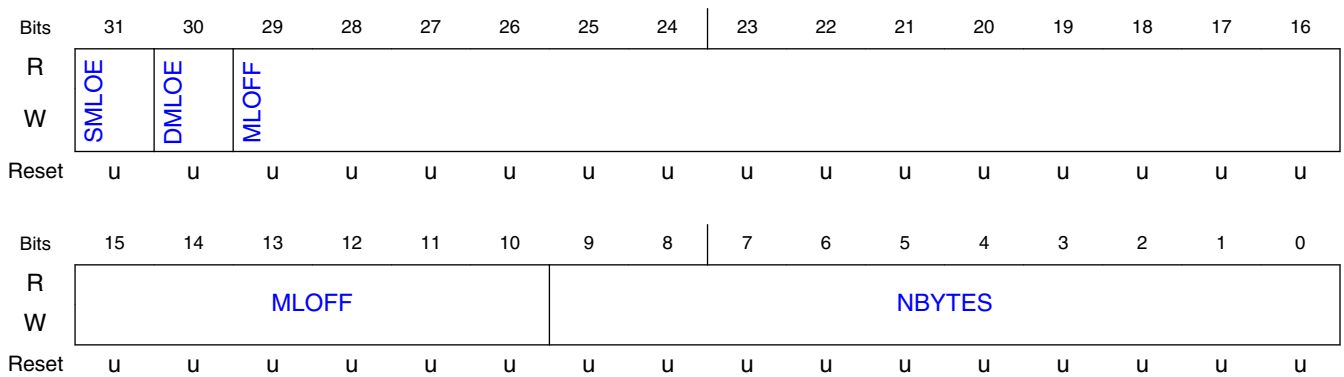
One of three registers (this register, TCD_NBYTES_MLNO, or TCD_NBYTES_MLOFFNO), defines the number of bytes to transfer per request. Which register to use depends on whether minor loop mapping is disabled, enabled but not used for this channel, or enabled and used.

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- Minor loop offset is enabled (SMLOE or DMLOE = 1)

If minor loop mapping is enabled and SMLOE and DMLOE are cleared, then refer to the TCD_NBYTES_MLOFFNO register description. If minor loop mapping is disabled, then refer to the TCD_NBYTES_MLNO register description.

14.4.5.24.3 Diagram



14.4.5.24.4 Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - The minor loop offset is not applied to the SADDR 1b - The minor loop offset is applied to the SADDR
30	Destination Minor Loop Offset enable

Table continues on the next page...

Field	Function
DMLOE	Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - The minor loop offset is not applied to the DADDR 1b - The minor loop offset is applied to the DADDR
29-10 MLOFF	If SMLOE or DMLOE is set, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
9-0 NBYTES	Minor Byte Transfer Count Number of bytes to be transferred in each service request of the channel. As a channel activates, the appropriate TCD contents load into the eDMA engine, and the appropriate reads and writes perform until the minor byte transfer count has transferred. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, the major iteration count is decremented and restored to the TCD memory. If the major iteration count is completed, additional processing is performed.

14.4.5.25 TCD Last Source Address Adjustment (TCD0_SLAST - TCD31_SLAST)

14.4.5.25.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_SLAST	100Ch + (n × 20h)

14.4.5.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SLAST															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SLAST															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

14.4.5.25.3 Fields

Field	Function
31-0 SLAST	Last Source Address Adjustment Adjustment value added to the source address at the completion of the major iteration count. This value can be applied to restore the source address to the initial value, or adjust the address to reference the next data structure. This register uses two's complement notation; the overflow bit is discarded.

14.4.5.26 TCD Destination Address (TCD0_DADDR - TCD31_DADDR)

14.4.5.26.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_DADDR	1010h + (n × 20h)

14.4.5.26.2 Function

This register contains the destination address of the transfer.

14.4.5.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

14.4.5.26.4 Fields

Field	Function
31-0	Destination Address

Field	Function
DADDR	Memory address pointing to the destination data.

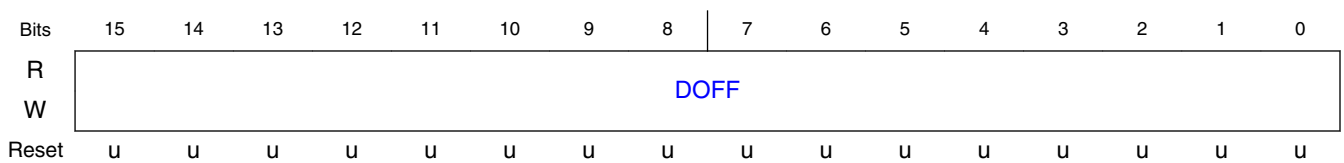
14.4.5.27 TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)

14.4.5.27.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_DOFF	1014h + (n × 20h)

14.4.5.27.2 Diagram



14.4.5.27.3 Fields

Field	Function
15-0 DOFF	Destination Address Signed Offset Sign-extended offset applied to the current destination address to form the next-state value as each destination write is completed.

14.4.5.28 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)

14.4.5.28.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_CITER_ELINKNO	1016h + (n × 20h)

14.4.5.28.2 Function

This register contains the minor-loop channel-linking configuration and the channel's current iteration count. It is the same register as TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES), but its fields are defined differently based on the state of the ELINK field. If the ELINK field is cleared, this register is defined as follows.

14.4.5.28.3 Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ELINK	CITER														
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

14.4.5.28.4 Fields

Field	Function
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported. 0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14-0 CITER	Current Major Iteration Count This field is the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

14.4.5.29 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)

14.4.5.29.1 Offset

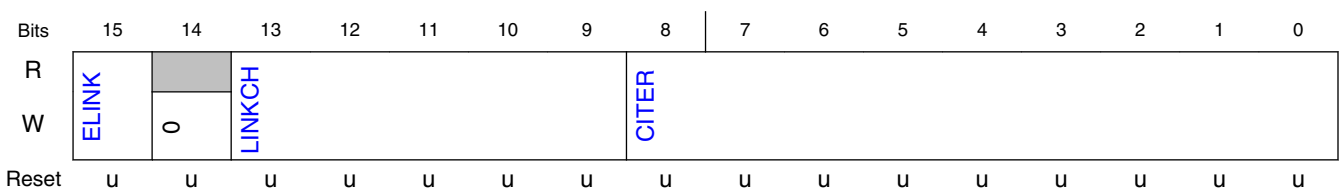
For n = 0 to 31:

Register	Offset
TCDn_CITER_ELINKYES	1016h + (n × 20h)

14.4.5.29.2 Function

This register contains the minor-loop channel-linking configuration and the channel's current iteration count. It is the same register as TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO), but its fields are defined differently based on the state of the ELINK field. If the ELINK field is set, this register is defined as follows.

14.4.5.29.3 Diagram



14.4.5.29.4 Fields

Field	Function
15 ELINK	Enable channel-to-channel linking on minor-loop complete As the channel completes the minor loop, this flag enables linking to another channel, defined by the LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** This bit must be equal to the BITER[ELINK] bit; otherwise, a configuration error is reported.

Table continues on the next page...

Field	Function
	0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14 —	Reserved
13-9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request to the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
8-0 CITER	Current Major Iteration Count This field is the current major loop count for the channel. It is decremented each time the minor loop is completed and updated in the transfer control descriptor memory. After the major iteration count is exhausted, the channel performs a number of operations, for example, final source and destination address calculations, optionally generating an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE:** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE:** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

14.4.5.30 TCD Last Destination Address Adjustment/Scatter Gather Address (TCD0_DLASTSGA - TCD31_DLASTSGA)

14.4.5.30.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_DLASTSGA	1018h + (n × 20h)

14.4.5.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DLASTSGA															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DLASTSGA															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

14.4.5.30.3 Fields

Field	Function
31-0 DLASTSGA	DLASTSGA Destination last address adjustment or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo-32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo-32-byte, otherwise a configuration error is reported.

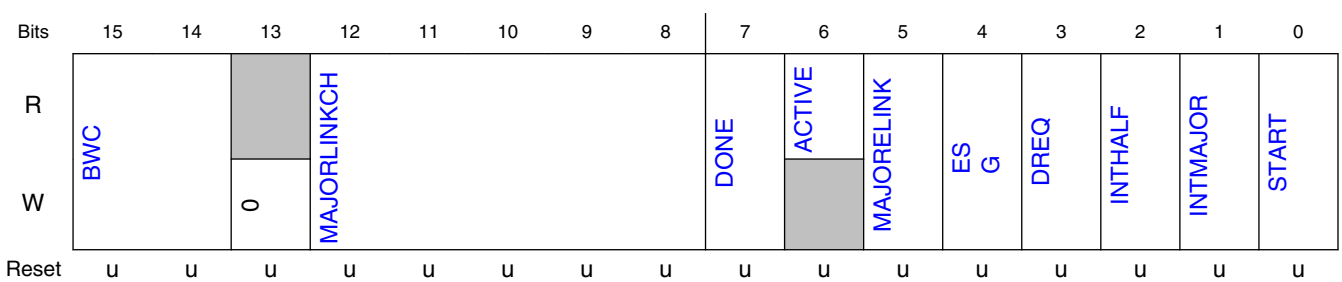
14.4.5.31 TCD Control and Status (TCD0_CSR - TCD31_CSR)

14.4.5.31.1 Offset

For n = 0 to 31:

Register	Offset
TCDn_CSR	101Ch + (n × 20h)

14.4.5.31.2 Diagram



14.4.5.31.3 Fields

Field	Function
15-14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces the eDMA to stall after the completion of each read/write access to control the bus request bandwidth seen by the crossbar switch. **NOTE:** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00b - No eDMA engine stalls. 01b - Reserved 10b - eDMA engine stalls for 4 cycles after each R/W. 11b - eDMA engine stalls for 8 cycles after each R/W.
13 —	Reserved
12-8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit.
7 DONE	Channel Done This flag indicates the eDMA has completed the major loop. The eDMA engine sets it as the CITER count reaches zero. The software clears it, or the hardware when the channel is activated. **NOTE:** This bit must be cleared to write the MAJORELINK or ESG bits.
6 ACTIVE	Channel Active This flag signals the channel is currently in execution. It is set when channel service begins, and is cleared by the eDMA as the minor loop completes or when any error condition is detected.
5 MAJORELINK	Enable channel-to-channel linking on major loop complete As the channel completes the major loop, this flag enables the linking to another channel, defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. **NOTE:** To support the dynamic linking coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0b - The channel-to-channel linking is disabled. 1b - The channel-to-channel linking is enabled.
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses DLASTSGA as a memory pointer to a 0-modulo-32 address containing a 32-byte data structure loaded as the transfer control descriptor into the local memory. **NOTE:** To support the dynamic scatter/gather coherency model, this field is forced to zero when written to while the TCDn_CSR[DONE] bit is set. 0b - The current channel's TCD is normal format. 1b - The current channel's TCD specifies a scatter gather format. The DLASTSGA field provides a memory pointer to the next TCD to be loaded into this channel after the major loop completes its execution.

Table continues on the next page...

Field	Function
3 DREQ	Disable Request If this flag is set, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches zero. 0b - The channel's ERQ bit is not affected. 1b - The channel's ERQ bit is cleared when the major loop is complete.
2 INTHALF	Enable an interrupt when major counter is half complete. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT register when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER == (BITER >> 1)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE:** If BITER = 1, do not use INTHALF. Use INTMAJOR instead. 0b - The half-point interrupt is disabled. 1b - The half-point interrupt is enabled.
1 INTMAJOR	Enable an interrupt when major iteration count completes. If this flag is set, the channel generates an interrupt request by setting the appropriate bit in the INT when the current major iteration count reaches zero. 0b - The end-of-major loop interrupt is disabled. 1b - The end-of-major loop interrupt is enabled.
0 START	Channel Start If this flag is set, the channel is requesting service. The eDMA hardware automatically clears this flag after the channel begins execution. 0b - The channel is not explicitly started. 1b - The channel is explicitly started via a software initiated service request.

14.4.5.32 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)

14.4.5.32.1 Offset

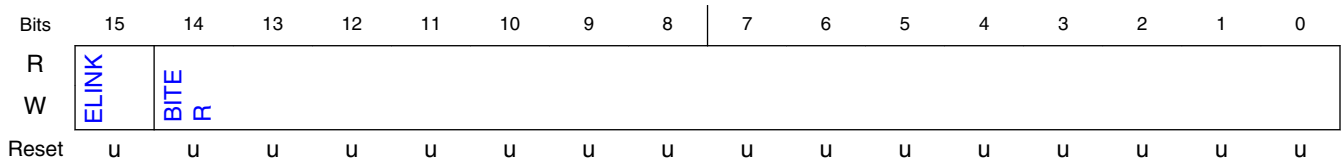
For n = 0 to 31:

Register	Offset
TCDn_BITER_ELINKNO	101Eh + (n × 20h)

14.4.5.32.2 Function

If the TCDn_BITER[ELINK] bit is cleared, the TCDn_BITER register is defined as follows.

14.4.5.32.3 Diagram



14.4.5.32.4 Fields

Field	Function
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field is reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

14.4.5.33 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)

14.4.5.33.1 Offset

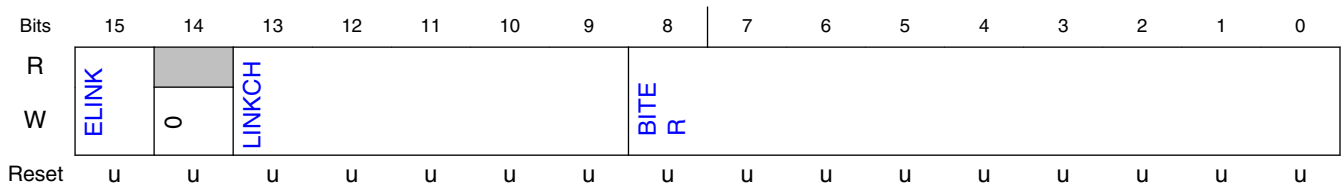
For n = 0 to 31:

Register	Offset
TCDn_BITER_ELINKYES	101Eh + (n × 20h)

14.4.5.33.2 Function

If the TCDn_BITER[ELINK] bit is set, the TCDn_BITER register is defined as follows.

14.4.5.33.3 Diagram



14.4.5.33.4 Fields

Field	Function
15 ELINK	Enables channel-to-channel linking on minor loop complete As the channel completes the minor loop, this flag enables the linking to another channel, defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. 0b - The channel-to-channel linking is disabled 1b - The channel-to-channel linking is enabled
14 —	Reserved
13-9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] bit. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field.
8-0 BITER	Starting major iteration count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be equal to the value in the CITER field. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. **NOTE:** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, the contents of this field are reloaded into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

14.5 Functional description

The operation of the eDMA is described in the following subsections.

14.5.1 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

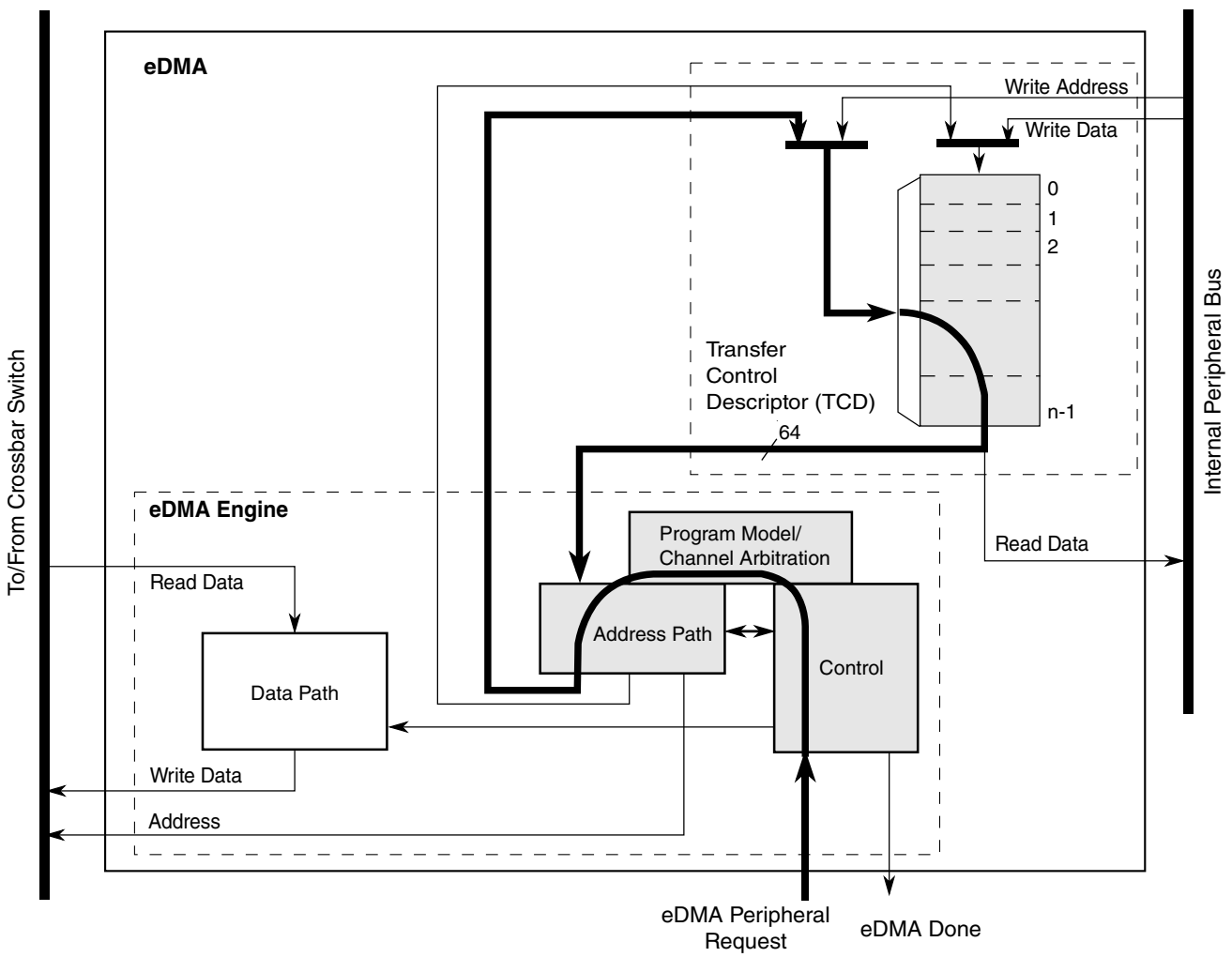


Figure 14-2. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] bit follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module,

then into the program model and channel arbitration. In the next cycle, the channel arbitration performs, using the fixed-priority or round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor read from the local memory and loaded into the eDMA engine address path channel x or y registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path channel x or y registers.

The following diagram illustrates the second part of the basic data flow:

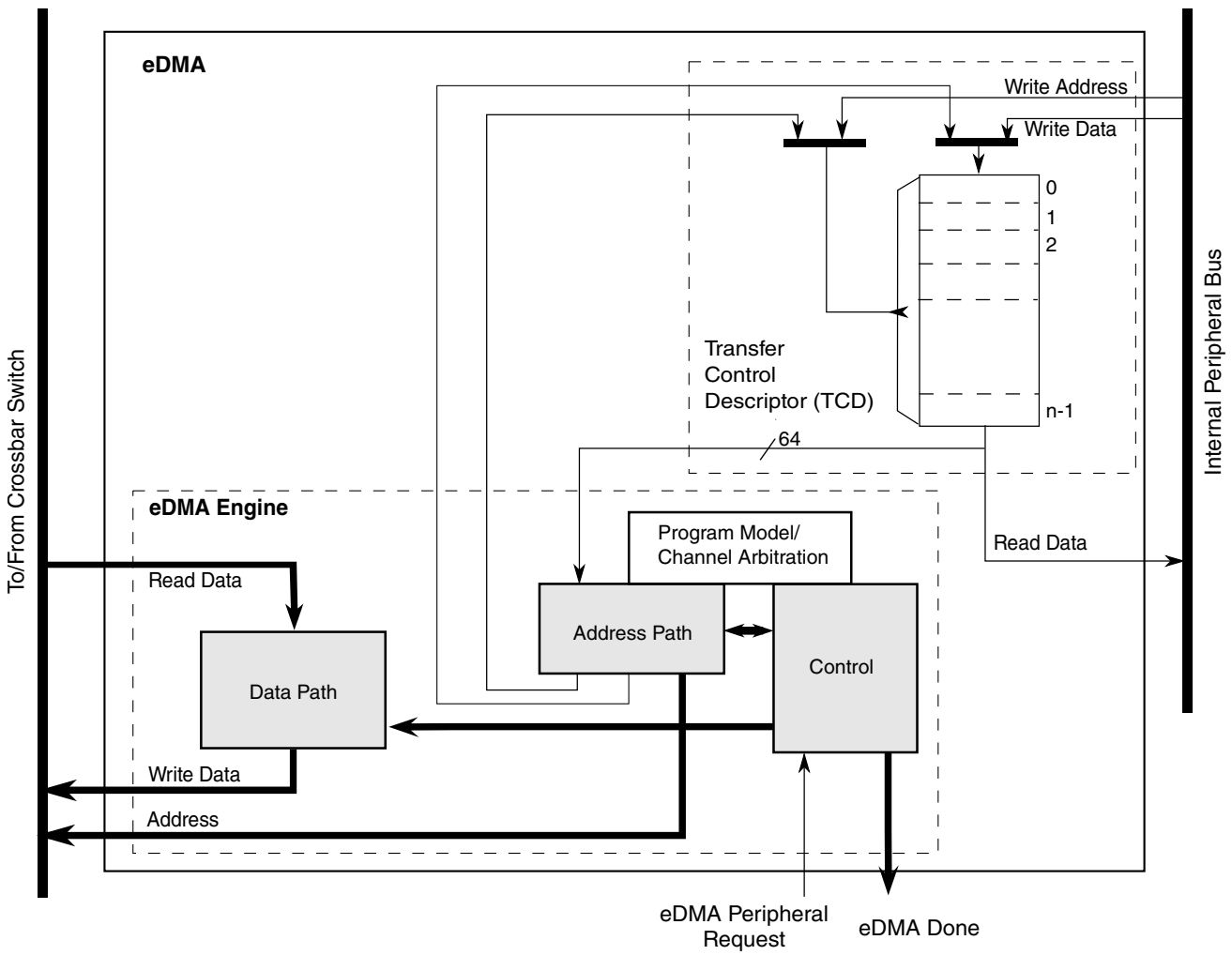


Figure 14-3. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) sequence through the required source reads and destination writes to perform the actual data movement. The source reads are initiated and the fetched data is temporarily stored

in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the minor byte count has transferred.

After the minor byte count has moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD, for example, SADDR, DADDR, CITER. If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

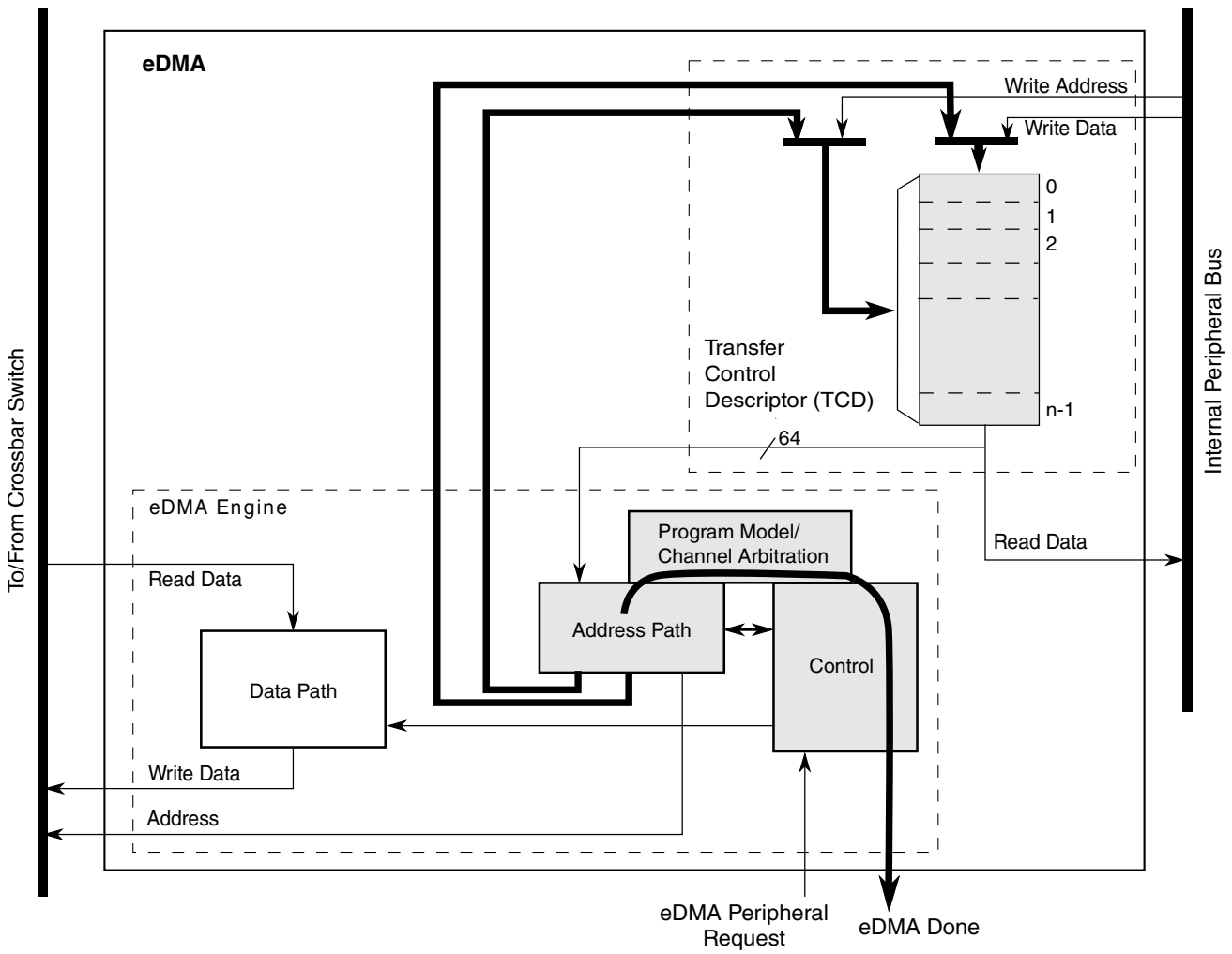


Figure 14-4. eDMA operation, part 3

14.5.2 Fault reporting and handling

Channel errors are reported in the Error Status register (DMAx_ES) and can be caused by:

- A configuration error, which is an illegal setting in the transfer-control descriptor or an illegal priority register setting in Fixed-Arbitration mode, or
- An error termination to a bus master read or write cycle

A configuration error is reported when the starting source or destination address, source or destination offsets, minor loop byte count, or the transfer size represent an inconsistent state. Each of these possible causes are detailed below:

- The addresses and offsets must be aligned on 0-modulo-transfer-size boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.
- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size respectively.
- In fixed arbitration mode, a configuration error is caused by any two channel priorities being equal. All channel priority levels must be unique when fixed arbitration mode is enabled.

NOTE

When two channels have the same priority, a channel priority error exists and will be reported in the Error Status register. However, the channel number will not be reported in the Error Status register. When all of the channel priorities within a group are not unique, the channel number selected by arbitration in undetermined.

To aid in Channel Priority Error (CPE) debug, set the Halt On Error bit in the DMA's Control Register. If all of the channel priorities within a group are not unique, the DMA will be halted after the CPE error is recorded. The DMA will remain halted and will not process any channel service requests. Once all of the channel priorities are set to unique numbers, the DMA may be enabled again by clearing the Halt bit.

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[E_LINK] bit does not equal the TCDn_BITER[E_LINK] bit.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, report as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion when properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

If a system bus read or write is terminated with an error, the data transfer is stopped and the appropriate bus error flag set. In this case, the state of the channel's transfer control descriptor is updated by the eDMA engine with the current source address, destination address, and current iteration count at the point of the fault. When a system bus error occurs, the channel terminates after the next transfer. Due to pipeline effect, the next transfer is already in progress when the bus error is received by the eDMA. If a bus error occurs on the last read prior to beginning the write sequence, the write executes using the data captured during the bus error. If a bus error occurs on the last write prior to switching to the next read sequence, the read sequence executes before the channel terminates due to the destination bus error.

A transfer may be cancelled by software with the CR[CX] bit. When a cancel transfer request is recognized, the DMA engine stops processing the channel. The current read-write sequence is allowed to finish. If the cancel occurs on the last read-write sequence of a major or minor loop, the cancel request is discarded and the channel retires normally.

The error cancel transfer is the same as a cancel transfer except the Error Status register (DMAx_ES) is updated with the cancelled channel number and ECX is set. The TCD of a cancelled channel contains the source and destination addresses of the last transfer saved in the TCD. If the channel needs to be restarted, you must re-initialize the TCD because the aforementioned fields no longer represent the original parameters. When a transfer is cancelled by the error cancel transfer mechanism, the channel number is loaded into DMA_ES[ERRCHN] and ECX and VLD are set. In addition, an error interrupt may be generated if enabled.

NOTE

The cancel transfer request allows the user to stop a large data transfer in the event the full data transfer is no longer needed. The cancel transfer bit does not abort the channel. It simply stops the transferring of data and then retires the channel through its normal shutdown sequence. The application

software must handle the context of the cancel. If an interrupt is desired (or not), then the interrupt should be enabled (or disabled) before the cancel request. The application software must clean up the transfer control descriptor since the full transfer did not occur.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel bit in the eDMA error register is asserted. At the same time, the details of the error condition are loaded into the Error Status register (DMAx_ES). The major loop complete indicators, setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request, are not affected when an error is detected. After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

14.5.3 Channel preemption

Channel preemption is enabled on a per-channel basis by setting the DCHPRIn[ECP] bit. Channel preemption allows the executing channel's data transfers to temporarily suspend in favor of starting a higher priority channel. After the preempting channel has completed all its minor loop data transfers, the preempted channel is restored and resumes execution. After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended and the higher priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted. Preemption is available only when fixed arbitration is selected.

A channel's ability to preempt another channel can be disabled by setting DCHPRIn[DPA]. When a channel's preempt ability is disabled, that channel cannot suspend a lower priority channel's data transfer, regardless of the lower priority channel's ECP setting. This allows for a pool of low priority, large data-moving channels to be defined. These low priority channels can be configured to not preempt each other, thus preventing a low priority channel from consuming the preempt slot normally available to a true, high priority channel.

14.5.4 Performance

This section addresses the performance of the eDMA module, focusing on two separate metrics:

- In the traditional data movement context, performance is best expressed as the peak data transfer rates achieved using the eDMA. In most implementations, this transfer rate is limited by the speed of the source and destination address spaces.
- In a second context where device-paced movement of single data values to/from peripherals is dominant, a measure of the requests that can be serviced in a fixed time is a more relevant metric. In this environment, the speed of the source and destination address spaces remains important. However, the microarchitecture of the eDMA also factors significantly into the resulting metric.

14.5.4.1 Peak transfer rates

The peak transfer rates for several different source and destination transfers are shown in the following tables. These tables assume:

- Internal SRAM can be accessed with zero wait-states when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states, when viewed from the system bus data phase
- All internal peripheral bus accesses are 32-bits in size

NOTE

All architectures will not meet the assumptions listed above. See the SRAM configuration section for more information.

This table compares peak transfer rates based on different possible system speeds. Specific chips/devices may not support all system speeds listed.

Table 14-6. eDMA peak transfer rates (Mbytes/sec)

System Speed, Width	Internal SRAM-to-Internal SRAM	32 bit internal peripheral bus-to-Internal SRAM	Internal SRAM-to-32 bit internal peripheral bus
66.7 MHz, 64 bit	266.7	66.6	53.3
83.3 MHz, 64 bit	333.3	83.3	66.7
100.0 MHz, 64 bit	400.0	100.0	80.0
133.3 MHz, 64 bit	533.3	133.3	106.7
150.0 MHz, 64 bit	600.0	150.0	120.0

Internal-SRAM-to-internal-SRAM transfers occur at the core's datapath width. For all transfers involving the internal peripheral bus, 32-bit transfer sizes are used. In all cases, the transfer rate includes the time to read the source plus the time to write the destination.

14.5.4.2 Peak request rates

The second performance metric is a measure of the number of DMA requests that can be serviced in a given amount of time. For this metric, assume that the peripheral request causes the channel to move a single internal peripheral bus-mapped operand to/from internal SRAM. The same timing assumptions used in the previous example apply to this calculation. In particular, this metric also reflects the time required to activate the channel.

The eDMA design supports the following hardware service request sequence. Note that the exact timing from Cycle 7 is a function of the response times for the channel's read and write accesses. In the case of an internal peripheral bus read and internal SRAM write, the combined data phase time is 4 cycles. For an SRAM read and internal peripheral bus write, it is 5 cycles.

Table 14-7. Hardware service request process

Cycle		Description
With internal peripheral bus read and internal SRAM write	With SRAM read and internal peripheral bus write	
1		eDMA peripheral request is asserted.
2		The eDMA peripheral request is registered locally in the eDMA module and qualified. TCD*n*_CSR[START] bit initiated requests start at this point with the registering of the user write to TCD*n* word 7.
3		Channel arbitration begins.
4		Channel arbitration completes. The transfer control descriptor local memory read is initiated.
5–6		The first two parts of the activated channel's TCD is read from the local memory. The memory width to the eDMA engine is 64 bits, so the entire descriptor can be accessed in four cycles
7		The first system bus read cycle is initiated, as the third part of the channel's TCD is read from the local memory. Depending on the state of the crossbar switch, arbitration at the system bus may insert an additional cycle of delay here.
8–11	8–12	The last part of the TCD is read in. This cycle represents the first data phase for the read, and the address phase for the destination write.
12	13	This cycle represents the data phase of the last destination write.

Table continues on the next page...

Table 14-7. Hardware service request process (continued)

Cycle		Description
With internal peripheral bus read and internal SRAM write	With SRAM read and internal peripheral bus write	
13	14	The eDMA engine completes the execution of the inner minor loop and prepares to write back the required TCD*n* fields into the local memory. The TCD*n* word 7 is read and checked for channel linking or scatter/gather requests.
14	15	The appropriate fields in the first part of the TCD*n* are written back into the local memory.
15	16	The fields in the second part of the TCD*n* are written back into the local memory. This cycle coincides with the next channel arbitration cycle start.
16	17	The next channel to be activated performs the read of the first part of its TCD from the local memory. This is equivalent to Cycle 4 for the first channel's service request.

Assuming zero wait states on the system bus, DMA requests can be processed every 9 cycles. Assuming an average of the access times associated with internal peripheral bus-to-SRAM (4 cycles) and SRAM-to-internal peripheral bus (5 cycles), DMA requests can be processed every 11.5 cycles (4 + (4+5)/2 + 3). This is the time from Cycle 4 to Cycle x +5. The resulting peak request rate, as a function of the system frequency, is shown in the following table.

Table 14-8. eDMA peak request rate (MReq/sec)

System frequency (MHz)	Request rate with zero wait states	Request rate with wait states
48.0	5.3	4.2
66.6	7.4	5.8
83.3	9.2	7.2
100.0	11.1	8.7
133.3	14.8	11.6
150.0	16.6	13.0

A general formula to compute the peak request rate with overlapping requests is:

PEAKreq = freq / [entry + (1 + read_ws) + (1 + write_ws) + exit]

where:

Table 14-9. Peak request formula operands

Operand	Description
PEAKreq	Peak request rate

Table continues on the next page...

Table 14-9. Peak request formula operands (continued)

Operand	Description
freq	System frequency
entry	Channel startup (4 cycles)
read_ws	Wait states seen during the system bus read data phase
write_ws	Wait states seen during the system bus write data phase
exit	Channel shutdown (3 cycles)

14.5.4.3 eDMA performance example

Consider a system with the following characteristics:

- Internal SRAM can be accessed with one wait-state when viewed from the system bus data phase
- All internal peripheral bus reads require two wait-states, and internal peripheral bus writes three wait-states viewed from the system bus data phase
- System operates at 150 MHz

For an SRAM to internal peripheral bus transfer,

PEAKreq = 150 MHz / [4 + (1 + 1) + (1 + 3) + 3] cycles = 11.5 Mreq/sec

For an internal peripheral bus to SRAM transfer,

PEAKreq = 150 MHz / [4 + (1 + 2) + (1 + 1) + 3] cycles = 12.5 Mreq/sec

Assuming an even distribution of the two transfer types, the average peak request rate would be:

PEAKreq = (11.5 Mreq/sec + 12.5 Mreq/sec) / 2 = 12.0 Mreq/sec

The minimum number of cycles to perform a single read/write, zero wait states on the system bus, from a cold start where no channel is executing and eDMA is idle are:

- 11 cycles for a software, that is, a TCD*n*_CSR[START] bit, request
- 12 cycles for a hardware, that is, an eDMA peripheral request signal, request

Two cycles account for the arbitration pipeline and one extra cycle on the hardware request resulting from the internal registering of the eDMA peripheral request signals. For the peak request rate calculations above, the arbitration and request registering is absorbed in or overlaps the previous executing channel.

Note

When channel linking or scatter/gather is enabled, a two cycle delay is imposed on the next channel selection and startup. This allows the link channel or the scatter/gather channel to be eligible and considered in the arbitration pool for next channel selection.

14.6 Initialization/application information

The following sections discuss initialization of the eDMA and programming considerations.

14.6.1 eDMA initialization

To initialize the eDMA:

1. Write to the CR if a configuration other than the default is desired.
2. Write the channel priority levels to the DCHPRI*n* registers if a configuration other than the default is desired.
3. Enable error interrupts in the EEI register if so desired.
4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the ERQ register.
6. Request channel service via either:
   - Software: setting the TCD*n*_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in the following table, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, as defined by TCD*n*_SADDR, to the destination, as defined by TCD*n*_DADDR, continue until the number of bytes specified by TCD*n*_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, further post processing executes, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 14-10. TCD Control and Status fields

TCD*n*_CSR field name	Description
START	Control bit to start channel explicitly when using a software initiated DMA service (Automatically cleared by hardware)
ACTIVE	Status bit indicating the channel is currently in execution
DONE	Status bit indicating major loop completion (cleared by software when using a software initiated DMA service)
D_REQ	Control bit to disable DMA request at end of major loop completion when using a hardware initiated DMA service
BWC	Control bits for throttling bandwidth control of a channel
E_SG	Control bit to enable scatter-gather feature
INT_HALF	Control bit to enable interrupt when major loop is half complete
INT_MAJ	Control bit to enable interrupt when major loop completes

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

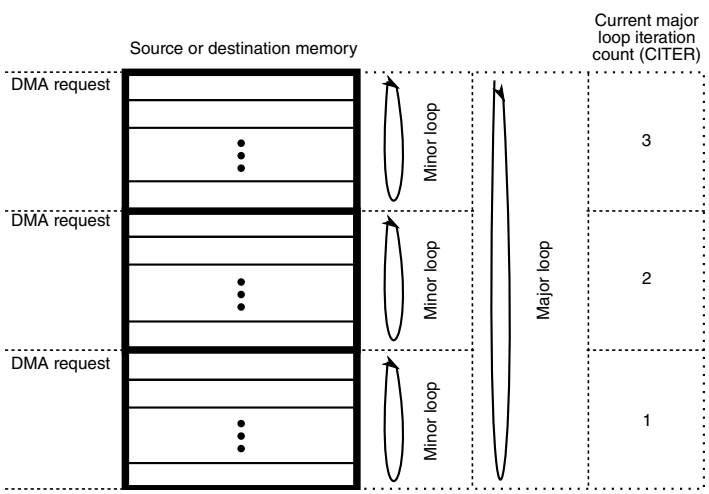


Figure 14-5. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings interrelate.

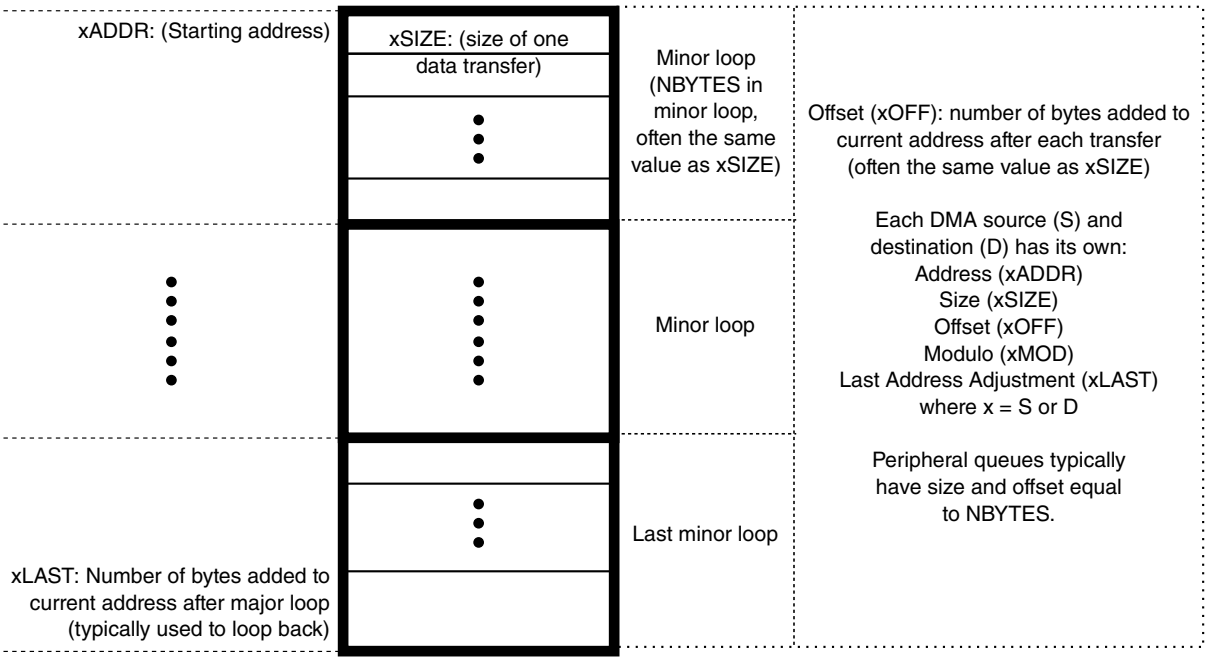


Figure 14-6. Memory array terms

14.6.2 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data. Most programming errors are reported on a per channel basis with the exception of channel priority error (ES[CPE]).

For all error types other than group or channel priority errors, the channel number causing the error is recorded in the Error Status register (DMAx_ES). If the error source is not removed before the next activation of the problem channel, the error is detected and recorded again.

Channel priority errors are identified within a group once that group has been selected as the active group. For example:

1. The eDMA is configured for fixed group and fixed channel arbitration modes.
2. Group 1 is the highest priority and all channels are unique in that group.
3. Group 0 is the next highest priority and has two channels with the same priority level.
4. If Group 1 has any service requests, those requests will be executed.
5. After all of Group 1 requests have completed, Group 0 will be the next active group.

6. If Group 0 has a service request, then an undefined channel in Group 0 will be selected and a channel priority error will occur.
7. This repeats until the all of Group 0 requests have been removed or a higher priority Group 1 request comes in.

In this sequence, for item 2, the eDMA acknowledge lines will assert only if the selected channel is requesting service via the eDMA peripheral request signal. If interrupts are enabled for all channels, the user will get an error interrupt, but the channel number for the ERR register and the error interrupt request line may be wrong because they reflect the selected channel. A group priority error is global and any request in any group will cause a group priority error.

If priority levels are not unique, when any channel requests service, a channel priority error is reported. The highest channel/group priority with an active request is selected, but the lowest numbered channel with that priority is selected by arbitration and executed by the eDMA engine. The hardware service request handshake signals, error interrupts, and error reporting is associated with the selected channel.

14.6.3 Arbitration mode considerations

This section discusses arbitration considerations for the eDMA.

14.6.3.1 Fixed group arbitration, Fixed channel arbitration

In this mode, the channel service request from the highest priority channel in the highest priority group is selected to execute. If the eDMA is programmed so that the channels within one group use "fixed" priorities, and that group is assigned the highest "fixed" priority of all groups, that group can take all the bandwidth of the eDMA controller. That is, no other groups will be serviced if there is always at least one DMA request pending on a channel in the highest priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly. Preemption is available in this scenario only.

14.6.3.2 Fixed group arbitration, Round-robin channel arbitration

The highest priority group with a request will be serviced. Lower priority groups will be serviced if no pending requests exist in the higher priority groups.

Within each group, c hannels are serviced starting with the highest channel number and rotating through to the lowest channel number without regard to the channel priority levels assigned within the group.

This scenario could cause the same bandwidth consumption problem as indicated in Fixed group arbitration, Fixed channel arbitration, but all the channels in the highest priority group will be serviced. Service latency will be short on the highest priority group, but could potentially be very much longer as the group priority decreases.

14.6.4 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

14.6.4.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the TCD*n*_CSR[DONE] bit is set and an interrupt generates if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
TCDn_DLAST_SGA= -16
TCDn_CSR[INT_MAJ] = 1
TCDn_CSR[START] = 1 (Should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] bit requests channel service.

2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
    b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
    c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
    d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
    e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
    f. Write 32-bits to location 0x2008 → third iteration of the minor loop.
    g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
    h. Write 32-bits to location 0x200C → last iteration of the minor loop → major loop complete.
6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[*n*] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

14.6.4.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop transferring 16 bytes per iteration. After the channel's hardware requests are enabled in the ERQ register, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware, that is, eDMA peripheral, request for channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source to destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32-bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32-bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
   f. Write 32-bits to location 0x2008 → third iteration of the minor loop.
   g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
   h. Write 32-bits to location 0x200C → last iteration of the minor loop.
6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0.
8. The channel retires → one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware, that is, eDMA peripheral, requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: TCD*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, TCD*n*_CSR[ACTIVE] = 1.

12. eDMA engine reads: channel TCD data from local memory to internal register file.
13. The source to destination transfers are executed as follows:
    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.
    b. Write 32-bits to location 0x2010 → first iteration of the minor loop.
    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.
    d. Write 32-bits to location 0x2014 → second iteration of the minor loop.
    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.
    f. Write 32-bits to location 0x2018 → third iteration of the minor loop.
    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.
    h. Write 32-bits to location 0x201C → last iteration of the minor loop → major loop complete.
14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).
15. eDMA engine writes: TCD*n*_CSR[ACTIVE] = 0, TCD*n*_CSR[DONE] = 1, INT[n] = 1.
16. The channel retires → major loop complete. The eDMA goes idle or services the next channel.

14.6.4.3 Using the modulo feature

The modulo feature of the eDMA provides the ability to implement a circular data queue in which the size of the queue is a power of 2. MOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value while the 28 upper address bits (0x1234567*x*) retain their original value. In this example the source address is set to 0x12345670, the offset is set to 4 bytes and the MOD field is set to 4, allowing for a 2^4 byte (16-byte) size queue.

Table 14-11. Modulo example

Transfer Number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

14.6.5 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

14.6.5.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software initiated service requests. The first is to read the TCD*n*_CITER field and test for a change. Another method may be extracted from the sequence shown below. The second method is to test the TCD*n*_CSR[START] bit and the TCD*n*_CSR[ACTIVE] bit. The minor-loop-complete condition is indicated by both bits reading zero after the TCD*n*_CSR[START] was set. Polling the TCD*n*_CSR[ACTIVE] bit may be inconclusive, because the active status may be missed if the channel execution is short in duration.

The TCD status bits execute the following sequence for a software activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	1	0	0	Channel service request via software
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

The best method to test for minor-loop completion when using hardware, that is, peripheral, initiated service requests is to read the TCD*n*_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status bits execute the following sequence for a hardware-activated channel:

Stage	TCD*n*_CSR bits			State
	START	ACTIVE	DONE	
1	0	0	0	Channel service request via hardware (peripheral request asserted)
2	0	1	0	Channel is executing
3a	0	0	0	Channel has completed the minor loop and is idle
3b	0	0	1	Channel has completed the major loop and is idle

For both activation types, the major-loop-complete status is explicitly indicated via the TCD*n*_CSR[DONE] bit.

The TCD*n*_CSR[START] bit is cleared automatically when the channel begins execution regardless of how the channel activates.

14.6.5.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCD*n*_SADDR, TCD*n*_DADDR, and TCD*n*_NBYTES values if read while a channel executes. The true values of the SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file and not the values in the TCD local memory for that channel. The addresses, SADDR and DADDR, and NBYTES, which decrement to zero as the transfer progresses, can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

14.6.5.3 Checking channel preemption status

Preemption is available only when fixed arbitration is selected for both group and channel arbitration modes. A preemptive situation is one in which a preempt-enabled channel runs and a higher priority request becomes active. When the eDMA engine is not operating in fixed group, fixed channel arbitration mode, the determination of the actively running relative priority outstanding requests become undefined. Channel and/or group priorities are treated as equal, that is, constantly rotating, when Round-Robin Arbitration mode is selected.

The TCD*n*_CSR[ACTIVE] bit for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended while the preempting channel executes one major loop iteration. If two TCD*n*_CSR[ACTIVE] bits are set simultaneously in the global TCD map, a higher priority channel is actively preempting a lower priority channel.

14.6.6 Channel Linking

Channel linking (or chaining) is a mechanism where one channel sets the TCD*n*_CSR[START] bit of another channel (or itself), therefore initiating a service request for that channel. When properly enabled, the EDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCD*n*_CITER[E_LINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, the initial fields of:

```
TCDn_CITER[E_LINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJOR_E_LINK] = 1
TCDn_CSR[MAJOR_LINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] bit
2. Minor loop done → set TCD12_CSR[START] bit
3. Minor loop done → set TCD12_CSR[START] bit
4. Minor loop done, major loop done→ set TCD7_CSR[START] bit

When minor loop linking is enabled (TCD*n*_CITER[E_LINK] = 1), the TCD*n*_CITER[CITER] field uses a nine bit vector to form the current iteration count. When minor loop linking is disabled (TCD*n*_CITER[E_LINK] = 0), the TCD*n*_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCD*n*_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

Note

The TCD*n*_CITER[E_LINK] bit and the TCD*n*_BITER[E_LINK] bit must equal or a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop, half-way done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, i.e, use another channel's TCD, at the end of a loop.

Table 14-12. Channel Linking Parameters

Desired Link Behavior	TCD Control Field Name	Description
Link at end of Minor Loop	CITER[E_LINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of Major Loop	CSR[MAJOR_E_LINK]	Enable channel-to-channel linking on major loop completion
	CSR[MAJOR_LINKCH]	Link channel number when linking at end of major loop

14.6.7 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

14.6.7.1 Dynamically changing the channel priority

The following two options are recommended for dynamically changing channel priority levels:

1. Switch to Round-Robin Channel Arbitration mode, change the channel priorities, then switch back to Fixed Arbitration mode,
2. Disable all the channels, change the channel priorities, then enable the appropriate channels.

14.6.7.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCDn_CSR[MAJORELINK] bit during channel execution (see the diagram in TCD structure). This bit is read from the TCD local memory at the end of channel execution, thus allowing the user to enable the feature during channel execution.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic channel link by enabling the TCDn_CSR[MAJORELINK] bit at the same time the eDMA engine is retiring the channel. The TCDn_CSR[MAJORELINK] would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

The following coherency model is recommended when executing a dynamic channel link request.

1. Write 1 to the TCDn_CSR[MAJORELINK] bit.
2. Read back the TCDn_CSR[MAJORELINK] bit.
3. Test the TCDn_CSR[MAJORELINK] request status:
   - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
   - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCDn_CSR[MAJORELINK] bit to zero on any writes to a channel's TCD.word7 after that channel's TCD.done bit is set, indicating the major loop is complete.

NOTE

The user must clear the The TCDn_CSR[DONE] bit before writing the TCDn_CSR[MAJORELINK] bit. The TCDn_CSR[DONE] bit is cleared automatically by the eDMA engine after a channel begins execution.

14.6.7.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources. When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in eDMA programmer's model, thus replacing the current descriptor.

Because the user is allowed to change the configuration during execution, a coherency model is needed. Consider the scenario where the user attempts to execute a dynamic scatter/gather operation by enabling the TCDn_CSR[ESG] bit at the same time the eDMA engine is retiring the channel. The ESG bit would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods for this coherency model are shown in the following subsections. Method 1 has the advantage of reading the MAJORLINKCH field and the ESG bit with a single read. For both dynamic channel linking and scatter/gather requests, the TCD local memory controller forces the TCD MAJOR.E_LINK and E_SG bits to zero on any writes to a channel's TCD word 7 if that channel's TCD.DONE bit is set indicating the major loop is complete.

NOTE

The user must clear the TCDn_CSR[DONE] bit before writing the MAJORELINK or ESG bits. The TCDn_CSR[DONE] bit is cleared automatically by the eDMA engine after a channel begins execution.

14.6.7.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCDn_CSR[MAJORELINK] bit is zero, the TCDn_CSR[MAJORLINKCH] field is not used by the eDMA. In this case, the MAJORLINKCH field may be used for other purposes. This method uses the MAJORLINKCH field as a TCD identification (ID).

1. When the descriptors are built, write a unique TCD ID in the TCDn_CSR[MAJORLINKCH] field for each TCD associated with a channel using dynamic scatter/gather.
2. Write 1b to the TCDn_CSR[DREQ] bit.

   Should a dynamic scatter/gather attempt fail, setting the DREQ bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (DADDR) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offset value.
3. Write the TCDn_DLASTSGA register with the scatter/gather address.
4. Write 1b to the TCDn_CSR[ESG] bit.

5. Read back the 16 bit TCD control/status field.
6. Test the ESG request status and MAJORLINKCH value in the TCDn_CSR register:

   If ESG = 1b, the dynamic link attempt was successful.

   If ESG = 0b and the MAJORLINKCH (ID) did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If ESG = 0b and the MAJORLINKCH (ID) changed, the dynamic link attempt was successful (the new TCD's E_SG value cleared the ESG bit).

14.6.7.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCD.DLAST_SGA field as a TCD identification (ID).

1. Write 1b to the TCDn_CSR[DREQ] bit.

   Should a dynamic scatter/gather attempt fail, setting the DREQ bit will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (DADDR) that was calculated using a scatter/gather address (written in the next step) instead of a dlast final offset value.
2. Write the TCDn_DLASTSGA register with the scatter/gather address.
3. Write 1b to the TCDn_CSR[ESG] bit.
4. Read back the ESG bit.
5. Test the ESG request status:

   If ESG = 1b, the dynamic link attempt was successful.

   If ESG = 0b, read the 32 bit TCDn_DLASTSGA field.

   If ESG = 0b and the TCDn_DLASTSGA did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If ESG = 0b and the TCDn_DLASTSGA changed, the dynamic link attempt was successful (the new TCD's E_SG value cleared the ESG bit).

14.6.8 Suspend/resume a DMA channel with active hardware service requests

The DMA allows the user to move data from memory or peripheral registers to another location in memory or peripheral registers without CPU interaction. Once the DMA and peripherals have been configured and are active, it is rare to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, a specific procedure must be followed. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (SPI), ADC, or other module.

14.6.8.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status Register (DMA_HRSn) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ bit on appropriate DMA channel.

14.6.8.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting the its ERQ bit.
2. Enable the DMA service request at the peripheral.

For example, assume the SPI is set as a master for transmitting data via a DMA service request when the SPI_TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If the user needs to suspend the DMA/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to SPI_RSER[TFFF_RE]. Confirm that SPI_RSER[TFFF_RE] is 0.
2. Ensure there is no DMA service request from the SPI by verifying that DMA_HRS[HRS*n*] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ bit. If a service request is present, wait until the request has been processed and the HRS bit reads zero.

Chapter 15
Direct Memory Access Multiplexer (DMAMUX)

15.1 Chip-specific DMAMUX information

Table 15-1. Reference links to related information

Topic	Related module	Reference
Full description	DMAMUX	DMAMUX
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

15.1.1 DMAMUX

The direct memory access multiplexer (DMAMUX) routes DMA sources, called slots, to any of the supported DMA channels. There is one 128x32 DMAMUX instance on DMA0 and one 128x32 DMAMUX instance on DMA1. The following table shows the configuration of the DMAMUX.

Table 15-2. DMAMUX configuration

Parameter	Description
Name	Direct Memory Access Multiplexer (DMAMUX)
Instances	2
Configurable features	• DMA0 DMAMUX: 128x32 • DMA1 DMAMUX: 128x32
Interface speed	NA
External I/O pins	NA

15.1.2 DMA0 request assignment

The following table shows DMA requests mapped into DMA0 channels

Table 15-3. DMAMUX0 channel assignments

M4 DMAMUX0 (128-to-32)		
Source/Slot Number	**Source Module**	**Source Description**
0	-	Channel disabled
1	QSPI	Receive
2	QSPI	Transmit
3	LTC	Low Power Trusted Cryptography RX FIFO
4	LTC	Low Power Trusted Cryptography TX FIFO
5	LPTMR0	Low Power Timer
6	LPTMR1	Low Power Timer
7	TPM0	Channel 0
8	TPM0	Channel 1
9	TPM0	Channel 2
10	TPM0	Channel 3
11	TPM0	Channel 4
12	TPM0	Channel 5
13	TPM0	Overflow
14	TPM1	Channel 0
15	TPM1	Channel 1
16	TPM1	Overflow
17	TPM2	Channel 0
18	TPM2	Channel 1
19	TPM2	Overflow
20	TPM3	Channel 0
21	TPM3	Channel 1
22	TPM3	Channel 2
23	TPM3	Channel 3
24	TPM3	Channel 4
25	TPM3	Channel 5
26	TPM3	Overflow
27	FlexIO0	Shifter 0
28	FlexIO0	Shifter 1
29	FlexIO0	Shifter 2
30	FlexIO0	Shifter 3
31	FlexIO0	Shifter 4
32	FlexIO0	Shifter 5

Table continues on the next page...

Table 15-3. DMAMUX0 channel assignments (continued)

M4 DMAMUX0 (128-to-32)		
Source/Slot Number	**Source Module**	**Source Description**
33	FlexIO0	Shifter 6
34	FlexIO0	Shifter 7
35	LPI2C0	Master/Slave Receive
36	LPI2C0	Master/Slave Transmit
37	LPI2C1	Master/Slave Receive
38	LPI2C1	Master/Slave Transmit
39	LPI2C2	Master/Slave Receive
40	LPI2C2	Master/Slave Transmit
41	LPI2C3	Master/Slave Receive
42	LPI2C3	Master/Slave Transmit
43	SAI0	Receive
44	SAI0	Transmit
45	SAI1	Receive
46	SAI1	Transmit
47	LPSPI0	Receive
48	LPSPI0	Transmit
49	LPSPI1	Receive
50	LPSPI1	Transmit
51	LPUART0	Receive
52	LPUART0	Transmit
53	LPUART1	Receive
54	LPUART1	Transmit
55	LPUART2	Receive
56	LPUART2	Transmit
57	LPUART3	Receive
58	LPUART3	Transmit
59	Reserved	Reserved
60	PCTLA	Port A pin request
61	PCTLB	Port B pin request
62	ADC0	Conversion Complete
63	ADC1	Conversion Complete
64	CMP0	Comparison Event
65	CMP1	Comparison Event
66	DAC0	DAC Request
67	DAC1	DAC Request
68-121	Reserved	Reserved
122	DMA MUX	Reserved
123	DMA MUX	Reserved

Table continues on the next page...

Table 15-3. DMAMUX0 channel assignments (continued)

M4 DMAMUX0 (128-to-32)		
Source/Slot Number	**Source Module**	**Source Description**
124	DMA MUX	Reserved
125	DMA MUX	Reserved
126	DMA MUX	Reserved
127	DMA MUX	Reserved

15.1.3 DMA1 request assignments

The following table shows DMA requests mapped into DMA1 channels.

Table 15-4. A7 DMAMUX1 assignments

Source/Slot number	DMAMUX1 (128-to-32)	
0	–	Channel disabled
1	FlexIO1	Shifter 0
2	FlexIO1	Shifter 1
3	FlexIO1	Shifter 2
4	FlexIO1	Shifter 3
5	FlexIO1	Shifter 4
6	FlexIO1	Shifter 5
7	FlexIO1	Shifter 6
8	FlexIO1	Shifter 7
9	LPI2C4	Master/Slave Receive
10	LPI2C4	Master/Slave Transmit
11	LPI2C5	Master/Slave Receive
12	LPI2C5	Master/Slave Transmit
13	LPI2C6	Master/Slave Receive
14	LPI2C6	Master/Slave Transmit
15	LPI2C7	Master/Slave Receive
16	LPI2C7	Master/Slave Transmit
17	LPUART4	Receive
18	LPUART4	Transmit
19	LPUART5	Receive
20	LPUART5	Transmit
21	LPUART6	Receive
22	LPUART6	Transmit
23	LPUART7	Receive
24	LPUART7	Transmit

Table continues on the next page...

Table 15-4. A7 DMAMUX1 assignments (continued)

Source/Slot number	DMAMUX1 (128-to-32)	
25	LPSPI2	Receive
26	LPSPI2	Transmit
27	LPSPI3	Receive
28	LPSPI3	Transmit
29	TPM4	Channel 0
30	TPM4	Channel 1
31	TPM4	Channel 2
32	TPM4	Channel 3
33	TPM4	Channel 4
34	TPM4	Channel 5
35	TPM4	Overflow
36	TPM5	Channel 0
37	TPM5	Channel 1
38	TPM5	Overflow
39	TPM6	Channel 3
40	TPM6	Channel 4
41	TPM6	Overflow
42	TPM7	Channel 0
43	TPM7	Channel 1
44	TPM7	Channel 2
45	TPM7	Channel 3
46	TPM7	Channel 4
47	TPM7	Channel 5
48	TPM7	Overflow
49	–	–
50	–	–
51	PCTLC	Port Interrupt Control module C
52	PCTLD	Port Interrupt Control module D
53	PCTLE	Port Interrupt Control module E
54	PCTLF	Port Interrupt Control module F
55	–	–
56	–	–
57	QSPI	Receive
58	QSPI	Transmit
59	SAI0	Receive
60	SAI0	Transmit
61	SAI1	Receive
62	SAI1	Transmit
63	PCTLA	Port Interrupt Control module A

Table continues on the next page...

Table 15-4. A7 DMAMUX1 assignments (continued)

Source/Slot number	DMAMUX1 (128-to-32)	
64	PCTLB	Port Interrupt Control module B
65-127	–	Channel disabled

15.2 Introduction

15.2.1 Overview

The Direct Memory Access Multiplexer (DMAMUX) routes DMA sources, called slots, to any of the 32 DMA channels. This process is illustrated in the following figure.

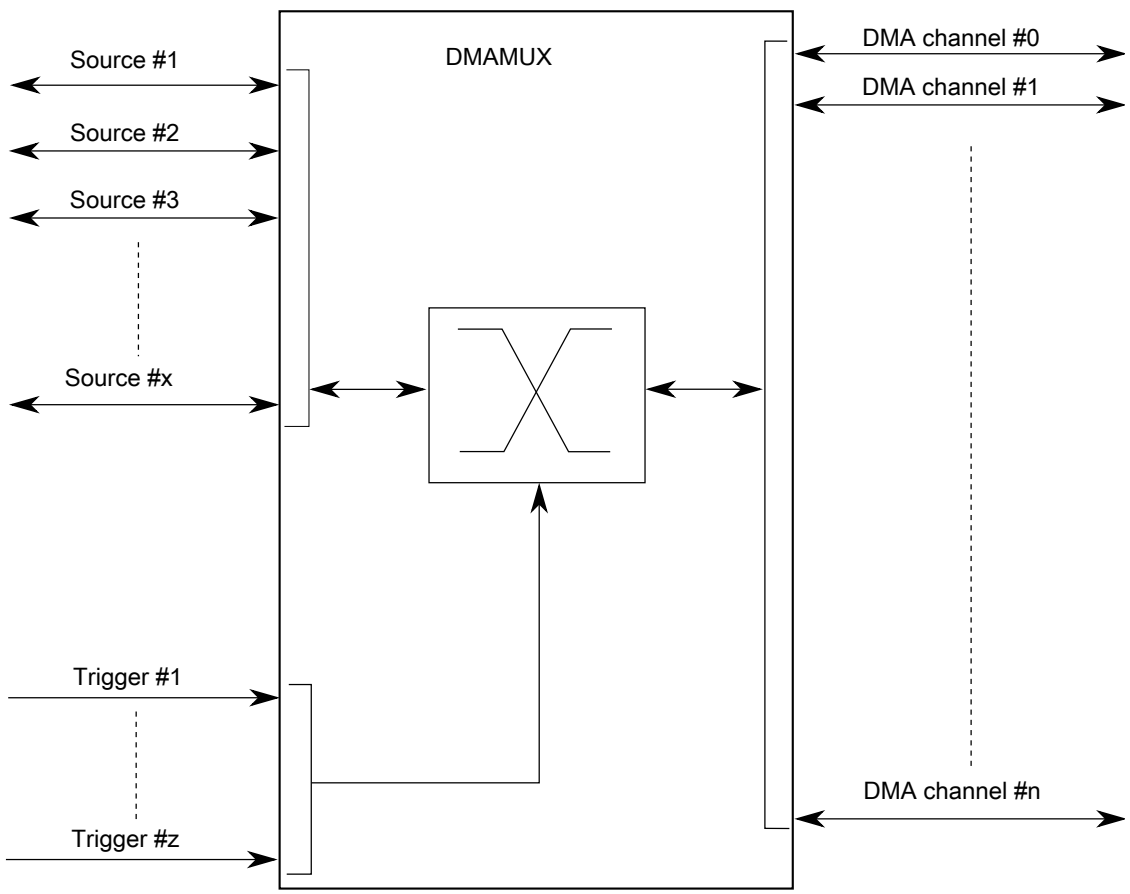


Figure 15-1. DMAMUX block diagram

15.2.2 Features

The DMAMUX module provides these features:

- Up to 128 peripheral slots can be routed to 32 channels.
- 32 independently selectable DMA channel routers.
  - Each channel output can be individually configured to be Always On and not depend on any of the peripheral slots.
  - The first 8 channels additionally provide a trigger functionality.
- Each channel router can be assigned to one of the possible peripheral DMA slots.
- On every memory map configuration change for a any channel, this module signals to the DMA Controller to reset the internal state machine for that channel and it can accept a new request based on the new configuration.

15.2.3 Modes of operation

The following operating modes are available:

- Disabled mode

  In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place, for example, changing the period of a DMA trigger.

- Normal mode

  In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMAMUX in this mode is completely transparent to the system.

- Periodic Trigger mode

  In this mode, a DMA source may only request a DMA transfer, such as when a transmit buffer becomes empty or a receive buffer becomes full, periodically.

  Configuration of the period is done in the registers of the periodic interrupt timer (PIT). This mode is available only for channels 0 to 7.

15.3 External signal description

The DMAMUX has no external pins.

15.4 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

15.4.1 DMA_CH_MUX register descriptions

15.4.1.1 DMA_CH_MUX Memory map

DMA_CH_MUX0 base address: 4102_0000h

DMA_CH_MUX1 base address: 4021_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 7Ch	Channel a Configuration Register (CHCFG0 - CHCFG31)	32	RW	0000_0000h

15.4.1.2 Channel a Configuration Register (CHCFG0 - CHCFG31)

15.4.1.2.1 Offset

For a = 0 to 31:

Register	Offset
CHCFGa	0h + (a × 4h)

15.4.1.2.2 Function

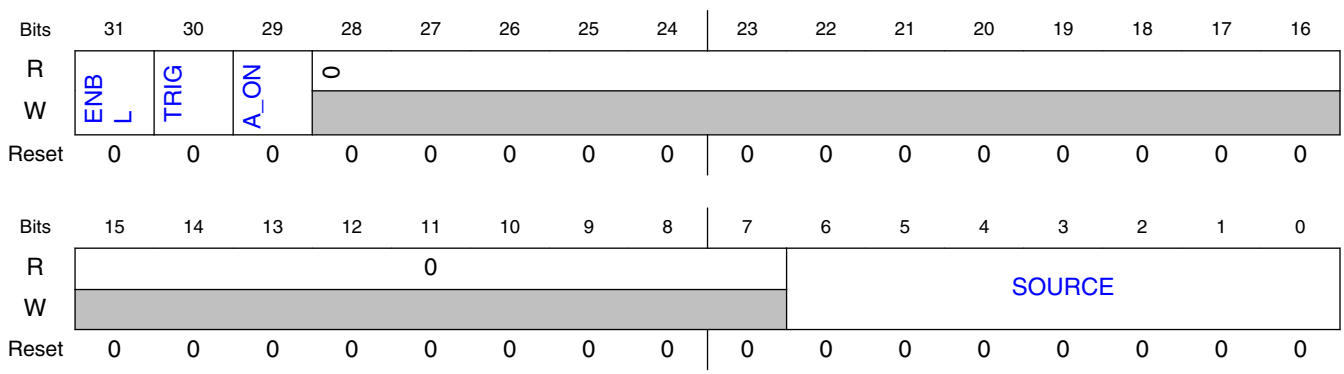
Each of the DMA channels can be independently enabled/disabled and associated with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

Setting multiple CHCFG registers with the same source value will result in unpredictable behavior. This is true, even if a channel is disabled (ENBL==0).

Before changing the trigger or source settings, a DMA channel must be disabled via CHCFGn[ENBL].

15.4.1.2.3 Diagram



15.4.1.2.4 Fields

Field	Function
31 ENBL	DMA Mux Channel Enable Enables the channel for DMA Mux. The DMA has separate channel enables/disables, which should be used to disable or reconfigure a DMA channel. 0b - DMA Mux channel is disabled 1b - DMA Mux channel is enabled
30 TRIG	DMA Channel Trigger Enable Enables the periodic trigger capability for the triggered DMA channel. 0b - Triggering is disabled. If triggering is disabled and ENBL is set, the DMA Channel will simply route the specified source to the DMA channel. (Normal mode) 1b - Triggering is enabled. If triggering is enabled and ENBL is set, the DMA_CH_MUX is in Periodic Trigger mode.
29 A_ON	DMA Channel Always Enable Enables the DMA Channel to be always ON. If TRIG bit is set, the module will assert request on every trigger. 0b - DMA Channel Always ON function is disabled

Table continues on the next page...

Field	Function
	1b - DMA Channel Always ON function is enabled
28-7 —	Reserved field
6-0 SOURCE	DMA Channel Source (Slot Number) Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMA_CH_MUX information for details about the peripherals and their slot numbers.

15.5 Functional description

The primary purpose of the DMAMUX is to provide flexibility in the system's use of the available DMA channels.

As such, configuration of the DMAMUX is intended to be a static procedure done during execution of the system boot code. However, if the procedure outlined in Enabling and configuring sources is followed, the configuration of the DMAMUX may be changed during the normal operation of the system.

Functionally, the DMAMUX channels may be divided into two classes:

- Channels that implement the normal routing functionality plus periodic triggering capability
- Channels that implement only the normal routing functionality

15.5.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first 8 channels of the DMAMUX provide a special periodic triggering capability that can be used to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention.

The trigger is generated by the periodic interrupt timer (PIT); as such, the configuration of the periodic triggering interval is done via configuration registers in the PIT. See the section on periodic interrupt timer for more information on this topic.

Note

Because of the dynamic nature of the system (due to DMA channel priorities, bus arbitration, interrupt service routine lengths, etc.), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

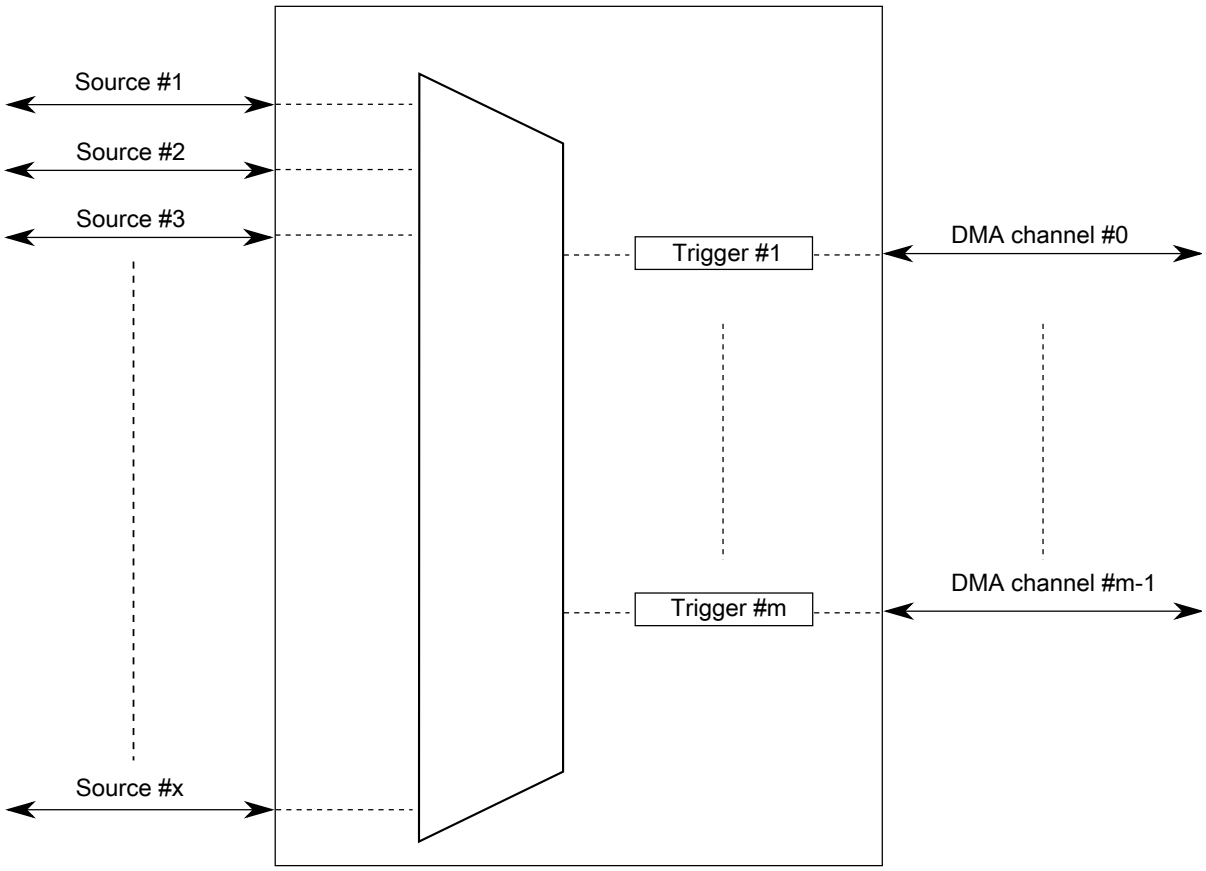


Figure 15-2. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to the DMA until a trigger event has been seen. This is illustrated in the following figure.

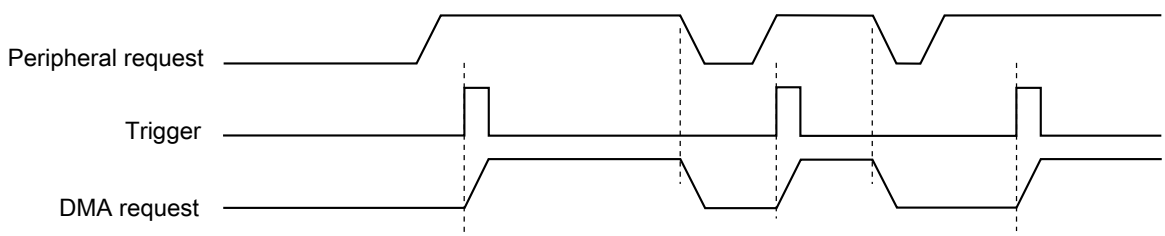


Figure 15-3. DMAMUX channel triggering: normal operation

After the DMA request has been serviced, the peripheral will negate its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger will be ignored. This situation is illustrated in the following figure.

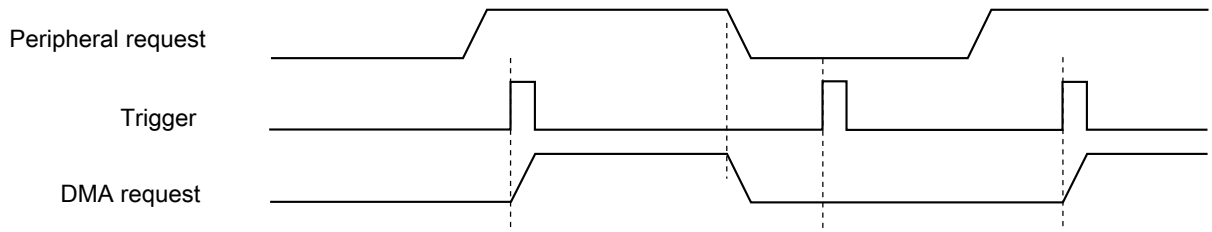


Figure 15-4. DMAMUX channel triggering: ignored trigger

This triggering capability may be used with any peripheral that supports DMA transfers, and is most useful for two types of situations:

- Periodically polling external devices on a particular bus

  As an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described above. After it has been set up, the SPI will request DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of the SPI, the SPI and DMA can be configured to transfer receive data into memory, effectively implementing a method to periodically read data from external devices and transfer the results into memory without processor intervention.

- Using the GPIO ports to drive or sample waveforms

  By configuring the DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using the DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in tabular form in on-chip memory.

A more detailed description of the capability of each trigger, including resolution, range of values, and so on, may be found in the periodic interrupt timer section.

15.5.2 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, each DMA Channel can be individually configured to function as an Always Enabled source. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the always enabled channel provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins, either unthrottled (that is, as fast as possible), or periodically (using the DMA triggering capability).
- Performing DMA transfers from memory to memory—Moving data from memory to memory, typically as fast as possible, sometimes with software activation.
- Performing DMA transfers from memory to the external bus, or vice-versa—Similar to memory to memory transfers, this is typically done as quickly as possible.
- Any DMA transfer that requires software activation—Any DMA transfer that should be explicitly started by software.

In cases where software should initiate the start of a DMA transfer, an always-enabled DMA channel can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation, or a transfer request from the DMA channel MUX. The options for doing this are:

- Transfer all data in a single minor loop.

  By configuring the DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will impose on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use explicit software reactivation.

  In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers *after every minor loop*. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use an always-enabled DMA source.

  In this option, the DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, the DMA channel should be enabled and configured as "always enabled" channel. Note that the reactivation of the channel can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of packets of data from one source to another, without processor intervention.

NOTE

When a channel is configured as "Always Enabled", then the peripheral DMA sources for that channel are ignored; i.e. SOURCE field has no effect.

15.6 Initialization/application information

This section provides instructions for initializing the DMA channel MUX.

15.6.1 Reset

The reset state of each individual bit is shown in Memory map/register definition. In summary, after reset, all channels are disabled and must be explicitly enabled before use.

15.6.2 Enabling and configuring sources

To enable a source with periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 8 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

NOTE

The following is an example. See the Features section to know the number of DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0x00000000 to CHCFG1.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write 0xC0000005 to CHCFG1.

The following code example illustrates steps 1 and 4 above:

```
void DMAMUX_Init(uint8_t DMA_CH, uint8_t DMAMUX_SOURCE)
{
    DMAMUX_0.CHCFG[DMA_CH].B.SOURCE = DMAMUX_SOURCE;
    DMAMUX_0.CHCFG[DMA_CH].B.ENBL   = 1;
    DMAMUX_0.CHCFG[DMA_CH].B.TRIG   = 1;
}
```

To enable a source without periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 8 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG[ENBL] is setwhile CHCFG[TRIG] is cleared.

NOTE

The following is an example. See the Features section to know the number of DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1 with no periodic triggering capability :

1. Write 0x00000000 to CHCFG1.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Write 0x80000005 to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes long is 32-bits */
volatile unsigned long *CHCFG0 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned long *CHCFG1 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned long *CHCFG2 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned long *CHCFG3 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned long *CHCFG4 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0010);
volatile unsigned long *CHCFG5 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0014);
volatile unsigned long *CHCFG6 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0018);
volatile unsigned long *CHCFG7 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x001C);
volatile unsigned long *CHCFG8 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0020);
volatile unsigned long *CHCFG9 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0024);
volatile unsigned long *CHCFG10= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0028);
volatile unsigned long *CHCFG11= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x002C);
volatile unsigned long *CHCFG12= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0030);
volatile unsigned long *CHCFG13= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0034);
volatile unsigned long *CHCFG14= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0038);
volatile unsigned long *CHCFG15= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x003C);

In File main.c:
#include "registers.h"
:
:
```

```
*CHCFG1 = 0x00000000;
*CHCFG1 = 0x80000005;
```

To disable a source:

A particular DMA source may be disabled by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configuration may be necessary. See the appropriate section for more details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

To switch DMA channel 8 from source #5 transmit to source #7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability.
2. Write 0x00000000 to CHCFG8.
3. Write 0x80000007 to CHCFG8. (In this example, setting CHCFG[TRIG] would have no effect due to the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes long is 32-bits */
volatile unsigned long *CHCFG0 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned long *CHCFG1 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned long *CHCFG2 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned long *CHCFG3 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned long *CHCFG4 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0010);
volatile unsigned long *CHCFG5 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0014);
volatile unsigned long *CHCFG6 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0018);
volatile unsigned long *CHCFG7 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x001C);
volatile unsigned long *CHCFG8 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0020);
volatile unsigned long *CHCFG9 = (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0024);
volatile unsigned long *CHCFG10= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0028);
volatile unsigned long *CHCFG11= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x002C);
volatile unsigned long *CHCFG12= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0030);
volatile unsigned long *CHCFG13= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0034);
volatile unsigned long *CHCFG14= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x0038);
volatile unsigned long *CHCFG15= (volatile unsigned long *) (DMAMUX_BASE_ADDR+0x003C);

In File main.c:
#include "registers.h"
:
:
*CHCFG8 = 0x00000000;
*CHCFG8 = 0x80000007;
```

15.6.2.1 Configuration options

Table 15-5. Channel Configuration Options

ENBL	TRIG	A_ON	Function	Mode
0	X	X	DMA channel is disabled	Disabled Mode
1	0	0	DMA channel is enabled with no triggering (transparent)	Normal Mode
1	1	0	DMA channel is enabled with triggering	Periodic Trigger Mode
1	0	1	DMA channel is always enabled	Always On Mode
1	1	1	DMA channel is always enabled with triggering	Always On Trigger Mode

Chapter 16
AIPS-Lite

16.1 Chip-specific AIPS-Lite information

Table 16-1. Reference links to related information

Topic	Related module	Reference
Full description	AIPS-Lite	AIPS-Lite
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

16.1.1 AHB-Lite to IP Bus Interface (AIPS-Lite)

The AIPS-Lite module interfaces an AHB-Lite bus to the peripheral bus. There are 2 AIPS-lite instances in this device. Each AIPS module occupies a total of 64 MB of address space. In order to support multicore enhancements, additional side-band signals that are needed to communicate the domain attributes. The following table shows the configuration of the AIPS-Lite.

Table 16-2. AIPS-Lite configuration

Parameter	Description
Name	AHB-Lite to IP Bus Interface (AIPS-Lite)
Instances	2
Configurable features	• AIPS0-1: 2x 32-bit AHB instances
Interface speed	NA
External I/O pins	NA

16.2 Introduction

The peripheral bridge converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

The peripheral bridge occupies 64 MB of the address space, which is divided into peripheral slots of 4 KB. (It might be possible that all the peripheral slots are not used. See the memory map chapter for details on slot assignments.) The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

16.2.1 Features

Key features of the peripheral bridge are:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width

16.2.2 General operation

The slave devices connected to the peripheral bridge are modules which contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge.

The register maps of the peripherals are located on 4-KB boundaries. Each peripheral is allocated one or more 4-KB block(s) of the memory map.

16.3 Memory map/register definition

The AIPS module(s) on this device do(es) not contain any user-programmable registers.

16.4 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus.

The peripheral bridge manages all transactions destined for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

16.4.1 Access support

Aligned and misaligned 32-bit, 16-bit, and byte accesses are supported for 32-bit peripherals. Misaligned accesses are supported to allow memory to be placed on the slave peripheral bus. Peripheral registers must not be misaligned, although no explicit checking is performed by the peripheral bridge. All accesses are performed with a single transfer.

All accesses to the peripheral slots must be sized less than or equal to the designated peripheral slot size. If an access is attempted that is larger than the targeted port, an error response is generated.

Chapter 17
Cross Bar Switch

17.1 Chip-specific AXBS information

Table 17-1. Reference links to related information

Topic	Related module	Reference
Full description	AXBS	AXBS
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

17.1.1 AHB Cross Bar Switch (AXBS)

The AXBS is a crossbar switch module that arbitrates between multiple AHB masters to grant access to downstream AHB slave devices. The crossbar enables multiple masters to connect to slaves and parallel processing. In order to support multicore enhancements, additional side-band signals are needed to communicate the domain attributes. The following table shows the configuration of the AXBS.

Table 17-2. AXBS configuration

Parameter	Description
Name	AHB Cross Bar Switch
Instances	2
Configurable features	See AHB crossbar switch for detail
Interface speed	NA
External I/O pins	NA

17.2 Introduction

The information found here provides information on the layout, configuration, and programming of the crossbar switch.

The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows up to four bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave.

17.2.1 Features

The crossbar switch includes these features:

- Symmetric crossbar bus switch implementation
  - Allows concurrent accesses from different masters to different slaves
- Up to single-clock 64-bit transfer
- Programmable configuration for fixed-priority or round-robin slave port arbitration (see the chip-specific information).
- 64-bit AHB crossbar bus switch compatible with ARM's AMBA Specification v2.0.

17.3 Functional Description

Information about general operation and arbitration can be found here.

17.3.1 General operation

When a master accesses the crossbar switch, the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master simply sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device has no knowledge of whether it actually owns the slave port it is targeting. While the master does not have control of the slave port it is targeting, it simply waits.

After the master has control of the slave port it is targeting, the master remains in control of the slave port until it relinquishes the slave port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it remains parked with the last master to use the slave port. This is done to save the initial clock of arbitration delay that otherwise would be seen if the same master had to arbitrate to gain control of the slave port.

17.3.2 Arbitration

The crossbar switch supports two arbitration algorithms:

- Fixed priority
- Round-robin

The selection of the global slave port arbitration algorithm is described in the crossbar switch chip-specific information.

17.3.2.1 Arbitration during undefined length bursts

Undefined length bursts can be interrupted.

17.3.2.2 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level with the highest numbered master having the highest priority (for example, in a system with 5 masters, master 1 has lower priority than master 3). If two masters request access to the same slave port, the master with the highest priority gains control over the slave port.

NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing accesses from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the proper master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port:

Table 17-3. How the Crossbar Switch grants control of a slave port to a master

When	Then the Crossbar Switch grants control to the requesting master
Both of the following are true: • The current master is not running a transfer. • The new requesting master's priority level is higher than that of the current master.	At the next clock edge
Both of the following are true: • The current master is running an undefined length burst transfer. • The requesting master's priority level is higher than that of the current master.	At the next arbitration point for the undefined length burst transfer
The requesting master's priority level is lower than the current master.	At the conclusion of one of the following cycles: • An IDLE cycle • A non-IDLE cycle to a location other than the current slave port

17.3.2.3 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requesting master becomes owner of the slave bus at the next transfer boundary. Priority is based on how far ahead the ID of the requesting master is to the ID of the last master.

After granted access to a slave port, a master may perform as many transfers as desired to that port until another master makes a request to the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and master 0, 4, and 5 make simultaneous requests, they are serviced in the order: 4 then 5 then 0.

The round-robin arbitration mode generally provides a more fair allocation of the available slave-port bandwidth (compared to fixed priority) as the fixed master priority does not affect the master selection.

17.4 Initialization/application information

No initialization is required for the crossbar switch.

Chapter 18
Memory Architecture

18.1 Introduction

This chapter presents memory hierarchical architecture and bus interconnection of the i.MX 7ULP microcontroller. This chapter also details bus master access latency to a variety of memory configurations.

18.2 Memory system

i.MX 7ULP memory system is designed for a balance of sufficient on-chip memory and efficient external memory needs. The architecture pays special attention to the support and implementation of memory system for the desired goals including performance, power and cost.

18.2.1 Memory hierarchy

The memory hierarchy is an arrangement of different types of memories with different capacities and operation speeds, to approximate the ideal memory behavior in a cost-efficient way.

A7 platform includes 32 KB instruction cache, 32 KB data cache in its level 1 memory and 256 KB level 2 cache, which is shared between instruction and data.

On the M4 side, level 1 memory consists of two SRAM blocks: 64 KB SRAMU and 192 KB SRAML, that the M4 processor can access in zero wait states. The SRAML block is further divided into two 32 KB and two 64 KB sub-blocks that allows each sub-block SRAM can be individually power-gated.

The bus system is connected so the SRAM memories on the M4 side can be accessed by the A7 processor as its level 3 memory. In addition, i.MX 7ULP integrates two SRAM blocks of 128 KB each, that either core can access as level 3 memory.

For external memory (like level 4 memory), i.MX 7ULP supports a variety of system memory configurations for application code, data storage, and execution space, including:

- LPDDR3/LPDDR2 via MMDC controller
- eMMC flash via uSDHC controller
- Serial flash via QSPI controller
- SRAM, PSRAM and parallel NOR via FlexBus controller

Chapter 19
System and Peripheral Memory Maps

19.1 System memory map

Table 19-1. System memory map

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0x0000_0000	0x0000_0FFF	0.00391	A7 Boot ROM (96KB)	M4 Boot ROM aliasing (4KB)
0x0001_1000	0x0001_7FFF	0.08984		Reserved
0x0001_8000	0x00FF_FFFF	15.90625	Reserved	Reserved
0x0100_0000	0x01FF_FFFF	16.00000	Reserved	Reserved
0x0200_0000	0x02FF_FFFF	16.00000	Reserved	Reserved
0x0300_0000	0x03FF_FFFF	16.00000		
0x0400_0000	0x04FF_FFFF	16.00000	Reserved	QuadSPI aliasing (128 MB)
0x0500_0000	0x05FF_FFFF	16.00000		
0x0600_0000	0x06FF_FFFF	16.00000		
0x0700_0000	0x07FF_FFFF	16.00000		
0x0800_0000	0x08FF_FFFF	16.00000		
0x0900_0000	0x09FF_FFFF	16.00000		
0x0A00_0000	0x0AFF_FFFF	16.00000		
0x0B00_0000	0x0BFF_FFFF	16.00000		
0x0C00_0000	0x0CFF_FFFF	16.00000	Reserved	Reserved
0x0D00_0000	0x0DFF_FFFF	16.00000		
0x0E00_0000	0x0EFF_FFFF	16.00000		
0x0F00_0000	0x0FFF_FFFF	16.00000		
0x1000_0000	0x10FF_FFFF	16.00000	Reserved	Reserved
0x1100_0000	0x11FF_FFFF	16.00000		
0x1200_0000	0x12FF_FFFF	16.00000		
0x1300_0000	0x13FF_FFFF	16.00000		
0x1400_0000	0x14FF_FFFF	16.00000	Reserved	Reserved
0x1500_0000	0x15FF_FFFF	16.00000		

Table continues on the next page...

Table 19-1. System memory map (continued)

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0x1600_0000	0x16FF_FFFF	16.00000		
0x1700_0000	0x17FF_FFFF	16.00000	Reserved	Reserved
0x1800_0000	0x18FF_FFFF	16.00000	Reserved	Reserved
0x1900_0000	0x19FF_FFFF	16.00000	Reserved	Reserved
0x1A00_0000	0x1AFF_FFFF	16.00000	Reserved	Reserved
0x1B00_0000	0x1BFF_FFFF	16.00000	Reserved	Reserved
0x1C00_0000	0x1C00_FFFF	0.06250	Reserved	Boot ROM (64 KB)
0x1C01_0000	0x1CFF_FFFF	15.93750	Reserved	Reserved
0x1D00_0000	0x1DFF_FFFF	16.00000	Reserved	Reserved
0x1E00_0000	0x1EFF_FFFF	16.00000	Reserved	Reserved
0x1F00_0000	0x1FFC_FFFF	15.81250	Reserved	Reserved
0x1FFD_0000	0x1FFF_FFFF	0.18750	SRAM (M4 SRAM)	SRAM_L (192 KB)
0x2000_0000	0x2000_FFFF	0.06250	SRAM (M4 SRAM)	SRAM_U(64 KB)
0x2001_0000	0x20FF_FFFF	15.93750	Reserved	Reserved
0x2100_0000	0x21FF_FFFF	16.00000	Reserved	Reserved
0x2200_0000	0x221F_FFFF	2.00000	Reserved	Bit Banding alias region for SRAM_U (2MB)
0x2220_0000	0x22FF_FFFF	14.00000	Reserved	Reserved
0x2300_0000	0x23FF_FFFF	16.00000	Reserved	Reserved
0x2400_0000	0x2400_007F	0.00012	QuadSPI Buffers (128 B)	QuadSPI Buffers (128 B)
0x2400_0080	0x24FF_FFFF	15.99988	Reserved	Reserved
0x2500_0000	0x25FF_FFFF	16.00000	Reserved	Reserved
0x2600_0000	0x2600_7FFF	0.03125	Secure SRAM (32 KB)	Secure SRAM (32 KB)
0x2600_8000	0x26FF_FFFF	15.96875	Reserved	Reserved
0x2700_0000	0x27FF_FFFF	16.00000	Reserved	Reserved
0x2800_0000	0x28FF_FFFF	16.00000	Reserved	Reserved
0x2900_0000	0x29FF_FFFF	16.00000	Reserved	Reserved
0x2A00_0000	0x2AFF_FFFF	16.00000	Reserved	Reserved
0x2B00_0000	0x2BFF_FFFF	16.00000	Reserved	Reserved
0x2C00_0000	0x2CFF_FFFF	16.00000	Reserved	Reserved
0x2D00_0000	0x2DFF_FFFF	16.00000	Reserved	Reserved
0x2E00_0000	0x2EFF_FFFF	16.00000	Reserved	Reserved
0x2F00_0000	0x2F00_FFFF	0.06250	SRAM0 (Always 128kB)	SRAM0 (Always 128kB)
0x2F01_0000	0x2F01_FFFF	0.06250		
0x2F02_0000	0x2F03_FFFF	0.12500	SRAM1 (128KB)	SRAM1 (128KB)
0x2F04_0000	0x2FFF_FFFF	15.75000	Reserved	Reserved
0x3000_0000	0x30FF_FFFF	16.00000	Reserved	Reserved
0x3100_0000	0x31FF_FFFF	16.00000	Reserved	Reserved

Table continues on the next page...

Table 19-1. System memory map (continued)

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0x3200_0000	0x32FF_FFFF	16.00000	Reserved	Reserved
0x3300_0000	0x33FF_FFFF	16.00000	Reserved	Reserved
0x3400_0000	0x34FF_FFFF	16.00000	Reserved	Reserved
0x3500_0000	0x35FF_FFFF	16.00000	Reserved	Reserved
0x3600_0000	0x36FF_FFFF	16.00000	Reserved	Reserved
0x3700_0000	0x37FF_FFFF	16.00000	Reserved	Reserved
0x3800_0000	0x38FF_FFFF	16.00000	Reserved	Reserved
0x3900_0000	0x39FF_FFFF	16.00000	Reserved	Reserved
0x3A00_0000	0x3AFF_FFFF	16.00000	Reserved	Reserved
0x3B00_0000	0x3BFF_FFFF	16.00000	Reserved	Reserved
0x3C00_0000	0x3CFF_FFFF	16.00000	Reserved	Reserved
0x3D00_0000	0x3DFF_FFFF	16.00000	Reserved	Reserved
0x3E00_0000	0x3EFF_FFFF	16.00000	Reserved	Reserved
0x3F00_0000	0x3FFF_FFFF	16.00000	Reserved	Reserved
0x4000_0000	0x407F_FFFF	8.00000	A7 AHB-PBridge0	A7 AHB-PBridge0
0x4080_0000	0x40FF_FFFF	8.00000	A7 AHB-PBridge1	A7 AHB-PBridge1
0x4100_0000	0x4107_FFFF	0.50000	M4 AIPS0	M4 AIPS0
0x4108_0000	0x410F_FFFF	0.50000	M4 AIPS1	M4 AIPS1
0x4110_0000	0x417F_FFFF	7.00000	Reserved	Reserved
0x4180_0000	0x418F_FFFF	0.50000	GPU-3D Programming Registers	GPU-3D Programming Registers
0x4188_0000	0x4187_FFFF	0.50000	GPU-2D Programming Registers	GPU-2D Programming Registers
0x4190_0000	0x41BF_FFFF	3.00000	Reserved	Reserved
0x41C0_0000	0x41CF_FFFF	1.00000	NIC0 Programming Registers	NIC0 Programming Registers
0x41D0_0000	0x41DF_FFFF	1.00000	NIC1 Programming Registers	NIC1 Programming Registers
0x41E0_0000	0x41FF_FFFF	2.00000	Reserved	Reserved
0x4200_0000	0x42FF_FFFF	16.00000	Reserved	Reserved
0x4300_0000	0x43FF_FFFF	16.00000	Reserved	Reserved
0x4400_0000	0x44FF_FFFF	16.00000	Reserved	BME (448 MB)
0x4500_0000	0x45FF_FFFF	16.00000	Reserved	
0x4600_0000	0x46FF_FFFF	16.00000	Reserved	
0x4700_0000	0x47FF_FFFF	16.00000	Reserved	
0x4800_0000	0x48FF_FFFF	16.00000	Reserved	
0x4900_0000	0x49FF_FFFF	16.00000	Reserved	
0x4A00_0000	0x4AFF_FFFF	16.00000	Reserved	
0x4B00_0000	0x4BFF_FFFF	16.00000	Reserved	
0x4C00_0000	0x4CFF_FFFF	16.00000	Reserved	

Table continues on the next page...

Table 19-1. System memory map (continued)

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0x4D00_0000	0x4DFF_FFFF	16.00000	Reserved	
0x4E00_0000	0x4EFF_FFFF	16.00000	Reserved	
0x4F00_0000	0x4FFF_FFFF	16.00000	Reserved	
0x5000_0000	0x50FF_FFFF	16.00000	Reserved	
0x5100_0000	0x51FF_FFFF	16.00000	Reserved	
0x5200_0000	0x52FF_FFFF	16.00000	Reserved	
0x5300_0000	0x53FF_FFFF	16.00000	Reserved	
0x5400_0000	0x54FF_FFFF	16.00000	Reserved	
0x5500_0000	0x55FF_FFFF	16.00000	Reserved	
0x5600_0000	0x56FF_FFFF	16.00000	Reserved	
0x5700_0000	0x57FF_FFFF	16.00000	Reserved	
0x5800_0000	0x58FF_FFFF	16.00000	Reserved	
0x5900_0000	0x59FF_FFFF	16.00000	Reserved	
0x5A00_0000	0x5AFF_FFFF	16.00000	Reserved	
0x5B00_0000	0x5BFF_FFFF	16.00000	Reserved	
0x5C00_0000	0x5CFF_FFFF	16.00000	Reserved	
0x5D00_0000	0x5DFF_FFFF	16.00000	Reserved	
0x5E00_0000	0x5EFF_FFFF	16.00000	Reserved	
0x5F00_0000	0x5FFF_FFFF	16.00000	Reserved	
0x6000_0000	0x60FF_FFFF	16.00000	DDR (1024 MB)	DDR (1024 MB)
0x6100_0000	0x61FF_FFFF	16.00000		
0x6200_0000	0x62FF_FFFF	16.00000		
0x6300_0000	0x63FF_FFFF	16.00000		
0x6400_0000	0x64FF_FFFF	16.00000		
0x6500_0000	0x65FF_FFFF	16.00000		
0x6600_0000	0x66FF_FFFF	16.00000		
0x6700_0000	0x67FF_FFFF	16.00000		
0x6800_0000	0x68FF_FFFF	16.00000		
0x6900_0000	0x69FF_FFFF	16.00000		
0x6A00_0000	0x6AFF_FFFF	16.00000		
0x6B00_0000	0x6BFF_FFFF	16.00000		
0x6C00_0000	0x6CFF_FFFF	16.00000		
0x6D00_0000	0x6DFF_FFFF	16.00000		
0x6E00_0000	0x6EFF_FFFF	16.00000		
0x6F00_0000	0x6FFF_FFFF	16.00000		
0x7000_0000	0x70FF_FFFF	16.00000		
0x7100_0000	0x71FF_FFFF	16.00000		
0x7200_0000	0x72FF_FFFF	16.00000		
0x7300_0000	0x73FF_FFFF	16.00000		

Table continues on the next page...

Table 19-1. System memory map (continued)

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0x7400_0000	0x74FF_FFFF	16.00000		
0x7500_0000	0x75FF_FFFF	16.00000		
0x7600_0000	0x76FF_FFFF	16.00000		
0x7700_0000	0x77FF_FFFF	16.00000		
0x7800_0000	0x78FF_FFFF	16.00000		
0x7900_0000	0x79FF_FFFF	16.00000		
0x7A00_0000	0x7AFF_FFFF	16.00000		
0x7B00_0000	0x7BFF_FFFF	16.00000		
0x7C00_0000	0x7CFF_FFFF	16.00000		
0x7D00_0000	0x7DFF_FFFF	16.00000		
0x7E00_0000	0x7EFF_FFFF	16.00000		
0x7F00_0000	0x7FFF_FFFF	16.00000		
0x8000_0000	0x80FF_FFFF	16.00000		
0x8100_0000	0x81FF_FFFF	16.00000		
0x8200_0000	0x82FF_FFFF	16.00000		
0x8300_0000	0x83FF_FFFF	16.00000		
0x8400_0000	0x84FF_FFFF	16.00000		
0x8500_0000	0x85FF_FFFF	16.00000		
0x8600_0000	0x86FF_FFFF	16.00000		
0x8700_0000	0x87FF_FFFF	16.00000		
0x8800_0000	0x88FF_FFFF	16.00000		
0x8900_0000	0x89FF_FFFF	16.00000		
0x8A00_0000	0x8AFF_FFFF	16.00000		
0x8B00_0000	0x8BFF_FFFF	16.00000		
0x8C00_0000	0x8CFF_FFFF	16.00000		
0x8D00_0000	0x8DFF_FFFF	16.00000		
0x8E00_0000	0x8EFF_FFFF	16.00000		
0x8F00_0000	0x8FFF_FFFF	16.00000		
0x9000_0000	0x90FF_FFFF	16.00000		
0x9100_0000	0x91FF_FFFF	16.00000		
0x9200_0000	0x92FF_FFFF	16.00000		
0x9300_0000	0x93FF_FFFF	16.00000		
0x9400_0000	0x94FF_FFFF	16.00000		
0x9500_0000	0x95FF_FFFF	16.00000		
0x9600_0000	0x96FF_FFFF	16.00000		
0x9700_0000	0x97FF_FFFF	16.00000		
0x9800_0000	0x98FF_FFFF	16.00000		
0x9900_0000	0x99FF_FFFF	16.00000		
0x9A00_0000	0x9AFF_FFFF	16.00000		

Table continues on the next page...

Table 19-1. System memory map (continued)

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0x9B00_0000	0x9BFF_FFFF	16.00000		
0x9C00_0000	0x9CFF_FFFF	16.00000		
0x9D00_0000	0x9DFF_FFFF	16.00000		
0x9E00_0000	0x9EFF_FFFF	16.00000		
0x9F00_0000	0x9FFF_FFFF	16.00000		
0xA000_0000	0xA0FF_FFFF	16.00000	Reserved	Reserved
0xA100_0000	0xA1FF_FFFF	16.00000		
0xA200_0000	0xA2FF_FFFF	16.00000		
0xA300_0000	0xA3FF_FFFF	16.00000		
0xA400_0000	0xA4FF_FFFF	16.00000		
0xA500_0000	0xA5FF_FFFF	16.00000		
0xA600_0000	0xA6FF_FFFF	16.00000		
0xA700_0000	0xA7FF_FFFF	16.00000		
0xA800_0000	0xA8FF_FFFF	16.00000		
0xA900_0000	0xA9FF_FFFF	16.00000		
0xAA00_0000	0xAAFF_FFFF	16.00000		
0xAB00_0000	0xABFF_FFFF	16.00000		
0xAC00_0000	0xACFF_FFFF	16.00000		
0xAD00_0000	0xADFF_FFFF	16.00000		
0xAE00_0000	0xAEFF_FFFF	16.00000		
0xAF00_0000	0xAFFF_FFFF	16.00000		
0xB000_0000	0xB0FF_FFFF	16.00000	FlexBus (256 MB)	FlexBus (256 MB)
0xB100_0000	0xB1FF_FFFF	16.00000		
0xB200_0000	0xB2FF_FFFF	16.00000		
0xB300_0000	0xB3FF_FFFF	16.00000		
0xB400_0000	0xB4FF_FFFF	16.00000		
0xB500_0000	0xB5FF_FFFF	16.00000		
0xB600_0000	0xB6FF_FFFF	16.00000		
0xB700_0000	0xB7FF_FFFF	16.00000		
0xB800_0000	0xB8FF_FFFF	16.00000		
0xB900_0000	0xB9FF_FFFF	16.00000		
0xBA00_0000	0xBAFF_FFFF	16.00000		
0xBB00_0000	0xBBFF_FFFF	16.00000		
0xBC00_0000	0xBCFF_FFFF	16.00000		
0xBD00_0000	0xBDFF_FFFF	16.00000		
0xBE00_0000	0xBEFF_FFFF	16.00000		
0xBF00_0000	0xBFFF_FFFF	16.00000		
0xC000_0000	0xC0FF_FFFF	16.00000	Quad SPI 0 (256 MB)	Quad SPI 0 (256 MB)
0xC100_0000	0xC1FF_FFFF	16.00000		

Table continues on the next page...

Table 19-1. System memory map (continued)

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0xC200_0000	0xC2FF_FFFF	16.00000		
0xC300_0000	0xC3FF_FFFF	16.00000		
0xC400_0000	0xC4FF_FFFF	16.00000		
0xC500_0000	0xC5FF_FFFF	16.00000		
0xC600_0000	0xC6FF_FFFF	16.00000		
0xC700_0000	0xC7FF_FFFF	16.00000		
0xC800_0000	0xC8FF_FFFF	16.00000		
0xC900_0000	0xC9FF_FFFF	16.00000		
0xCA00_0000	0xCAFF_FFFF	16.00000		
0xCB00_0000	0xCBFF_FFFF	16.00000		
0xCC00_0000	0xCCFF_FFFF	16.00000		
0xCD00_0000	0xCDFF_FFFF	16.00000		
0xCE00_0000	0xCEFF_FFFF	16.00000		
0xCF00_0000	0xCFFF_FFFF	16.00000		
0xD000_0000	0xD0FF_FFFF	16.00000	Reserved	Reserved
0xD100_0000	0xD1FF_FFFF	16.00000		
0xD200_0000	0xD2FF_FFFF	16.00000		
0xD300_0000	0xD3FF_FFFF	16.00000		
0xD400_0000	0xD4FF_FFFF	16.00000		
0xD500_0000	0xD5FF_FFFF	16.00000		
0xD600_0000	0xD6FF_FFFF	16.00000		
0xD700_0000	0xD7FF_FFFF	16.00000		
0xD800_0000	0xD8FF_FFFF	16.00000		
0xD900_0000	0xD9FF_FFFF	16.00000		
0xDA00_0000	0xDAFF_FFFF	16.00000		
0xDB00_0000	0xDBFF_FFFF	16.00000		
0xDC00_0000	0xDCFF_FFFF	16.00000		
0xDD00_0000	0xDDFF_FFFF	16.00000		
0xDE00_0000	0xDEFF_FFFF	16.00000		
0xDF00_0000	0xDFFF_FFFF	16.00000		
0xE000_0000	0xE007_FFFF	0.50000	Reserved	Reserved
0xE008_0000	0xE008_0FFF	0.00391	Reserved	MCM
0xE008_1000	0xE008_1FFF	0.00391	Reserved	MMCAU[1]
0xE008_2000	0xE008_2FFF	0.00391	Reserved	M4 Cache Controller
0xE008_3000	0xE0FF_FFFF	15.48828	Reserved	Reserved
0xE100_0000	0xE1FF_FFFF	16.00000	Reserved	Reserved
0xE200_0000	0xE2FF_FFFF	16.00000	Reserved	Reserved
0xE300_0000	0xE3FF_FFFF	16.00000	Reserved	Reserved
0xE400_0000	0xE4FF_FFFF	16.00000	Reserved	Reserved

Table continues on the next page...

Table 19-1. System memory map (continued)

Start address	End address	Size (MB)	Cortex-A7 Memory map	Cortex-M4 Memory map
0xE500_0000	0xE5FF_FFFF	16.00000		
0xE600_0000	0xE6FF_FFFF	16.00000		
0xE700_0000	0xE7FF_FFFF	16.00000		
0xE800_0000	0xE8FF_FFFF	16.00000	Reserved	Reserved
0xE900_0000	0xE9FF_FFFF	16.00000		
0xEA00_0000	0xEAFF_FFFF	16.00000		
0xEB00_0000	0xEBFF_FFFF	16.00000		
0xEC00_0000	0xECFF_FFFF	16.00000	Reserved	Reserved
0xED00_0000	0xEDFF_FFFF	16.00000	Reserved	Reserved
0xEE00_0000	0xEEFF_FFFF	16.00000	Reserved	Reserved
0xEF00_0000	0xEFFF_FFFF	16.00000	Reserved	Reserved
0xF000_0000	0xF00F_FFFF	1.00000	SYS PPB (Physical)	SYS PPB (Physical)
0xF010_0000	0xF0FF_FFFF	15.00000	Reserved	Reserved
0xF100_0000	0xF17F_FFFF	8.00000	Reserved	Reserved
0xF180_0000	0xF1FF_FFFF	8.00000	Reserved	Reserved
0xF200_0000	0xF2FF_FFFF	16.00000	Reserved	Reserved
0xF300_0000	0xF3FF_FFFF	16.00000	Reserved	Reserved
0xF400_0000	0xF4FF_FFFF	16.00000	Reserved	Reserved
0xF500_0000	0xF5FF_FFFF	16.00000	Reserved	Reserved
0xF600_0000	0xF6FF_FFFF	16.00000	Reserved	Reserved
0xF700_0000	0xF7FF_FFFF	16.00000	Reserved	Reserved
0xF800_0000	0xF80F_FFFF	1.00000	Reserved	Reserved
0xF810_0000	0xF8FF_FFFF	15.00000	Reserved	Reserved
0xF900_0000	0xF90F_FFFF	1.00000	Reserved	RT IOPORT
0xF910_0000	0xF9FF_FFFF	15.00000	Reserved	Reserved
0xFA00_0000	0xFAFF_FFFF	16.00000	Reserved	Reserved
0xFB00_0000	0xFBFF_FFFF	16.00000	Reserved	Reserved
0xFC00_0000	0xFCFF_FFFF	16.00000	Reserved	Reserved
0xFD00_0000	0xFDFF_FFFF	16.00000	Reserved	Reserved
0xFE00_0000	0xFEFF_FFFF	16.00000	Reserved	Reserved
0xFF00_0000	0xFFFF_FFFF	16.00000	Reserved	Reserved

1. This chapter is available only in the i.MX7ULP Security Reference Manual

19.2 Peripheral memory maps

19.2.1 M4 peripheral memory map

Table 19-2. M4 peripheral memory map - AIPS0 Assignments

Offset address	Slot	AIPS0	
		Module	Description
0x0_0000	0	Reserved	—
0x0_1000	1		
0x0_2000	2		
0x0_3000	3		
0x0_4000	4		
0x0_5000	5		
0x0_6000	6		
0x0_7000	7		
0x0_8000	8	eDMA0	Direct Memory Access Controller 0
0x0_9000	9	eDMA0 TCD	Direct Memory Access Controller 0 Transfer Control Descriptors
0x0_A000	10	Reserved	—
0x0_B000	11		
0x0_C000	12		
0x0_D000	13		
0x0_E000	14		
0x0_F000	15	RGPIO2P0	Rapid GPIO Controller 0 - Two Port
0x1_0000	16	Reserved	—
0x1_1000	17		
0x1_2000	18		
0x1_3000	19		
0x1_4000	20	XRDC	Extended Resource Domain Controller
0x1_5000	21	XRDC	Extended Resource Domain Controller
0x1_6000	22	XRDC	Extended Resource Domain Controller
0x1_7000	23	XRDC	Extended Resource Domain Controller
0x1_8000	24	Reserved	—
0x1_9000	25		
0x1_A000	26		
0x1_B000	27	SEMA42_0	Hardware Semaphore Module 0
0x1_C000	28	Reserved	—
0x1_D000	29		

Table continues on the next page...

Table 19-2. M4 peripheral memory map - AIPS0 Assignments (continued)

Offset address	Slot	AIPS0	
		Module	Description
0x1_E000	30		
0x1_F000	31		
0x2_0000	32	DMAMUX0	Direct Memory Access Multiplexer 0
0x2_1000	33	LLWU	Low-Leakage Wakeup Unit
0x2_2000	34	MU_A	Messaging Unit - Side A
0x2_3000	35	Reserved	Reserved
0x2_4000	36	TRGMUX0	Trigger MUX Control 0
0x2_5000	37	WDOG0	Watchdog Timer 0
0x2_6000	38	PCC0	Peripheral Clock Control 0
0x2_7000	39	SCG0	System Clock Generator
0x2_8000	40	Reserved	Reserved
0x2_9000	41	CRC	Cyclic Redundancy Check
0x2_A000	42	LTC	Low Power Trusted Cryptography
0x2_B000	43	Reserved	Reserved
0x2_C000	44	TRNG	True Random Number Generator
0x2_D000	45	LPIT0	Low Power Interrupt Timer
0x2_E000	46	LPTMR0	Low Power Timer 0
0x2_F000	47	LPTMR1	Low power Timer 1
0x3_0000	48	TPM0	Timer/PWM Module 0
0x3_1000	49	TPM1	Timer/PWM Module 1
0x3_2000	50	FlexIO0	Flexible IO Controller 0
0x3_3000	51	LPI2C0	Low Power Inter-Integrated Circuit 0
0x3_4000	52	LPI2C1	Low Power Inter-Integrated Circuit 1
0x3_5000	53	LPI2C2	Low Power Inter-Integrated Circuit 2
0x3_6000	54	LPI2C3	Low Power Inter-Integrated Circuit 3
0x3_7000	55	SAI0	Synchronous Audio Interface 0
0x3_8000	56	LPSPI0	Low Power Serial Peripheral Interface 0
0x3_9000	57	LPSPI1	Low Power Serial Peripheral Interface 1
0x3_A000	58	LPUART0	Low Power Universal Asynchronous Receiver/Transmitter 0
0x3_B000	59	LPUART1	Low Power Universal Asynchronous Receiver/Transmitter 1
0x3_C000	60	Reserved	Reserved
0x3_D000	61	IOMUXC0	Input/Output Multiplexing Controller 0
0x3_E000	62	Reserved	Reserved
0x3_F000	63	PCTLA	Port Interrupt Control A
0x4_0000	64	PCTLB	Port Interrupt Control B
0x4_1000	65	ADC0	Analog to Digital Converter 0
0x4_2000	66	CMP0	Analog Comparator 0

Table continues on the next page...

Table 19-2. M4 peripheral memory map - AIPS0 Assignments (continued)

Offset address	Slot	AIPS0	
		Module	Description
0x4_3000	67	CMP1	Analog Comparator 1
0x4_4000	68	DAC0	Digital to Analog Converter 0
0x4_5000	69	DAC1	Digital to Analog Converter 1
0x4_6000	70	Reserved	—
0x4_7000	71	Reserved	—
0x4_8000	72	Reserved	—
0x4_9000	73	Reserved	—
0x4_A000	74	Reserved	—
0x4_B000	75	Reserved	—
0x4_C000	76	Reserved	—
0x4_D000	77		
0x4_E000	78		
0x4_F000	79		
0x5_0000	80		
0x5_1000	81		
0x5_2000	82		
0x5_3000	83		
0x5_4000	84		
0x5_5000	85		
0x5_6000	86		
0x5_7000	87		
0x5_8000	88		
0x5_9000	89		
0x5_A000	90		
0x5_B000	91		
0x5_C000	92		
0x5_D000	93		
0x5_E000	94		
0x5_F000	95		
0x6_0000	96		
0x6_1000	97		
0x6_2000	98		
0x6_3000	99		
0x6_4000	100		
0x6_5000	101		
0x6_6000	102		
0x6_7000	103		
0x6_8000	104		

Table continues on the next page...

Table 19-2. M4 peripheral memory map - AIPS0 Assignments (continued)

Offset address	Slot	AIPS0	
		Module	Description
0x6_9000	105		
0x6_A000	106		
0x6_B000	107		
0x6_C000	108		
0x6_D000	109		
0x6_E000	110		
0x6_F000	111		
0x7_0000	112	SNVS	Secure Non-Volatile Storage (SNVS)
0x7_1000	113	Reserved	—
0x7_2000	114		
0x7_3000	115		
0x7_4000	116		
0x7_5000	117		
0x7_6000	118		
0x7_7000	119		
0x7_8000	120		
0x7_9000	121		
0x7_A000	122		
0x7_B000	123		
0x7_C000	124		
0x7_D000	125		
0x7_E000	126		
0x7_F000	127		

Table 19-3. M4 peripheral memory map - AIPS1

Offset address	Slot	AIPS1	
		Module	Description
0x0_0000	0	Reserved	—
0x0_1000	1		
0x0_2000	2		
0x0_3000	3		
0x0_4000	4		
0x0_5000	5		
0x0_6000	6		
0x0_7000	7		
0x0_8000	8		
0x0_9000	9		

Table continues on the next page...

Table 19-3. M4 peripheral memory map - AIPS1 (continued)

Offset address	Slot	AIPS1	
		Module	Description
0x0_A000	10		
0x0_B000	11		
0x0_C000	12		
0x0_D000	13		
0x0_E000	14		
0x0_F000	15		
0x1_0000	16	Reserved	Reserved
0x1_1000	17	DAPROM	Debug Access Port ROM
0x1_2000	18	FUNNEL	Coresight Funnel
0x1_3000	19	ETF	Embedded Trace FIFO
0x1_4000	20	TPIU	Trace Port Interface Unit
0x1_5000	21	ETR	Embedded Trace Router
0x1_6000	22	CTI	Cross Trigger Interface
0x1_7000	23	SWO	Serial Wire Output
0x1_8000	24	Timestamp Generator	Timestamp Generator
0x1_9000	25	Reserved	—
0x1_A000	26	A7 APB ROM	Cortex-A7 APB ROM
0x1_B000	27	A7 APB CPU DBG	Cortex-A7 APB CPU Debug
0x1_C000	28	A7 APB PMU	Cortex-A7 APB Performance Monitoring Unit
0x1_D000	29	A7 APB CTI	Cortex-A7 APB Cross Trigger Interface
0x1_E000	30	A7 APB ETM	Cortex-A7 APB Embedded Trace Module
0x1_F000	31	Reserved	—
0x2_0000	32	EWM	External Watchdog Monitor
0x2_1000	33	PMC0	Power Management Controller 0
0x2_2000	34	Reserved	—
0x2_3000	35	SIM	System Integration Module
0x2_4000	36	CMC0	Core Mode Controller 0
0x2_5000	37	QSPI and OTFAD[1]	Quad Serial Peripheral Interface And On-the-fly AES Decryptor
0x2_6000	38	OCOTP_CTRL	On-chip One Time Programmable Controller
0x2_7000	39	Reserved	—
0x2_8000	40	TPM2	Timer/PWM Module 2
0x2_9000	41	TPM3	Timer/PWM Module 3
0x2_A000	42	SAI1	Synchronous Audio Interface 1
0x2_B000	43	LPUART2	Low Power Universal Asynchronous Receiver/Transmitter 2
0x2_C000	44	LPUART3	Low Power Universal Asynchronous Receiver/Transmitter 3

Table continues on the next page...

Table 19-3. M4 peripheral memory map - AIPS1 (continued)

Offset address	Slot	AIPS1	
		Module	Description
0x2_D000	45	ADC1	Analog to Digital Converter 1
0x2_E000	46	Reserved	—
0x2_F000	47		
0x3_0000	48		
0x3_1000	49		
0x3_2000	50	PCC1	Peripheral Clock Control 1
0x3_3000	51	Reserved	—
0x3_4000	52		
0x3_5000	53		
0x3_6000	54		
0x3_7000	55		
0x3_8000	56		
0x3_9000	57		
0x3_A000	58		
0x3_B000	59		
0x3_C000	60		
0x3_D000	61		
0x3_E000	62		
0x3_F000	63		
0x4_0000	64		
0x4_1000	65		
0x4_2000	66		
0x4_3000	67		
0x4_4000	68		
0x4_5000	69		
0x4_6000	70		
0x4_7000	71		
0x4_8000	72		
0x4_9000	73		
0x4_A000	74		
0x4_B000	75		
0x4_C000	76		
0x4_D000	77		
0x4_E000	78		
0x4_F000	79		
0x5_0000	80		
0x5_1000	81		
0x5_2000	82		

Table continues on the next page...

Table 19-3. M4 peripheral memory map - AIPS1 (continued)

Offset address	Slot	AIPS1	
		Module	Description
0x5_3000	83		
0x5_4000	84		
0x5_5000	85		
0x5_6000	86		
0x5_7000	87		
0x5_8000	88		
0x5_9000	89		
0x5_A000	90		
0x5_B000	91		
0x5_C000	92		
0x5_D000	93		
0x5_E000	94		
0x5_F000	95		
0x6_0000	96		
0x6_1000	97		
0x6_2000	98		
0x6_3000	99		
0x6_4000	100		
0x6_5000	101		
0x6_6000	102		
0x6_7000	103		
0x6_8000	104		
0x6_9000	105		
0x6_A000	106		
0x6_B000	107		
0x6_C000	108		
0x6_D000	109		
0x6_E000	110		
0x6_F000	111		
0x7_0000	112		
0x7_1000	113		
0x7_2000	114		
0x7_3000	115		
0x7_4000	116		
0x7_5000	117		
0x7_6000	118		
0x7_7000	119		
0x7_8000	120		

Table continues on the next page...

Table 19-3. M4 peripheral memory map - AIPS1 (continued)

Offset address	Slot	AIPS1	
		Module	Description
0x7_9000	121		
0x7_A000	122		
0x7_B000	123		
0x7_C000	124		
0x7_D000	125		
0x7_E000	126		
0x7_F000	127		

1. See the i.MX 7ULP Security Reference Manual for the complete OTFAD chapter

19.2.2 A7 peripheral memory map

The following tables show the IP bus address space assigned for A7 peripherals.

Table 19-4. A7 Peripheral memory map - AHB-PBridge0

Offset address	Slot	AHB-PBridge0
0x0_0000	0	–
0x1_0000	1	–
0x2_0000	2	–
0x3_0000	3	–
0x4_0000	4	–
0x5_0000	5	–
0x6_0000	6	–
0x7_0000	7	–
0x8_0000	8	DMA Controller 1 (DMA1)
0x8_1000	9	DMA Descriptors 1 (DMA1 Descriptors)
0x9_0000	9	–
0xA_0000	10	–
0xB_0000	11	–
0xC_0000	12	–
0xD_0000	13	–
0xE_0000	14	–
0xF_0000	15	RGPIO2P
0x10_0000	16	FlexBus
0x11_0000	17	–
0x12_0000	18	–
0x13_0000	19	–

Table continues on the next page...

Table 19-4. A7 Peripheral memory map - AHB-PBridge0 (continued)

Offset address	Slot	AHB-PBridge0
0x14_0000	20	–
0x15_0000	21	–
0x16_0000	22	–
0x17_0000	23	–
0x18_0000	24	–
0x19_0000	25	–
0x1A_0000	26	–
0x1B_0000	27	Semaphore v2 1 (SEMA42_1)
0x1C_0000	28	–
0x1D_0000	29	–
0x1E_0000	30	–
0x1F_0000	31	–
0x20_0000	32	–
0x21_0000	33	DMA Multiplexer 1 (DMAMUX1)
0x22_0000	34	Message Unit Side B (MU_B)
0x23_0000	35	Reserved
0x24_0000	36	Cryptographic Acceleration and Assurance Module (CAAM)[1]
0x25_0000	37	Timer/PWM Module 4 (TPM4)
0x26_0000	38	Timer/PWM Module 5 (TPM5)
0x27_0000	39	Low Power Periodic Interrupt Timer 1 (LPIT1)
0x28_0000	40	–
0x29_0000	41	Low Power SPI 2 (LPSPI2)
0x2A_0000	42	Low Power SPI 3 (LPSPI3)
0x2B_0000	43	Low Power I2C 4 (LPI2C4)
0x2C_0000	44	Low Power I2C 5 (LPI2C5)
0x2D_0000	45	Low Power UART 4 (LPUART4)
0x2E_0000	46	Low Power UART 5 (LPUART5)
0x2F_0000	47	–
0x30_0000	48	–
0x31_0000	49	Flexible Input/Ouput 1 (FlexIO1)
0x32_0000	50	–
0x33_0000	51	High Speed On-The-Go USB 1 (USBOTG1)
0x34_0000	52	High Speed On-The-Go USB 2 (USBOTG2)
0x35_0000	53	USB-PHY Control
0x36_0000	54	USB PL301
0x37_0000	55	ultra Secure Digital Host Controller 0 (uSDHC0)

Table continues on the next page...

Table 19-4. A7 Peripheral memory map - AHB-PBridge0 (continued)

Offset address	Slot	AHB-PBridge0
0x38_0000	56	ultra Secure Digital Host Controller 1 (uSDHC1)
0x39_0000	57	–
0x3A_0000	58	Trigger Multiplexer 1 (TRGMUX1)
0x3B_0000	59	–
0x3C_0000	60	–
0x3D_0000	61	Watchdog 1 (WDOG1)
0x3E_0000	62	System Clock Generator 1 (SCG1)
0x3F_0000	63	Peripheral Clock Control 2 (PCC2)
0x40_0000	64	Power Management Control 1 (Digital PMC1)
0x41_0000	65	Core Mode Controller 1 (CMC1)
0x42_0000	66	–
0x43_0000	67	Watchdog2 (WDOG2)
0x44_0000	68	–
0x45_0000	69	–
0x46_0000	70	–
0x47_0000	71	–
0x48_0000	72	–
0x49_0000	73	–
0x4A_0000	74	–
0x4B_0000	75	–
0x4C_0000	76	–
0x4D_0000	77	–
0x4E_0000	78	–
0x4F_0000	79	–
0x50_0000	80	–
0x51_0000	81	–
0x52_0000	82	–
0x53_0000	83	–
0x54_0000	84	–
0x55_0000	85	–
0x56_0000	86	–
0x57_0000	87	–
0x58_0000	88	–
0x59_0000	89	–
0x5A_0000	90	–
0x5B_0000	91	–
0x5C_0000	92	–
0x5D_0000	93	–

Table continues on the next page...

Table 19-4. A7 Peripheral memory map - AHB-PBridge0 (continued)

Offset address	Slot	AHB-PBridge0
0x5E_0000	94	–
0x5F_0000	95	–
0x60_0000	96	–
0x61_0000	97	–
0x62_0000	98	–
0x63_0000	99	–
0x64_0000	100	–
0x65_0000	101	–
0x66_0000	102	–
0x67_0000	103	–
0x68_0000	104	–
0x69_0000	105	–
0x6A_0000	106	–
0x6B_0000	107	–
0x6C_0000	108	–
0x6D_0000	109	–
0x6E_0000	110	–
0x6F_0000	111	–
0x70_0000	112	–
0x71_0000	113	–
0x72_0000	114	–
0x73_0000	115	–
0x74_0000	116	–
0x75_0000	117	–
0x76_0000	118	–
0x77_0000	119	–
0x78_0000	120	–
0x79_0000	121	–
0x7A_0000	122	–
0x7B_0000	123	–
0x7C_0000	124	–
0x7D_0000	125	–
0x7E_0000	126	–
0x7F_0000	127	–

1. See the i.MX 7ULP Security Reference Manual for the complete chapter.

Table 19-5. A7 Peripheral memory map - AHB-PBridge1

Offset address	Slot	AHB-PBridge1
0x0_0000	0	–

Table continues on the next page...

Table 19-5. A7 Peripheral memory map - AHB-PBridge1 (continued)

Offset address	Slot	AHB-PBridge1
0x1_0000	1	–
0x2_0000	2	–
0x3_0000	3	–
0x4_0000	4	–
0x5_0000	5	–
0x6_0000	6	–
0x7_0000	7	–
0x8_0000	8	–
0x9_0000	9	–
0xA_0000	10	–
0xB_0000	11	–
0xC_0000	12	–
0xD_0000	13	–
0xE_0000	14	–
0xF_0000	15	–
0x10_0000	16	Reserved
0x11_0000	17	–
0x12_0000	18	–
0x13_0000	19	–
0x14_0000	20	–
0x15_0000	21	–
0x16_0000	22	–
0x17_0000	23	–
0x18_0000	24	–
0x19_0000	25	–
0x1A_0000	26	–
0x1B_0000	27	–
0x1C_0000	28	–
0x1D_0000	29	–
0x1E_0000	30	–
0x1F_0000	31	–
0x20_0000	32	–
0x21_0000	33	Timer/PWM Module 6 (TPM6)
0x22_0000	34	Timer/PWM Module 7 (TPM7)
0x23_0000	35	–
0x24_0000	36	Low Power I2C 6 (LPI2C6)
0x25_0000	37	Low Power I2C 7 (LPI2C7)
0x26_0000	38	Low Power UART 6 (LPUART6)
0x27_0000	39	Low Power UART 7 (LPUART7)

Table continues on the next page...

Table 19-5. A7 Peripheral memory map - AHB-PBridge1 (continued)

Offset address	Slot	AHB-PBridge1
0x28_0000	40	Video-In Unit (VIU)
0x29_0000	41	MIPI Display Serial Interface (DSI)
0x2A_0000	42	LCD Interface (LCDIF)
0x2B_0000	43	Multi Mode DDR Controller (MMDC)
0x2C_0000	44	Input/Output Multiplexing Control 1 (IOMUXC1)
0x2D_0000	45	Input/Output Multiplexing Control DDR (IOMUXC_DDR)
0x2E_0000	46	Port Interupt Control C (PCTLC)
0x2F_0000	47	Port Interupt Control D (PCTLD)
0x30_0000	48	Port Interupt Control E (PCTLE)
0x31_0000	49	Port Interupt Control F (PCTLF)
0x32_0000	50	–
0x33_0000	51	Peripheral Clock Control 3 (PCC3)
0x34_0000	52	–
0x35_0000	53	-
0x36_0000	54	–
0x37_0000	55	–
0x38_0000	56	–
0x39_0000	57	–
0x3A_0000	58	–
0x3B_0000	59	–
0x3C_0000	60	–
0x3D_0000	61	–
0x3E_0000	62	–
0x3F_0000	63	–
0x40_0000	64	–
0x41_0000	65	–
0x42_0000	66	–
0x43_0000	67	–
0x44_0000	68	–
0x45_0000	69	–
0x46_0000	70	–
0x47_0000	71	–
0x48_0000	72	–
0x49_0000	73	–
0x4A_0000	74	–
0x4B_0000	75	–
0x4C_0000	76	–
0x4D_0000	77	–

Table continues on the next page...

Table 19-5. A7 Peripheral memory map - AHB-PBridge1 (continued)

Offset address	Slot	AHB-PBridge1
0x4E_0000	78	–
0x4F_0000	79	–
0x50_0000	80	–
0x51_0000	81	–
0x52_0000	82	–
0x53_0000	83	–
0x54_0000	84	–
0x55_0000	85	–
0x56_0000	86	–
0x57_0000	87	–
0x58_0000	88	–
0x59_0000	89	–
0x5A_0000	90	–
0x5B_0000	91	–
0x5C_0000	92	–
0x5D_0000	93	–
0x5E_0000	94	–
0x5F_0000	95	–
0x60_0000	96	–
0x61_0000	97	–
0x62_0000	98	–
0x63_0000	99	–
0x64_0000	100	–
0x65_0000	101	–
0x66_0000	102	–
0x67_0000	103	–
0x68_0000	104	–
0x69_0000	105	–
0x6A_0000	106	–
0x6B_0000	107	–
0x6C_0000	108	–
0x6D_0000	109	–
0x6E_0000	110	–
0x6F_0000	111	–
0x70_0000	112	–
0x71_0000	113	–
0x72_0000	114	–
0x73_0000	115	–
0x74_0000	116	–

Table continues on the next page...

Table 19-5. A7 Peripheral memory map - AHB-PBridge1 (continued)

Offset address	Slot	AHB-PBridge1
0x75_0000	117	–
0x76_0000	118	–
0x77_0000	119	–
0x78_0000	120	–
0x79_0000	121	–
0x7A_0000	122	–
0x7B_0000	123	–
0x7C_0000	124	–
0x7D_0000	125	–
0x7E_0000	126	–
0x7F_0000	127	–

Chapter 20
Quad Serial Peripheral Interface Controller (QuadSPI)

20.1 Chip-specific QSPI information

Table 20-1. Reference links to related information

Topic	Related module	Reference
Full description	QSPI	QSPI
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

20.1.1 Quad Serial Peripheral Interface Controller (QuadSPI)

The Quad Serial Peripheral Interface (QuadSPI) module acts as an interface to a Quad SPI or Octal SPI serial flash device. The QuadSPI module may be used as the host interface to a serial memory-mapped device, such as FPGA. The QuadSPI module supports both Single Transfer Rate (STR) and Double Transfer Rate (DTR) modes. The following table shows the configuration of the QSPI.

Table 20-2. QuadSPI configuration

Parameter	Description
Name	Quad Serial Peripheral Interface
Instances	1
Configurable features	• 128-byte AHB Buffer • 16x32-bit RX FIFO • 16x32-bit TX FIFO • 64-bit AHB master port • DQS line supported

Table continues on the next page...

Table 20-2. QuadSPI configuration (continued)

Parameter	Description
	• Data learning enabled for DDR feature. • Support On-The-Fly AES Decryption (OTFAD) with 4 keys[1]
Interface speed	See the i.MX 7ULP Data Sheet for QuadSPI frequency specifications.
External I/O pins	See the attached IOMUXC spreadsheet for pin details

1. For details on this feature, see i.MX 7ULP Security Reference Manual

20.1.2 Number of supported flash device interfaces

This device only supports Flash Device A in dual-die package (2x4-bit (quad) mode) or single-die package (1x4-bit (quad) mode). Parallel mode is not supported in this device. So, all the sections on Flash Device B and parallel mode should be ignored.

20.1.3 QuadSPI register reset values

All reset values for the QuadSPI registers that are device specific are shown in the following tables. All other register reset values are shown in the module memory map.

Table 20-3. QuadSPI Module Configuration Register Reset Values

Register	Reset Value
Module Configuration Register (QuadSPI_MCR)	000F_400C

Table 20-4. QuadSPI Buffer Configuration Register Reset Values

Register	Reset Value
Buffer0 Configuration Register (QuadSPI_BUF0CR)	0000_000F
Buffer1 Configuration Register (QuadSPI_BUF1CR)	0000_000F
Buffer2 Configuration Register (QuadSPI_BUF2CR)	0000_000F
Buffer3 Configuration Register (QuadSPI_BUF3CR)	0000_000F

Table 20-5. Serial Flash A1 Top Address Reset Values

Register	Reset Value
QuadSPI_SFA1AD	C800_0000

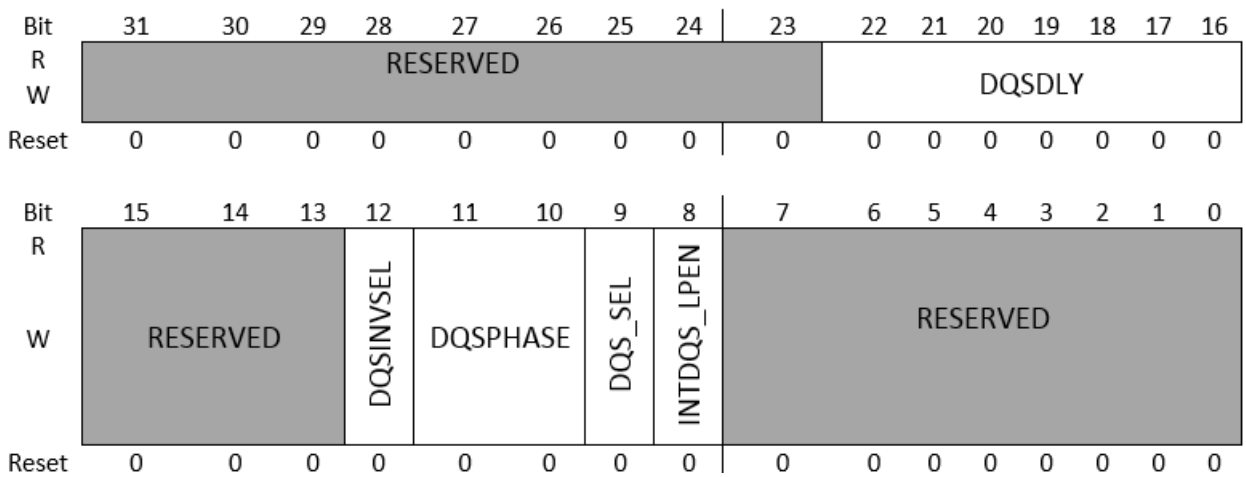
Table 20-6. Serial Flash A2 Top Address Reset Values

Register	Reset Value
QuadSPI_SFA2AD	D000_0000

20.1.4 Data strobe (DQS) sampling method using SOCCR register

For using data strobe sampling method, QuadSPI_SOCCR and QuadSPI_MCR registers should be configured.

On i.MX 7ULP, the QuadSPI_MCR[SCLKCFG] bitfield is not used and all clock-specific settings are defined by registers of the Peripheral Clock Control module (PCC). The QuadSPI_SOCCR register is used and controls settings related to the DQS.



Field	Description
31-23 RESERVED	This field is reserved
22-16 DQSDLY	Delay Chain Tap Selection (fine tuning) for QuadSPI DQS clock 0000000 – Select 1 delay chain tap 0000001 – Select 2 delay chain taps 0000010 – Select 3 delay chain taps …. 1111111 – Select 128 delay chain taps
15-13 RESERVED	This field is reserved
12 DQSINVSEL	Select clock source for the internal DQS generation 0 – Use 1x internal reference clock for the DQS generation 1 – Use 1x inverted internal reference clock for the DQS generation
11-10 DQSPHASE	Select phase shift for the internal DQS generation. These bits are always zero in SDR mode.

Table continues on the next page...

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>00 – No phase shift
01 – Select 45 degree phase shift
10 – Select 90 degree phase shift
11 – select 135 degree phase shift</td></tr>
<tr><td>9
INTDQS_LPEN</td><td>Internal DQS with Loopback Enable. When this bit is set, the DQS pad is configured to be an output, which will be pushing out the internal DQS clock. The DQS clock is then looped back to the QuadSPI module upon chip select assertion of either Flash A1 or Flash A2.

0 – internal DQS mode with loopback disabled. If DQS_SEL is set, internal DQS mode without loopback is in place. Else, External DQS is enabled.
1 – internal DQS mode with loopback enabled regardless of the DQS_SEL configuration.</td></tr>
<tr><td>8
DQS_SEL</td><td>DQS Selection.
This setting in conjunction with INTDQS_LPEN is used to select the three possible DQS modes: external DQS, internal DQS without loopback, and internal DQS with loopback.
• On external DQS mode, a dedicated DQS line comes from external Flash (see INTDQS_LPEN). For flashes not supporting a dedicated DQS line, the internal DQS modes can alternatively be used.
• On Internal DQS mode with loopback, the internal SFIF 1x clock is sent to the pad and then looped back to the QSPI.
• On internal DQS mode without loopback, the internal SFIF 1x clock is used as the DQS signal without going through a pads.
. The following Truth table shows the possible configurations:
<table>
<tr><th colspan="2">DQS configurations</th><th colspan="3">DQS modes</th></tr>
<tr><th>INTDQS_LPEN</th><th>DQS_SEL</th><th>Internal DQS with loopback</th><th>Internal DQS without loopback</th><th>External DQS</th></tr>
<tr><td>0</td><td>0</td><td>Disabled</td><td>Disabled</td><td>Enabled</td></tr>
<tr><td>0</td><td>1</td><td>Disabled</td><td>Enabled</td><td>Disabled</td></tr>
<tr><td>1</td><td>X</td><td>Enabled</td><td>Disabled</td><td>Disabled</td></tr>
</table>
</td></tr>
<tr><td>7-0
RESERVED</td><td>This field is reserved</td></tr>
</table>

20.1.5 Internal Data Strobe (DQS) Scheme

i.MX 7ULP does provide a dedicated DQS pad. The problem arises when the external memory does not provide such signal. In this condition, the chip can be used by generating an internal DQS signal. This internal signal is generated by looping back the serial clock outgoing to external memory to the QuadSPI after adding some delay.

The internal DQS signal is generated after passing through two delay mechanism: the coarse delay and the fine delay (delay chain). The coarse delay mechanism is adjusted by flops that run off the 2x and inverted 2x clocks in order to provide 45, 90, and 135 degree shift. See Data strobe (DQS) sampling method using SOCCR register for details of how to configure the DQS clock. Even for external flash that provides the DQS clock, this clock is passed through the delay chain in order to match the data valid period. This adjustment is performed by writing into the QuadSPI_SOCCR[DQSDLY]. QuadSPI internally samples the incoming data at both edges of the DQS clock.

The QuadSPI_MCR[DQS_EN]) is used to enable the DQS clock only during read cycles. This signal in turn may be delayed appropriately by writing into the QuadSPI_SMPR[FSDLY] and QuadSPI_SMPR[FSPHS].

NOTE

Any glitch on the muxes used above will not cause any impact to the functionality as when the select line of these muxes are changing, the DQS clock to the QuadSPI is gated off.

20.2 Introduction

The Quad Serial Peripheral Interface (QuadSPI) block acts as an interface to a single serial flash device, with up to eight bidirectional data lines.

20.2.1 Features

The QuadSPI supports the following features:

- Flexible sequence engine to support various flash vendor devices. As there is no specific standard, the module supports various kind of flashes from different vendors. Refer Serial Flash Devices for example sequences.
- Single, dual, quad and octal modes of operation.
- Double data rate (DDR)/Double trasfer rate (DTR) mode wherein the data is generated on every edge of the serial flash clock.
- Support for flash data strobe signal for data sampling in DDR and Single data rate (SDR) mode.
- AHB master to read RX Buffer data via AMBA AHB bus (64-bit width interface) or IP registers space (32-bit access) and fill TX Buffer via IPS register space (32-bit access).

    - AHB master can be a DMA with configurable inner loop size.
- Multimaster accesses with priority
    - Flexible and configurable buffer for each master
- Multiple interrupt conditions (see Table 20-14)
- All AHB accesses to flash devices are directly memory mapped to the chip system memory.
- Programmable sequence engine to cater to future command/protocol changes and able to support all existing vendor commands and operations. Software needs to select the corresponding sequence according to the connected flash device.
    - Supports all types of addressing.

20.2.2 Block Diagram

The following figure is a block diagram of the Quad Serial Peripheral Interface (QuadSPI) module.

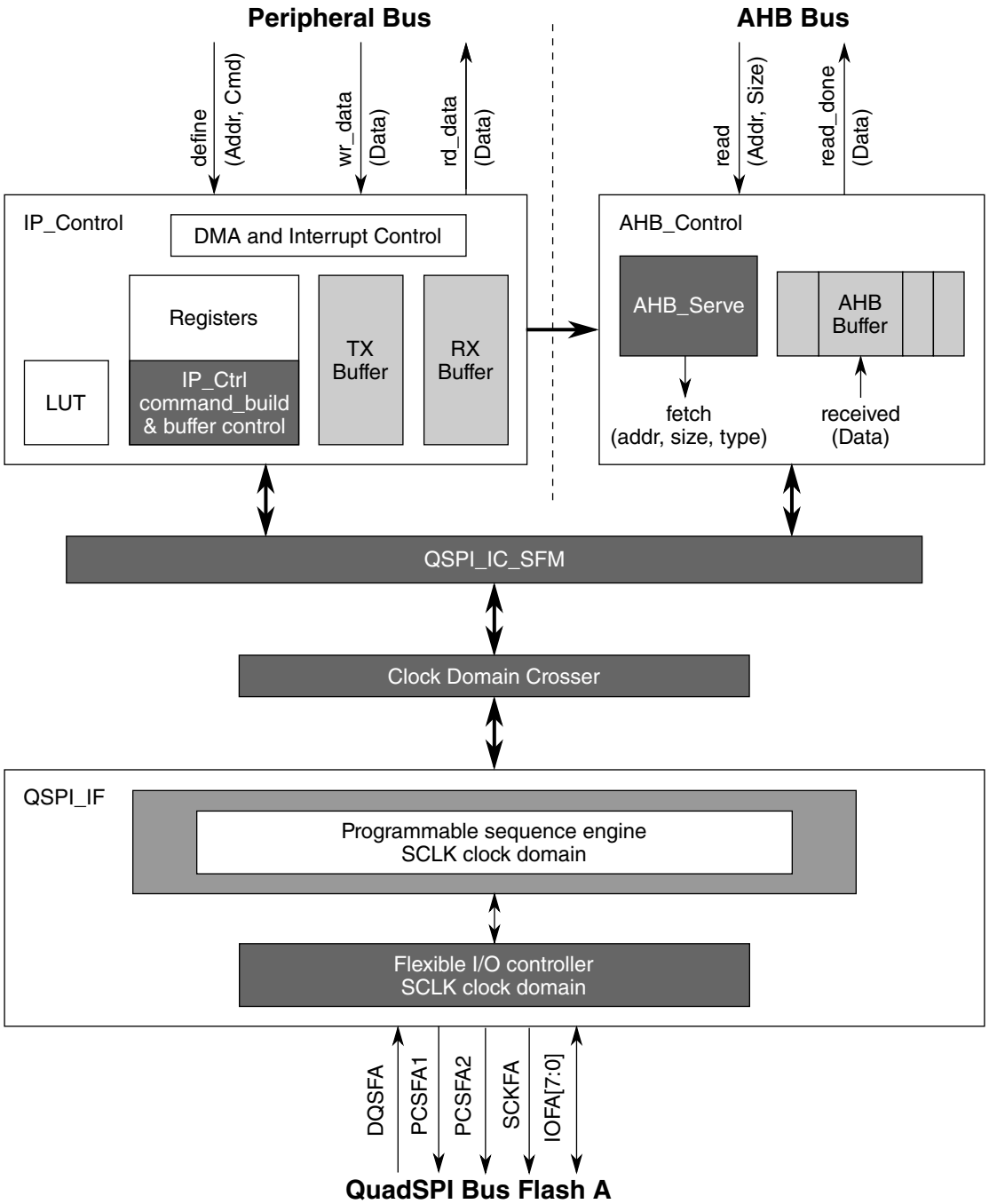


Figure 20-1. QuadSPI Block Diagram

20.2.3 QuadSPI Modes of Operation

For power management through IPS interface access and correct config register programming sequences, QuadSPI supports three modes: normal, module disable and stop mode.

- Normal Mode can be used for write or read accesses to an external serial flash device.
  - Serial Flash Write: Data can be programmed into the flash via the IP interface only. Refer to Flash Programming for further details.
  - Serial Flash Read: Read the contents of the serial flash device. Two separate read channels are available via RX Buffer and AHB Buffer, see Flash Read.
- Stop Mode: The mode is used for power management. When a request is made to enter Stop Mode, the QuadSPI block acknowledges the request and completes the SFM Command in progress, then the system clocks to the QuadSPI block may be shut off
- Module Disable Mode: The mode is used for power management. The clock to the non-memory mapped logic in the QuadSPI can be stopped while in Module Disable Mode. The module enters the mode by setting QSPI_MCR[MDIS].

20.2.4 Acronyms and Abbreviations

The following table contains acronyms and abbreviations used in this document.

Table 20-7. Acronyms and Abbreviations

Terms	Description
AHB	Advanced High-performance Bus, version of AMBA
AMBA	Advanced Microcontroller Bus Architecture
BE	Big Endian Byte Ordering
CS	Chip Select
DMA	Direct Memory Access
MB	Megabyte. Each MB is 1024 * 1024 bytes
IFM	Individual Flash Mode
LSB	Least Significant Bit
MSB	Most Significant Bit
PCS	Peripheral Chip Select
QSPI, QuadSPI	Quad Serial Peripheral Interface
SCK	Serial Communications Clock
w1c	Write 1 to clear, writing a 1 to this field resets the flag
I/O	Input output. In this document, I/O lines are also referred as **pads**.

20.2.5 Glossary for QuadSPI module

Table 20-8. Glossary

Term	Definition
AHB Command	An AHB Command is a SFM Command triggered by a read access to the address range belonging to the memory mapped access defined in Table 20-13. Refer to AHB Commands for details.
Asserted	A signal that is asserted is in its active state. An active low signal changes from logic level one to logic level zero when asserted, and an active high signal changes from logic level zero to logic level one.
Clear	To clear a bit or bits means to establish logic level zero on the bit or bits.
Clock Phase	Determines when the data should be sampled relative to the active edge of SCK
Clock Polarity	Determines the idle state of the SCK signal.
Drain	To remove entries from a FIFO by software or hardware.
Endianness	Byte Ordering scheme.
Field	Two or more register bits grouped together.
Fill	To add entries to a FIFO by software or hardware.
Host	Refers to another functional block in the device containing the QuadSPI module
Instruction Code	8 bits defining the type of command to be executed.
IP Command	An IP Command is a SFM Command triggered by writing into the QSPI_IPCR[SEQID] field.
Logic level one	The voltage that corresponds to Boolean true (1) state.
Logic level zero	The voltage that corresponds to Boolean false (0) state.
Negated	A signal that is negated is in its inactive state. An active low signal changes from logic level 0 to logic level 1 when negated, and an active high signal changes from logic level 1 to logic level 0.
QSPI_AMBA_BASE	The first address of the serial flash device as presented to the QuadSPI controller. This may be the base of the serial flash in the system address map or it may be a remapping, for instance to 0x0, done by the system. Refer to the system address map.
QSPI_ARDB_BASE	First address of QuadSPI Rx Buffer on system memory map.
Set	To set a bit or bits means to establish logic level one on the bit or bits.
RX Buffer PUSH Event	Addition of valid entries into the RX Buffer. In the default case each Buffer PUSH Event adds 2 entries to the RX Buffer since the interface to the serial clock domain is 64 bits in width. Depending on the number of bytes read from the serial flash device it is possible for the very last Buffer PUSH Event that only one entry is added. The QSPI_RBSR[RDBFL] field is incremented by the number of entries added to the RX Buffer.
RX Buffer POP Event	Removal of valid entries from the RX Buffer. Each Buffer POP Event removes (QSPI_RBCT[WMRK] + 1) valid entries from the buffer. The QSPI_RBSR[RDBFL] field is decremented by the same number and the QSPI_RBSR[RDCTR] field is incremented accordingly.
Individual Flash Mode	Access to a single, individual serial flash device. Refer to Serial Flash Access Schemes for details.
SFM Command	Serial Flash Memory Command. A SFM command consists of an instruction code and all other parameters (for example, size or mode bytes) needed for that specific instruction code. Triggering a command either initiates a transaction on the external serial flash or results in an error. Refer to Table 20-20 for details on errors.
Single/Dual/Quad/Octal Instructions	Depending on the serial flash device connected to the QuadSPI module there will be instructions using a different number of data lines. • Single pad: Single line I/O with one data out and one data in line to/from the serial flash device. • Dual pad: Dual line I/O with two bidirectional I/O lines, driven alternatively by the serial flash device or the QuadSPI module

Table continues on the next page...

Table 20-8. Glossary (continued)

Term	Definition
	• Quad pad: Quad line I/O with 4 bidirectional I/O lines, driven alternatively by the serial flash device or the QuadSPI • Octal pad: Octal line I/O with 8 bidirectional I/O lines, driven alternatively by the serial flash device or the QuadSPI
Transaction	A transaction consists of all flags, data and signals in either direction to execute a command for an attached serial flash device. It is a combination of chip select, sclk, instruction code, address, mode- and/or dummy bytes, transmit and/or receive data.
LUT	Look-up table.

20.3 External Signal Description

This section provides the external signal information of the QuadSPI module.

The following table lists the external signals belonging to the module in conjunction with the different modes of operation.

Table 20-9. Signal Properties

Signal Name	Function	Direction	Description
PCSFA1	Peripheral Chip Select Flash A1	O	This signal is the chip select for the serial flash device A1. A1 represents the first device in a dual-die package flash A or the first of the two flash devices that share IOFA. Refer to Dual Die Flashes for more details.
PCSFA2	Peripheral Chip Select Flash A2	O	This signal is the chip select for the serial flash device A2. A2 represents the second device in a dual-die package flash A or the second of the two flash devices that share IOFA. Refer to Dual Die Flashes for more details.
SCKFA	Serial Clock Flash A	O	This signal is the serial clock output to the serial flash device A.
IOFA[7:0]	Serial I/O Flash A	I/O	These signals are the data I/O lines to/from the serial flash device A. Refer to Driving External Signals for details about the signal drive and timing behavior. Note that the signal pins of the serial flash device may change their function according to the SFM Command executed, leaving them as control inputs when Single and Dual Instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFA[0] and expects data on IOFA[1].
DQSFA	Data Strobe signal Flash A	I	Data strobe signal for port A. Some flash vendors provide the DQS signal to which the read data is aligned in DDR mode.

20.3.1 Driving External Signals

The different phases of the serial flash access scheme are shown in the following figure.

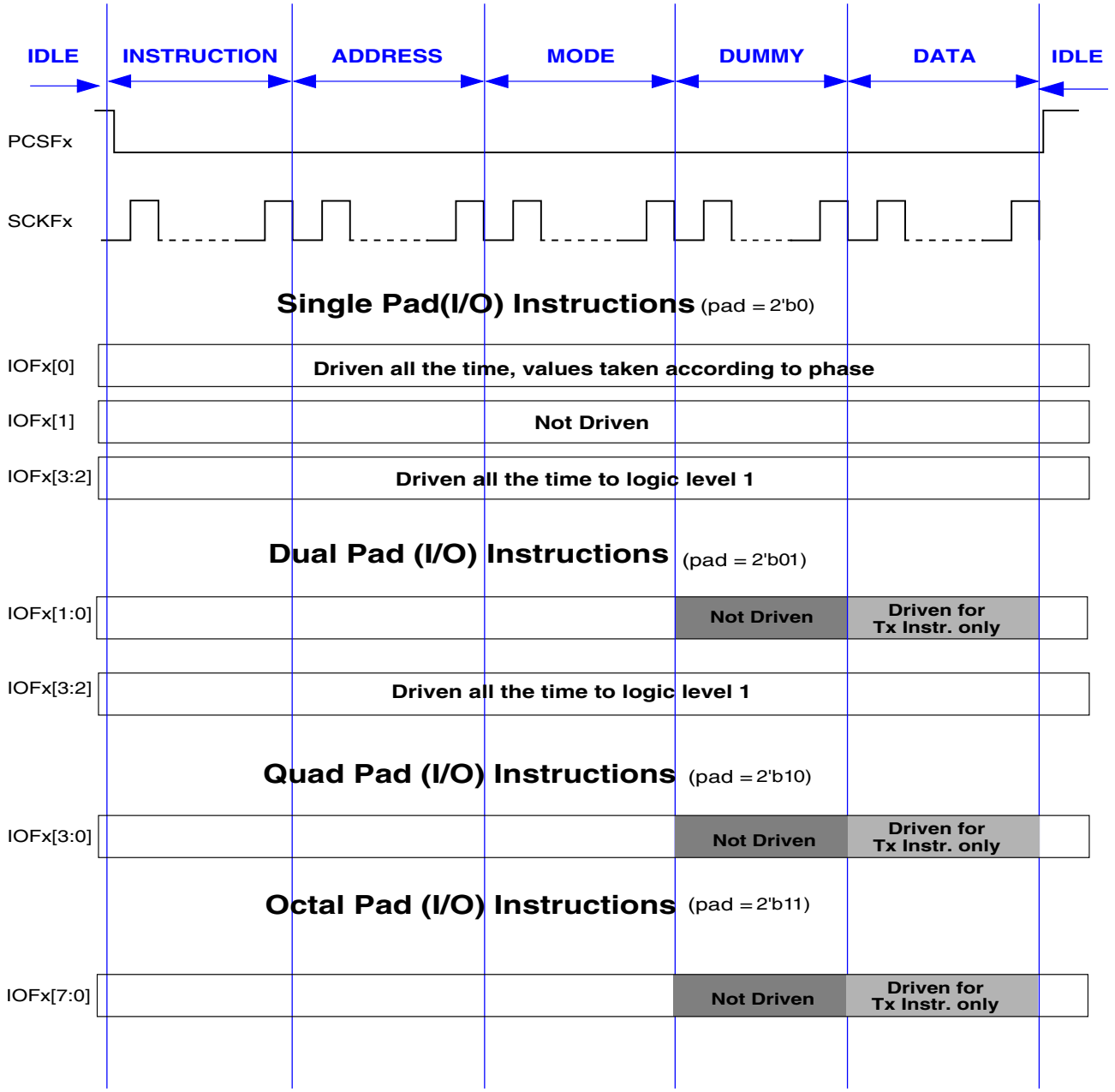


Figure 20-2. Serial Flash Access Scheme

The different phases and the I/O driving characteristics of the QuadSPI module are characterized in the following way:

- IDLE: Serial flash device not selected. No interaction with the serial flash device. All IOFx signals driven.
- INSTRUCTION: Serial flash device selected. The instruction is sent to the serial flash device. All IOFx signals are driven.

- ADDRESS: Serial Flash Address is sent to the device. All IOFx signals are driven. Note that this phase is not applicable for all SFM Commands.
- MODE: Mode bytes are sent to the serial flash device. All IOFx signals are driven. Note that this phase is not applicable for all SFM Commands.
- DUMMY: Dummy clocks are provided to the serial flash device. Refer to the Figure 20-2 for the IOFx signals driven. The actual data lines required for the SFM Command executed are not driven for data read commands. Note that this phase is not applicable for all SFM Commands.
- DATA: Serial flash data are sent to or received from the serial flash device. Refer to the preceding figure for the IOFx signals driven. The actual data lines required for the SFM Command executed are not driven for data read commands. Note that this phase is not applicable for all SFM Commands.

The PCSFx and SCKFx signals are driven permanently throughout all the phases.

In Individual Flash Mode this applies to the selected flash device.

20.4 Memory Map and Register Definition

This section provides the memory map and register definitions of the QuadSPI module.

20.4.1 Register Write Access

This section describes the write access restriction terms that apply to all registers, which can be one of the following:

- **Register Write Access Restriction**

  For each register bit and register field, the write access conditions are specified in the detailed register description. A description of the write access conditions is given in the following table. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed.

  The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

Table 20-10. Register Write Access Restrictions

Condition	Description
Anytime	No write access restriction.
Disabled Mode	Write access only if *QSPI_MCR[MDIS] = 1*.
Normal Mode	Write access only if the module is in *Normal Mode*.

- **Register Write Access Requirements**

  All registers can be accessed with 8-bit, 16-bit, and 32-bit wide operations. For some of the registers, at least a 16/32-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each register affected.

20.4.2 Peripheral Bus Register Descriptions

This section provides the memory map and register definitions of the QuadSPI module.

NOTE

Access to location 0x4 does not give transfer error.

QuadSPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
410A_5000	Module Configuration Register (QuadSPI0_MCR)	32	R/W	See section	20.4.2.1/ 424
410A_5008	IP Configuration Register (QuadSPI0_IPCR)	32	R/W	0000_0000h	20.4.2.2/ 427
410A_500C	Flash Configuration Register (QuadSPI0_FLSHCR)	32	R/W	0000_0303h	20.4.2.3/ 428
410A_5010	Buffer0 Configuration Register (QuadSPI0_BUF0CR)	32	R/W	See section	20.4.2.4/ 429
410A_5014	Buffer1 Configuration Register (QuadSPI0_BUF1CR)	32	R/W	See section	20.4.2.5/ 431
410A_5018	Buffer2 Configuration Register (QuadSPI0_BUF2CR)	32	R/W	See section	20.4.2.6/ 432
410A_501C	Buffer3 Configuration Register (QuadSPI0_BUF3CR)	32	R/W	See section	20.4.2.7/ 433

Table continues on the next page...

QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
410A_5020	Buffer Generic Configuration Register (QuadSPI0_BFGENCR)	32	R/W	0000_0000h	20.4.2.8/ 434
410A_5024	SOC Configuration Register (QuadSPI0_SOCCR)	32	R/W	0000_0000h	20.4.2.9/ 435
410A_5030	Buffer0 Top Index Register (QuadSPI0_BUF0IND)	32	R/W	0000_0000h	20.4.2.10/ 436
410A_5034	Buffer1 Top Index Register (QuadSPI0_BUF1IND)	32	R/W	0000_0000h	20.4.2.11/ 436
410A_5038	Buffer2 Top Index Register (QuadSPI0_BUF2IND)	32	R/W	0000_0000h	20.4.2.12/ 437
410A_5100	Serial Flash Address Register (QuadSPI0_SFAR)	32	R/W	0000_0000h	20.4.2.13/ 438
410A_5104	Serial Flash Address Configuration Register (QuadSPI0_SFACR)	32	R/W	0000_0000h	20.4.2.14/ 439
410A_5108	Sampling Register (QuadSPI0_SMPR)	32	R/W	0000_0000h	20.4.2.15/ 440
410A_510C	RX Buffer Status Register (QuadSPI0_RBSR)	32	R	0000_0000h	20.4.2.16/ 442
410A_5110	RX Buffer Control Register (QuadSPI0_RBCT)	32	R/W	0000_0000h	20.4.2.17/ 443
410A_5150	TX Buffer Status Register (QuadSPI0_TBSR)	32	R	0000_0000h	20.4.2.18/ 444
410A_5154	TX Buffer Data Register (QuadSPI0_TBDR)	32	R/W	0000_0000h	20.4.2.19/ 445
410A_5158	Tx Buffer Control Register (QuadSPI0_TBCT)	32	R/W	0000_0000h	20.4.2.20/ 446
410A_515C	Status Register (QuadSPI0_SR)	32	R	0200_3800h	20.4.2.21/ 447
410A_5160	Flag Register (QuadSPI0_FR)	32	w1c	0800_0000h	20.4.2.22/ 450
410A_5164	Interrupt and DMA Request Select and Enable Register (QuadSPI0_RSER)	32	R/W	0000_0000h	20.4.2.23/ 453
410A_5168	Sequence Suspend Status Register (QuadSPI0_SPNDST)	32	R	0000_0000h	20.4.2.24/ 457
410A_516C	Sequence Pointer Clear Register (QuadSPI0_SPTRCLR)	32	R/W	0000_0000h	20.4.2.25/ 459
410A_5180	Serial Flash A1 Top Address (QuadSPI0_SFA1AD)	32	R/W	See section	20.4.2.26/ 460
410A_5184	Serial Flash A2 Top Address (QuadSPI0_SFA2AD)	32	R/W	See section	20.4.2.27/ 460
410A_5190	Data Learn Pattern Register (QuadSPI0_DLPR)	32	R/W	AA55_3443h	20.4.2.28/ 461
410A_5200	RX Buffer Data Register (QuadSPI0_RBDR0)	32	R	0000_0000h	20.4.2.29/ 461

Table continues on the next page...

QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
410A_5204	RX Buffer Data Register (QuadSPI0_RBDR1)	32	R	0000_0000h	20.4.2.29/ 461
410A_5208	RX Buffer Data Register (QuadSPI0_RBDR2)	32	R	0000_0000h	20.4.2.29/ 461
410A_520C	RX Buffer Data Register (QuadSPI0_RBDR3)	32	R	0000_0000h	20.4.2.29/ 461
410A_5210	RX Buffer Data Register (QuadSPI0_RBDR4)	32	R	0000_0000h	20.4.2.29/ 461
410A_5214	RX Buffer Data Register (QuadSPI0_RBDR5)	32	R	0000_0000h	20.4.2.29/ 461
410A_5218	RX Buffer Data Register (QuadSPI0_RBDR6)	32	R	0000_0000h	20.4.2.29/ 461
410A_521C	RX Buffer Data Register (QuadSPI0_RBDR7)	32	R	0000_0000h	20.4.2.29/ 461
410A_5220	RX Buffer Data Register (QuadSPI0_RBDR8)	32	R	0000_0000h	20.4.2.29/ 461
410A_5224	RX Buffer Data Register (QuadSPI0_RBDR9)	32	R	0000_0000h	20.4.2.29/ 461
410A_5228	RX Buffer Data Register (QuadSPI0_RBDR10)	32	R	0000_0000h	20.4.2.29/ 461
410A_522C	RX Buffer Data Register (QuadSPI0_RBDR11)	32	R	0000_0000h	20.4.2.29/ 461
410A_5230	RX Buffer Data Register (QuadSPI0_RBDR12)	32	R	0000_0000h	20.4.2.29/ 461
410A_5234	RX Buffer Data Register (QuadSPI0_RBDR13)	32	R	0000_0000h	20.4.2.29/ 461
410A_5238	RX Buffer Data Register (QuadSPI0_RBDR14)	32	R	0000_0000h	20.4.2.29/ 461
410A_523C	RX Buffer Data Register (QuadSPI0_RBDR15)	32	R	0000_0000h	20.4.2.29/ 461
410A_5300	LUT Key Register (QuadSPI0_LUTKEY)	32	R/W	5AF0_5AF0h	20.4.2.30/ 462
410A_5304	LUT Lock Configuration Register (QuadSPI0_LCKCR)	32	R/W	0000_0002h	20.4.2.31/ 462
410A_5310	Look-up Table register (QuadSPI0_LUT0)	32	R/W	See section	20.4.2.32/ 464
410A_5314	Look-up Table register (QuadSPI0_LUT1)	32	R/W	See section	20.4.2.32/ 464
410A_5318	Look-up Table register (QuadSPI0_LUT2)	32	R/W	See section	20.4.2.32/ 464
410A_531C	Look-up Table register (QuadSPI0_LUT3)	32	R/W	See section	20.4.2.32/ 464
410A_5320	Look-up Table register (QuadSPI0_LUT4)	32	R/W	See section	20.4.2.32/ 464

Table continues on the next page...

QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
410A_5324	Look-up Table register (QuadSPI0_LUT5)	32	R/W	See section	20.4.2.32/ 464
410A_5328	Look-up Table register (QuadSPI0_LUT6)	32	R/W	See section	20.4.2.32/ 464
410A_532C	Look-up Table register (QuadSPI0_LUT7)	32	R/W	See section	20.4.2.32/ 464
410A_5330	Look-up Table register (QuadSPI0_LUT8)	32	R/W	See section	20.4.2.32/ 464
410A_5334	Look-up Table register (QuadSPI0_LUT9)	32	R/W	See section	20.4.2.32/ 464
410A_5338	Look-up Table register (QuadSPI0_LUT10)	32	R/W	See section	20.4.2.32/ 464
410A_533C	Look-up Table register (QuadSPI0_LUT11)	32	R/W	See section	20.4.2.32/ 464
410A_5340	Look-up Table register (QuadSPI0_LUT12)	32	R/W	See section	20.4.2.32/ 464
410A_5344	Look-up Table register (QuadSPI0_LUT13)	32	R/W	See section	20.4.2.32/ 464
410A_5348	Look-up Table register (QuadSPI0_LUT14)	32	R/W	See section	20.4.2.32/ 464
410A_534C	Look-up Table register (QuadSPI0_LUT15)	32	R/W	See section	20.4.2.32/ 464
410A_5350	Look-up Table register (QuadSPI0_LUT16)	32	R/W	See section	20.4.2.32/ 464
410A_5354	Look-up Table register (QuadSPI0_LUT17)	32	R/W	See section	20.4.2.32/ 464
410A_5358	Look-up Table register (QuadSPI0_LUT18)	32	R/W	See section	20.4.2.32/ 464
410A_535C	Look-up Table register (QuadSPI0_LUT19)	32	R/W	See section	20.4.2.32/ 464
410A_5360	Look-up Table register (QuadSPI0_LUT20)	32	R/W	See section	20.4.2.32/ 464
410A_5364	Look-up Table register (QuadSPI0_LUT21)	32	R/W	See section	20.4.2.32/ 464
410A_5368	Look-up Table register (QuadSPI0_LUT22)	32	R/W	See section	20.4.2.32/ 464
410A_536C	Look-up Table register (QuadSPI0_LUT23)	32	R/W	See section	20.4.2.32/ 464
410A_5370	Look-up Table register (QuadSPI0_LUT24)	32	R/W	See section	20.4.2.32/ 464
410A_5374	Look-up Table register (QuadSPI0_LUT25)	32	R/W	See section	20.4.2.32/ 464
410A_5378	Look-up Table register (QuadSPI0_LUT26)	32	R/W	See section	20.4.2.32/ 464

Table continues on the next page...

QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
410A_537C	Look-up Table register (QuadSPI0_LUT27)	32	R/W	See section	20.4.2.32/ 464
410A_5380	Look-up Table register (QuadSPI0_LUT28)	32	R/W	See section	20.4.2.32/ 464
410A_5384	Look-up Table register (QuadSPI0_LUT29)	32	R/W	See section	20.4.2.32/ 464
410A_5388	Look-up Table register (QuadSPI0_LUT30)	32	R/W	See section	20.4.2.32/ 464
410A_538C	Look-up Table register (QuadSPI0_LUT31)	32	R/W	See section	20.4.2.32/ 464
410A_5390	Look-up Table register (QuadSPI0_LUT32)	32	R/W	See section	20.4.2.32/ 464
410A_5394	Look-up Table register (QuadSPI0_LUT33)	32	R/W	See section	20.4.2.32/ 464
410A_5398	Look-up Table register (QuadSPI0_LUT34)	32	R/W	See section	20.4.2.32/ 464
410A_539C	Look-up Table register (QuadSPI0_LUT35)	32	R/W	See section	20.4.2.32/ 464
410A_53A0	Look-up Table register (QuadSPI0_LUT36)	32	R/W	See section	20.4.2.32/ 464
410A_53A4	Look-up Table register (QuadSPI0_LUT37)	32	R/W	See section	20.4.2.32/ 464
410A_53A8	Look-up Table register (QuadSPI0_LUT38)	32	R/W	See section	20.4.2.32/ 464
410A_53AC	Look-up Table register (QuadSPI0_LUT39)	32	R/W	See section	20.4.2.32/ 464
410A_53B0	Look-up Table register (QuadSPI0_LUT40)	32	R/W	See section	20.4.2.32/ 464
410A_53B4	Look-up Table register (QuadSPI0_LUT41)	32	R/W	See section	20.4.2.32/ 464
410A_53B8	Look-up Table register (QuadSPI0_LUT42)	32	R/W	See section	20.4.2.32/ 464
410A_53BC	Look-up Table register (QuadSPI0_LUT43)	32	R/W	See section	20.4.2.32/ 464
410A_53C0	Look-up Table register (QuadSPI0_LUT44)	32	R/W	See section	20.4.2.32/ 464
410A_53C4	Look-up Table register (QuadSPI0_LUT45)	32	R/W	See section	20.4.2.32/ 464
410A_53C8	Look-up Table register (QuadSPI0_LUT46)	32	R/W	See section	20.4.2.32/ 464
410A_53CC	Look-up Table register (QuadSPI0_LUT47)	32	R/W	See section	20.4.2.32/ 464
410A_53D0	Look-up Table register (QuadSPI0_LUT48)	32	R/W	See section	20.4.2.32/ 464

Table continues on the next page...

QuadSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
410A_53D4	Look-up Table register (QuadSPI0_LUT49)	32	R/W	See section	20.4.2.32/ 464
410A_53D8	Look-up Table register (QuadSPI0_LUT50)	32	R/W	See section	20.4.2.32/ 464
410A_53DC	Look-up Table register (QuadSPI0_LUT51)	32	R/W	See section	20.4.2.32/ 464
410A_53E0	Look-up Table register (QuadSPI0_LUT52)	32	R/W	See section	20.4.2.32/ 464
410A_53E4	Look-up Table register (QuadSPI0_LUT53)	32	R/W	See section	20.4.2.32/ 464
410A_53E8	Look-up Table register (QuadSPI0_LUT54)	32	R/W	See section	20.4.2.32/ 464
410A_53EC	Look-up Table register (QuadSPI0_LUT55)	32	R/W	See section	20.4.2.32/ 464
410A_53F0	Look-up Table register (QuadSPI0_LUT56)	32	R/W	See section	20.4.2.32/ 464
410A_53F4	Look-up Table register (QuadSPI0_LUT57)	32	R/W	See section	20.4.2.32/ 464
410A_53F8	Look-up Table register (QuadSPI0_LUT58)	32	R/W	See section	20.4.2.32/ 464
410A_53FC	Look-up Table register (QuadSPI0_LUT59)	32	R/W	See section	20.4.2.32/ 464
410A_5400	Look-up Table register (QuadSPI0_LUT60)	32	R/W	See section	20.4.2.32/ 464
410A_5404	Look-up Table register (QuadSPI0_LUT61)	32	R/W	See section	20.4.2.32/ 464
410A_5408	Look-up Table register (QuadSPI0_LUT62)	32	R/W	See section	20.4.2.32/ 464
410A_540C	Look-up Table register (QuadSPI0_LUT63)	32	R/W	See section	20.4.2.32/ 464

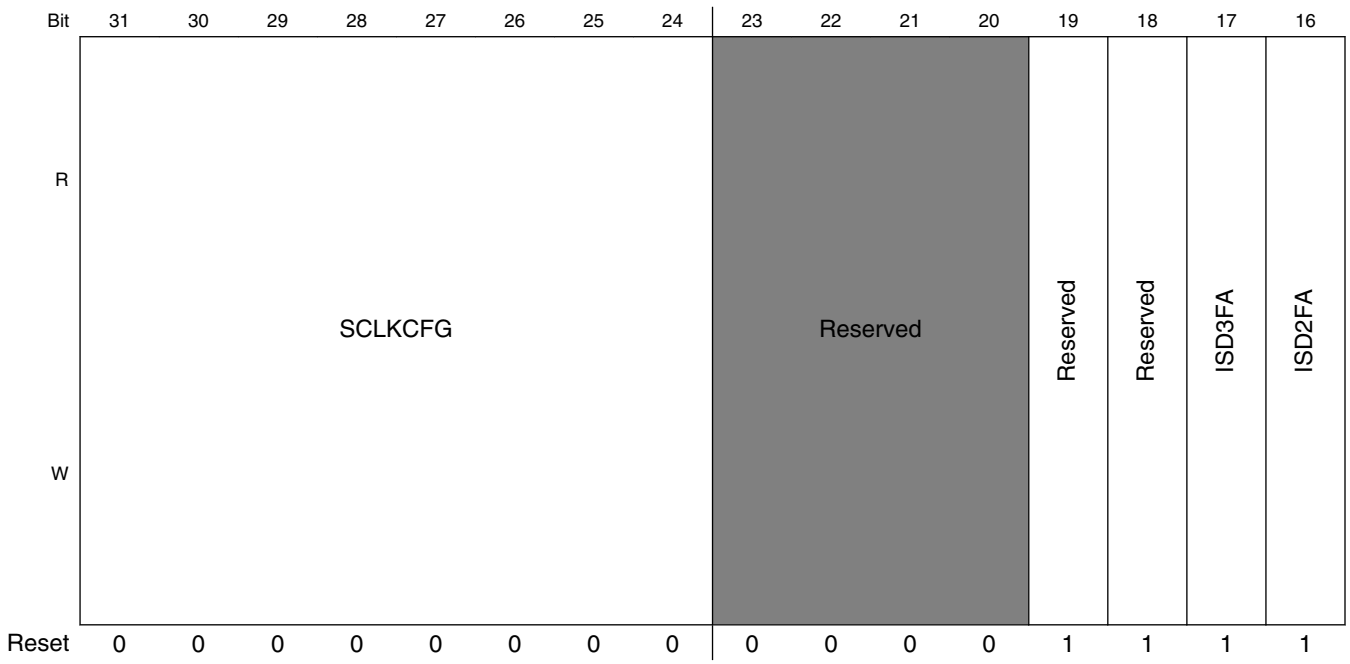
20.4.2.1 Module Configuration Register (QuadSPIx_MCR)

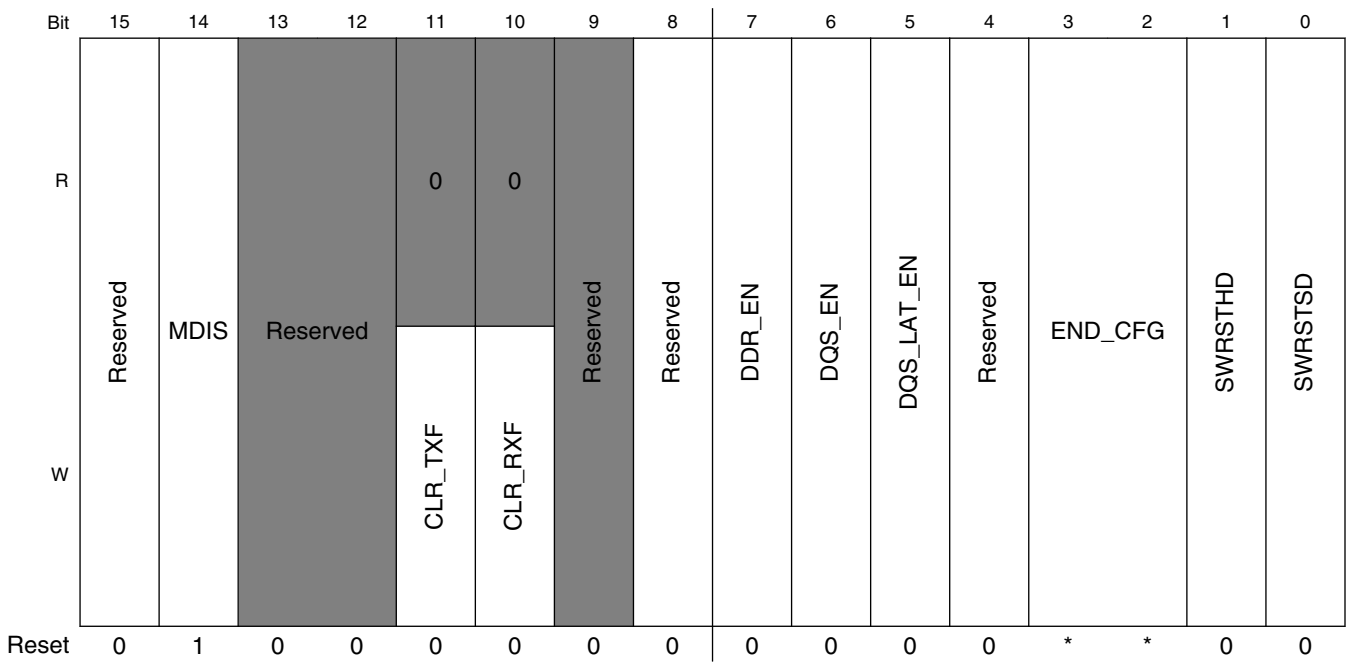
The QuadSPI_MCR holds configuration data associated with QuadSPI operation.

Write:

- *SCLKCFG: Disabled Mode*
- *ISD3FA, ISD2FA: Disabled Mode*
- *All other fields: Anytime*

Address: 410A_5000h base + 0h offset = 410A_5000h





* Notes:
- END_CFG field: See the module configuration for the device specific reset value.

QuadSPIx_MCR field descriptions

Field	Description
31–24 SCLKCFG	Serial Clock Configuration. This field configuration is chip specific. For details, refer to chip-specific QuadSPI information. It may be used for dividing clocks.

Table continues on the next page...

QuadSPIx_MCR field descriptions (continued)

Field	Description
23–20 Reserved	This field is reserved.
19 Reserved	This field is reserved.
18 Reserved	This field is reserved.
17 ISD3FA	Idle Signal Drive IOFA[3] Flash A. This bit determines the logic level the IOFA[3] output of the QuadSPI module is driven to in the inactive state. Refer to Driving Flash Control Signals in Single and Dual Mode . 0 IOFA[3] is driven to logic L 1 IOFA[3] is driven to logic H
16 ISD2FA	Idle Signal Drive IOFA[2] Flash A. This bit determines the logic level the IOFA[2] output of the QuadSPI module is driven to in the inactive state. Refer to Driving Flash Control Signals in Single and Dual Mode . 0 IOFA[2] is driven to logic L 1 IOFA[2] is driven to logic H
15 Reserved	This field is reserved. This field is reserved.
14 MDIS	Module Disable. The MDIS bit allows the clock to the non-memory mapped logic in the QuadSPI to be stopped, putting the QuadSPI in a software controlled power-saving state. 0 Enable QuadSPI clocks. 1 Allow external logic to disable QuadSPI clocks.
13–12 Reserved	This field is reserved.
11 CLR_TXF	Clear TX FIFO/Buffer. Invalidates the TX Buffer content. This is a self-clearing field. 0 No action. 1 Read and write pointers of the TX Buffer are reset to 0. QSPI_TBSR[TRCTR] is reset to 0.
10 CLR_RXF	Clear RX FIFO. Invalidates the RX Buffer. This is a self-clearing field. 0 No action. 1 Read and write pointers of the RX Buffer are reset to 0. QSPI_RBSR[RDBFL] is reset to 0.
9 Reserved	This field is reserved.
8 Reserved	This field is reserved.
7 DDR_EN	DDR mode enable 0 2x and 4x clocks are disabled for SDR instructions only 1 2x and 4x clocks are enabled supports both SDR and DDR instruction.
6 DQS_EN	DQS enable. This field is valid for both SDR and DDR mode. For more details, refer to Data Strobe (DQS) sampling method.

Table continues on the next page...

QuadSPIx_MCR field descriptions (continued)

Field	Description
	0 DQS disabled. 1 DQS enabled. When enabled, the incoming data is sampled on both the edges of DQS input when QSPI_MCR[DDR_EN] is set, else, on only one edge when QSPI_MCR[DDR_EN] is 0. The QSPI_SMPR[DDR_SMP] values are ignored.
5 DQS_LAT_EN	DQS Latency Enable. This field is valid when latency is included in between read access from flash memory in cases when QSPI_MCR[DQS_EN] is 1. For more details, refer to Data Strobe (DQS) sampling method. 0 DQS Latency disabled 1 DQS feature with latency included enabled
4 Reserved	This field is reserved.
3–2 END_CFG	Defines the endianness of the QuadSPI module. For more details refer to Byte Ordering Endianess
1 SWRSTHD	Software reset for AHB domain 0 No action 1 AHB domain flops are reset. Does not reset configuration registers. It is advisable to reset both the serial flash domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE:** The software resets need the clock to be running to propagate to the design. The MCR[MDIS] should therefore be set to 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTHD] to 0), it is recommended to set the MCR[MDIS] bit to 1. Once the software resets have been deasserted, the normal operation can be started by setting the MCR[MDIS] bit to 0.
0 SWRSTSD	Software reset for serial flash domain 0 No action 1 Serial Flash domain flops are reset. Does not reset configuration registers. It is advisable to reset both the serial flash domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE:** The software resets need the clock to be running to propagate to the design. The MCR[MDIS] should therefore be set to 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTSD] to 0), it is recommended to set the MCR[MDIS] bit to 1. Once the software resets have been deasserted, the normal operation can be started by setting the MCR[MDIS] bit to 0.

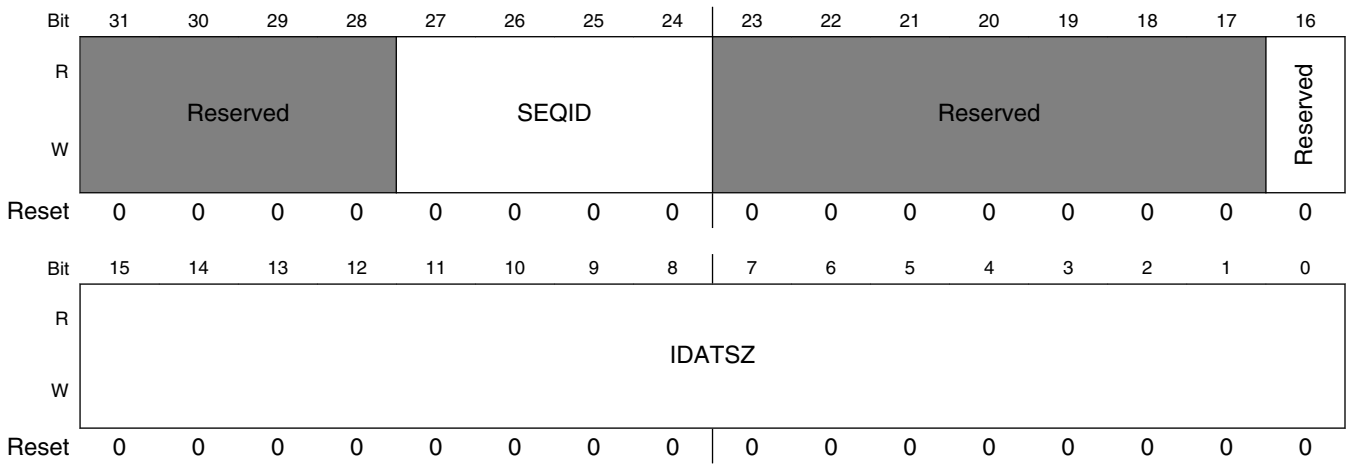
20.4.2.2 IP Configuration Register (QuadSPIx_IPCR)

The IP configuration register provides all the configuration required for an IP initiated command. An IP command can be triggered by writing in the SEQID field of this register. If the SEQID field is written successfully, a new command to the external serial flash is started as per the sequence pointed to by the SEQID field. Refer to Normal Mode , for details about the command triggering and command execution.

Write:

- *QSPI_SR[IP_ACC]=0*

Address: 410A_5000h base + 8h offset = 410A_5008h



QuadSPIx_IPCR field descriptions

Field	Description
31–28 Reserved	This field is reserved.
27–24 SEQID	Points to a sequence in the Look-up table. This field defines bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. Refer to Look-up Table for more details. A write to this field triggers a transaction on the serial flash interface.
23–17 Reserved	This field is reserved.
16 Reserved	This field is reserved.
IDATSZ	IP data transfer size. Defines the data transfer size in bytes of the IP command.

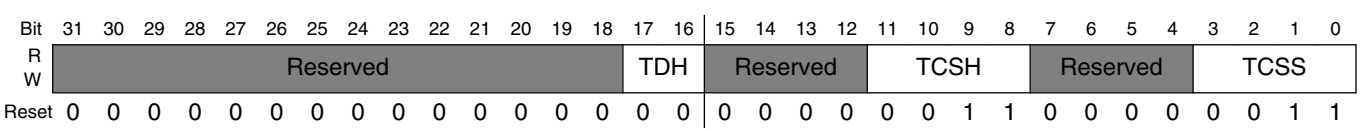
20.4.2.3 Flash Configuration Register (QuadSPIx_FLSHCR)

The Flash configuration register contains the flash device specific timings that must be met by the QuadSPI controller for the device to function correctly.

Write:

- *QSPI_SR[AHB_ACC] = 0*
- *QSPI_SR[IP_ACC] = 0*

Address: 410A_5000h base + Ch offset = 410A_500Ch



QuadSPIx_FLSHCR field descriptions

Field	Description
31–18 Reserved	This field is reserved.
17–16 TDH	Serial flash data in hold time. This helps in meeting the Data In Hold time requirement of a flash. This is valid only in DDR mode. **NOTE:** This field should be set to 0x00 in SDR mode (QuadSPI_MCR[DDR_EN]=0). Refer to Data input hold requirement of Flash for details. The valid TDH programming options are shown below. Other combinations are invalid. 00 Data aligned with the posedge of Internal reference clock of QuadSPI 01 Data aligned with 2x serial flash half clock 10 Data aligned with 4x serial flash half clock
15–12 Reserved	This field is reserved.
11–8 TCSH	Serial flash CS hold time in terms of serial flash clock cycles. Default value '3' should be used in case AHB prefetch size is less than or equal to 32-bytes DDR/16-bytes SDR. **NOTE:** The actual delay between chip select and clock is defined as: • TCSH = 1 SCK clk if N= 0/1 else, N SCK clk if N>1, where N is the setting of TCSH.
7–4 Reserved	This field is reserved. Reserved.
TCSS	Serial flash CS setup time in terms of serial flash clock cycles. **NOTE:** 1. The actual delay between chip select and clock is defined as: • TCSS = 0.5 SCK clk if N= 0/1 else, N+0.5 SCK clk if N>1, where N is the setting of TCSS. 2. Any update to the TCSS register bits is visible on the flash interface only from the second transaction following the update.

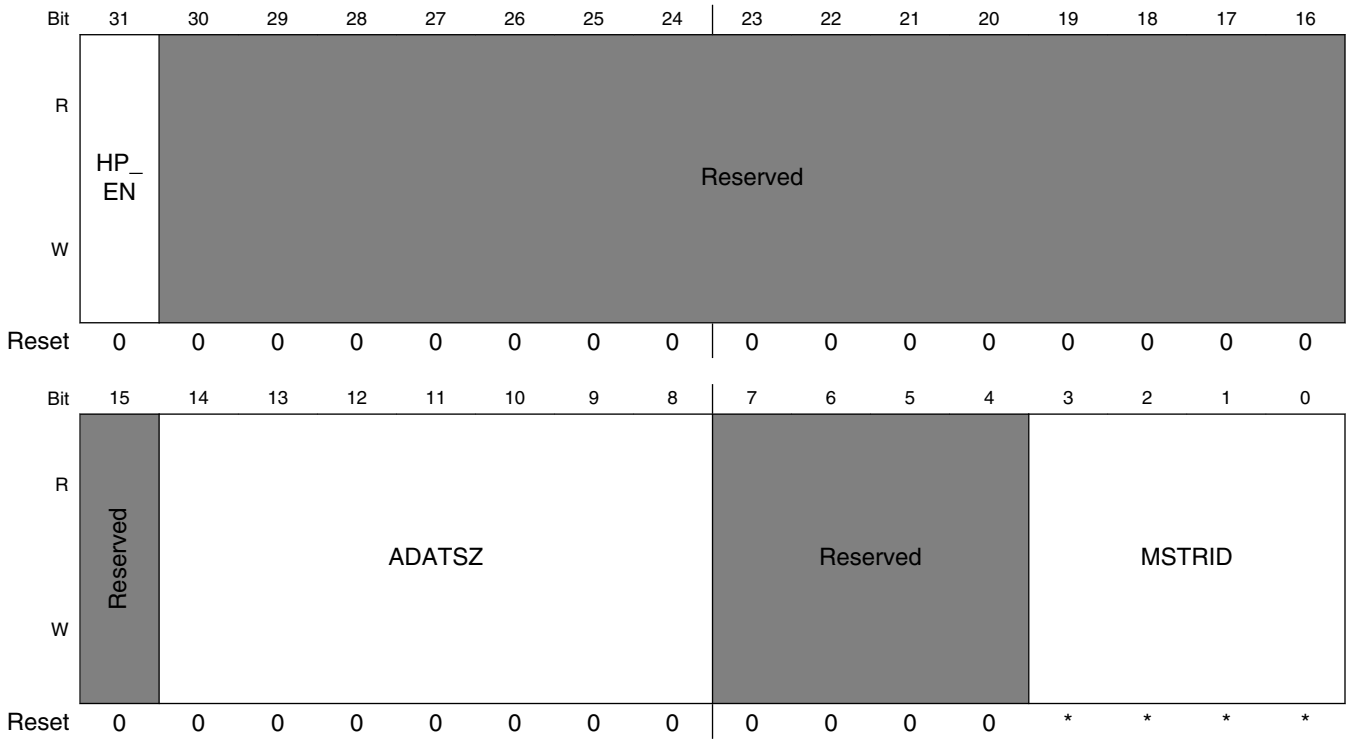
20.4.2.4 Buffer0 Configuration Register (QuadSPIx_BUF0CR)

This register provides the configuration for any access to buffer0. An access is routed to buffer0 when the master port number of the incoming AHB request matches the MSTRID field of BUF0CR. Any buffer "miss" leads to a serial flash transaction being triggered as per the sequence pointed to the SEQID field. Buffer0 may also be configured as a high priority buffer by setting the HP_EN field of this register.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 10h offset = 410A_5010h



* Notes:

- MSTRID field: See the module configuration for reset value.

QuadSPIx_BUF0CR field descriptions

Field	Description
31 HP_EN	High Priority Enable. When set, the master associated with this buffer is assigned a priority higher than the rest of the masters. An access by a high priority master will suspend any ongoing prefetch by another AHB master and will be serviced on high priority. Refer to Flexible AHB Buffers for details.
30–16 Reserved	This field is reserved.
15 Reserved	This field is reserved.
14–8 ADATSZ	AHB data transfer size Defines the data transfer size in 8 bytes of an AHB triggered access to serial flash. For example, a value of 0x2 will set transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7–4 Reserved	This field is reserved.
MSTRID	Master ID. The ID of the AHB master associated with BUFFER0. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

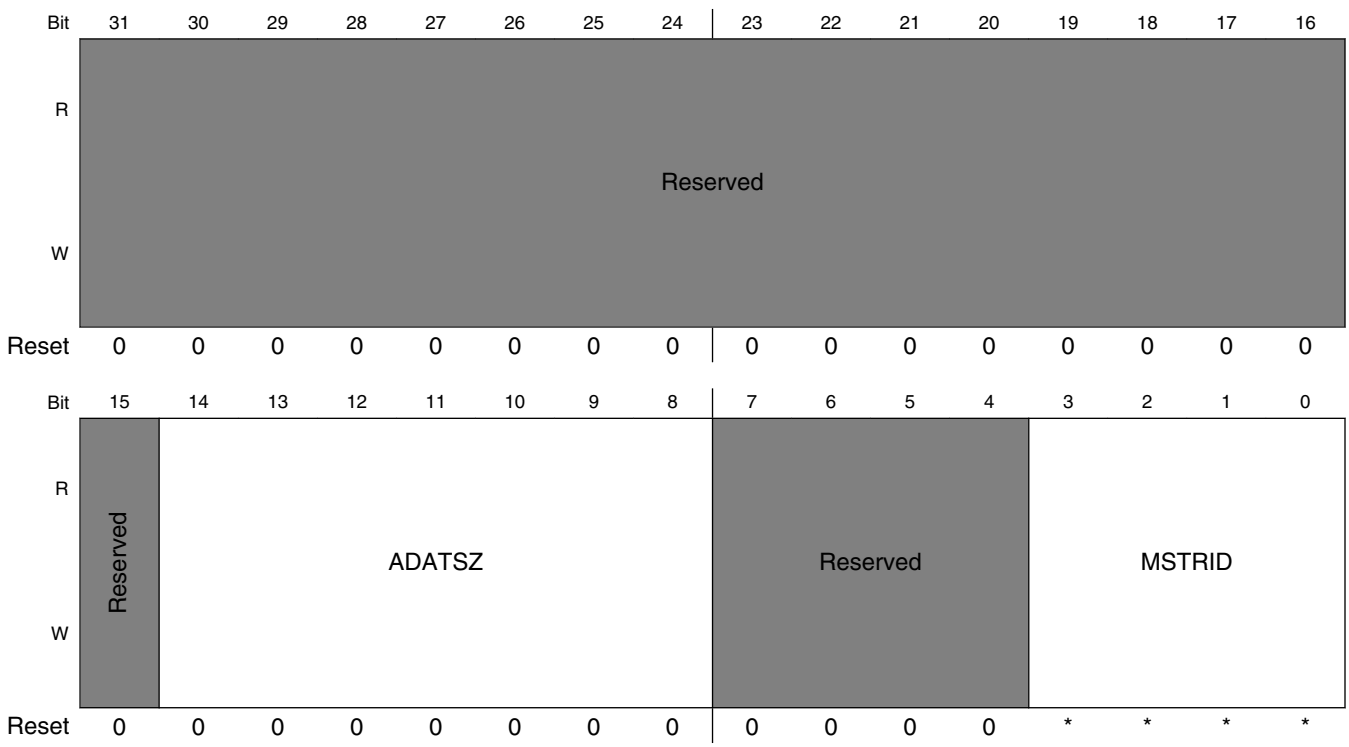
20.4.2.5 Buffer1 Configuration Register (QuadSPIx_BUF1CR)

This register provides the configuration for any access to buffer1. An access is routed to buffer1 when the master port number of the incoming AHB request matches the MSTRID field of BUF1CR. Any buffer "miss" leads to the buffer being flushed and a serial flash transaction being triggered as per the sequence pointed to by the SEQID field.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 14h offset = 410A_5014h



* Notes:

- MSTRID field: See the module configuration for reset value.

QuadSPIx_BUF1CR field descriptions

Field	Description
31–16 Reserved	This field is reserved.
15 Reserved	This field is reserved.
14–8 ADATSZ	AHB data transfer size Defines the data transfer size in 8 bytes of an AHB triggered access to serial flash.For example, a value of 0x2 will set transfer size to 16bytes. When ADATSZ = 0, the data size mentioned the sequence pointed to

Table continues on the next page...

QuadSPIx_BUF1CR field descriptions (continued)

Field	Description
	by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7–4 Reserved	This field is reserved.
MSTRID	Master ID. The ID of the AHB master associated with BUFFER1. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

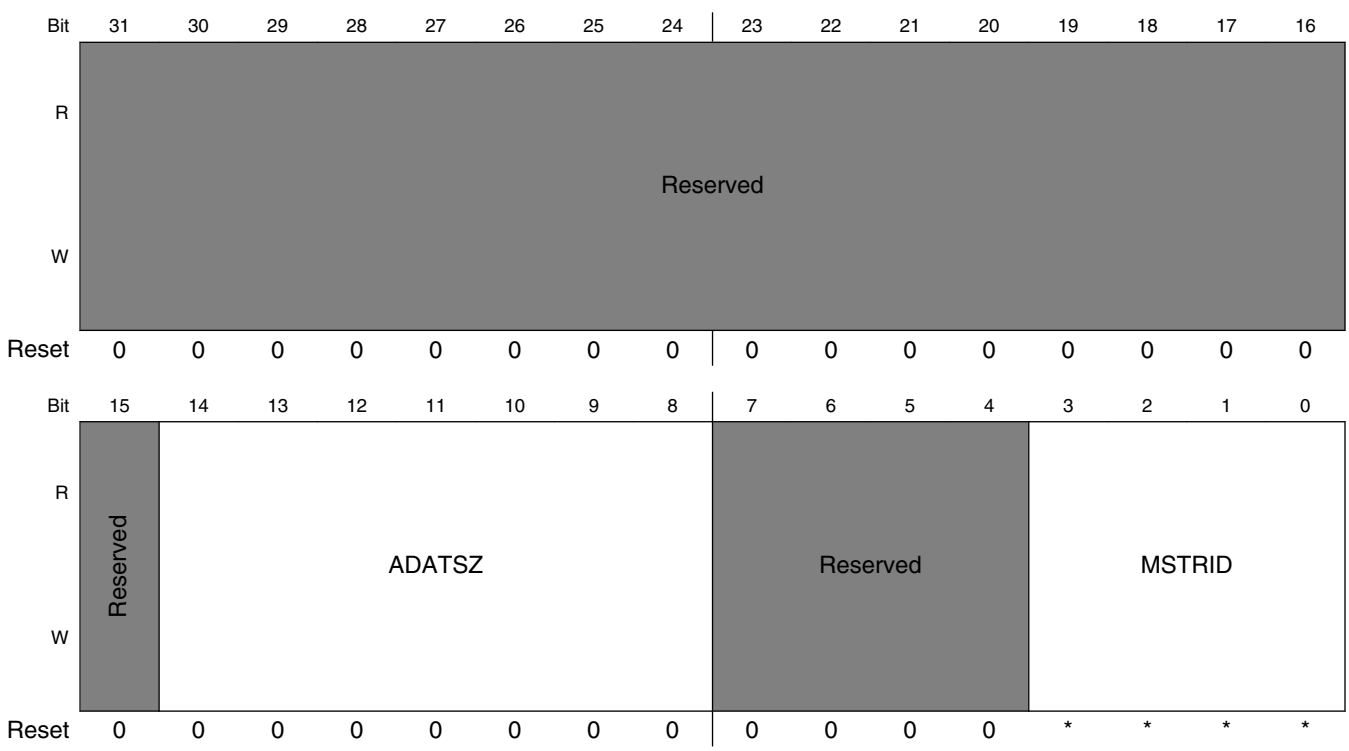
20.4.2.6 Buffer2 Configuration Register (QuadSPIx_BUF2CR)

This register provides the configuration for any access to buffer2. An access is routed to buffer2 when the master port number of the incoming AHB request matches the MSTRID field of BUF2CR. Any buffer "miss" leads to the buffer being flushed and a serial flash transaction being triggered as per the sequence pointed to by the SEQID field.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 18h offset = 410A_5018h



* Notes:

- MSTRID field: See the module configuration for the device specific reset value.

QuadSPIx_BUF2CR field descriptions

Field	Description
31–16 Reserved	This field is reserved. Reserved.
15 Reserved	This field is reserved.
14–8 ADATSZ	AHB data transfer size Defines the data transfer size in 8 bytes of an AHB triggered access to serial flash. For example, a value of 0x2 will set transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7–4 Reserved	This field is reserved. Reserved.
MSTRID	Master ID. The ID of the AHB master associated with BUFFER2. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

20.4.2.7 Buffer3 Configuration Register (QuadSPIx_BUF3CR)

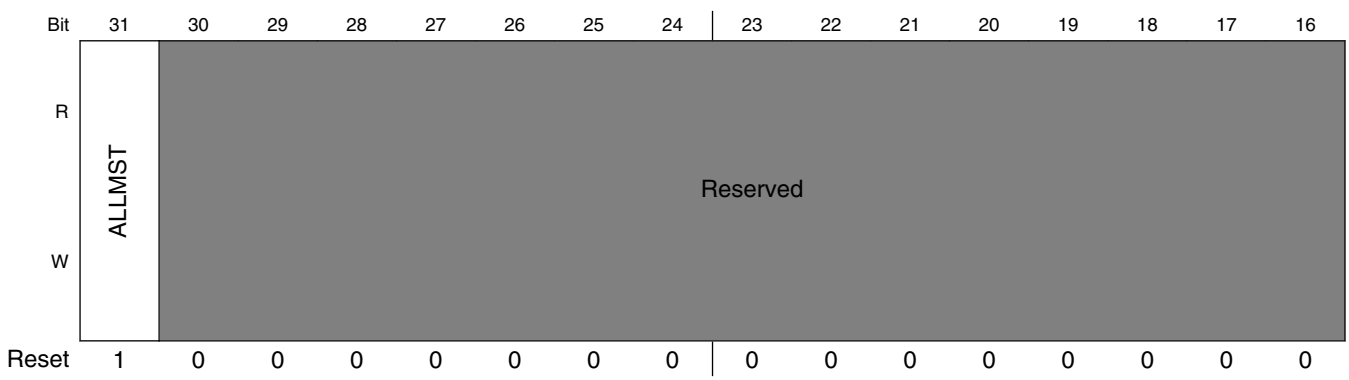
This register provides the configuration for any access to buffer3. An access is routed to buffer3 when the master port number of the incoming AHB request matches the MSTRID field of BUF3CR. Any buffer "miss" leads to the buffer being flushed a serial flash transaction being triggered as per the sequence pointed to by the SEQID field.

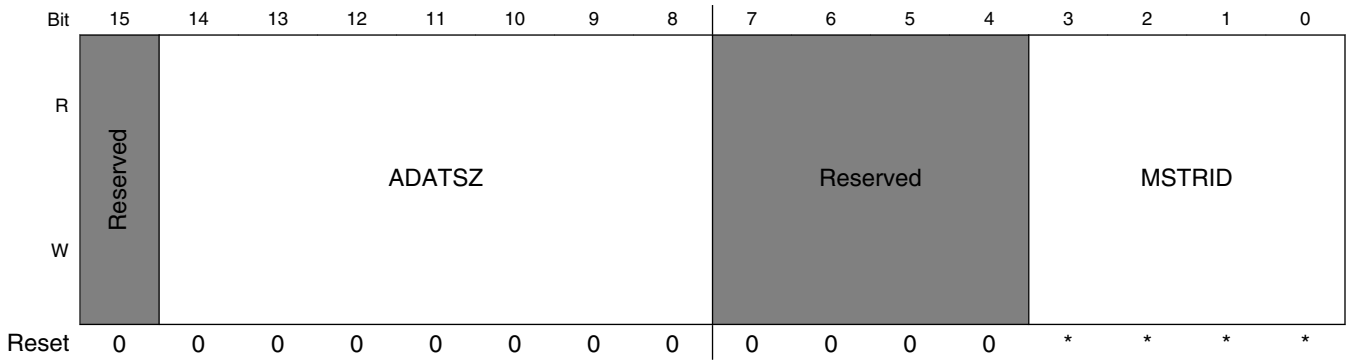
In the case that the ALLMST field is not set, any such transaction (where master port number does not match any of the MSTRID fields) will be returned an ERROR response.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 1Ch offset = 410A_501Ch





* Notes:
- MSTRID field: See the module configuration for the device specific reset value.

QuadSPIx_BUF3CR field descriptions

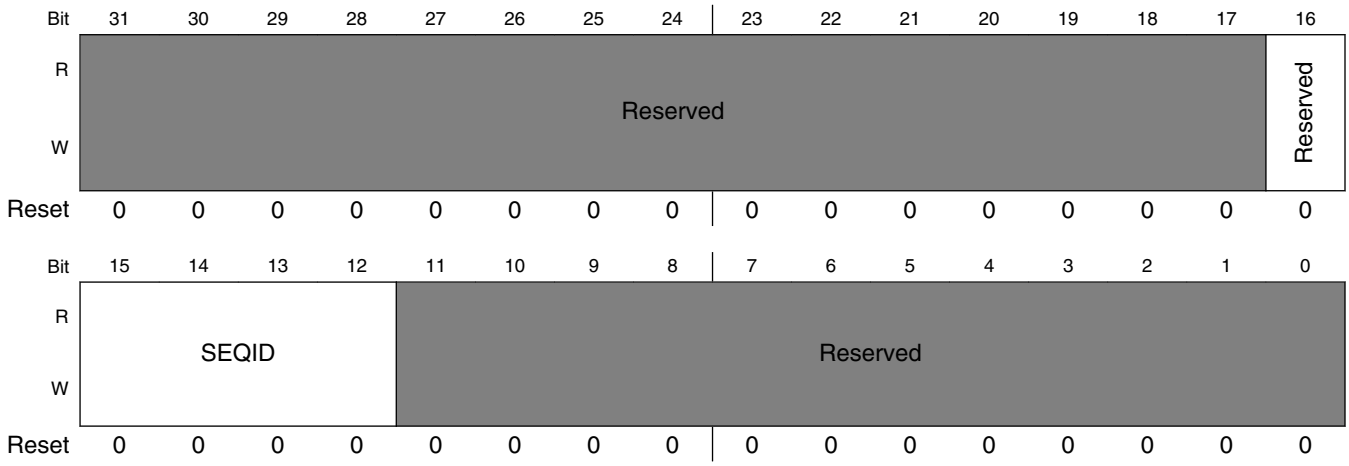
Field	Description
31 ALLMST	All master enable. When set, buffer3 acts as an all-master buffer. Any AHB access with a master port number not matching with the master ID of buffer0 or buffer1 or buffer2 is routed to buffer3. When set, the MSTRID field of this register is ignored.
30–16 Reserved	This field is reserved. Reserved.
15 Reserved	This field is reserved.
14–8 ADATSZ	AHB data transfer size Defines the data transfer size in 8 Bytes of an AHB triggered access to serial flash. When ADATSZ = 0, the data size mentioned the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer.
7–4 Reserved	This field is reserved.
MSTRID	Master ID. The ID of the AHB master associated with BUFFER3. Any AHB access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers must be different.

20.4.2.8 Buffer Generic Configuration Register (QuadSPIx_BFGENCR)

This register provides the generic configuration to any of the buffer accesses. Any buffer "miss" leads to the buffer being flushed and a serial flash transaction being triggered as per the sequence pointed to by the SEQID field.

Write:
- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 20h offset = 410A_5020h



QuadSPIx_BFGENCR field descriptions

Field	Description
31–17 Reserved	This field is reserved.
16 Reserved	This field is reserved.
15–12 SEQID	Points to a sequence in the Look-up-table. The SEQID defines the bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. Refer to Look-up Table. **NOTE:** If the sequence pointer differs between the new and previous sequence then the user should reset this. See QSPI_SPTRCLR for more information.
Reserved	This field is reserved.

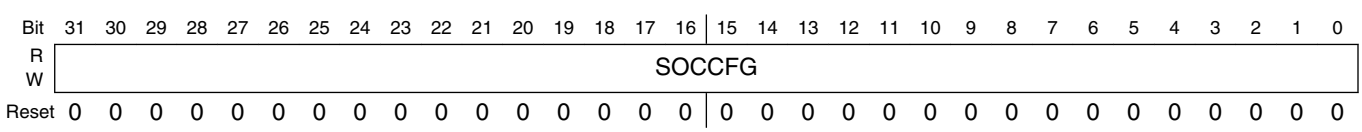
20.4.2.9 SOC Configuration Register (QuadSPIx_SOCCR)

This register is programmed at chip level for QuadSPI delay chain configuration. For details, refer to chip-specific QuadSPI information.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 24h offset = 410A_5024h



QuadSPIx_SOCCR field descriptions

Field	Description
SOCCFG	SOC Configuration

QuadSPIx_SOCCR field descriptions (continued)

Field	Description
	For details, refer to chip-specific QuadSPI information.

20.4.2.10 Buffer0 Top Index Register (QuadSPIx_BUF0IND)

This register specifies the top index of buffer0, which defines its size. Note that that the 3 LSBs of this register are set to zero. This ensures that the buffer is 64-bit aligned, as each buffer entry is 64 bits long.

The register value should be set to the desired number of bytes. For example, setting BUF0IND[31:3] to 0 gives 0 bytes, 1 gives 8 bytes etc.

The size of buffer0 is the difference between BUF0IND and 0.

Software should ensure that BUF0IND value is not greater than the overall size of the buffer. The hardware does not provide any protection against illegal programming.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 30h offset = 410A_5030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TPINDX0																													Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

QuadSPIx_BUF0IND field descriptions

Field	Description
31–3 TPINDX0	Top index of buffer 0.
Reserved	This field is reserved. Reserved.

20.4.2.11 Buffer1 Top Index Register (QuadSPIx_BUF1IND)

This register specifies the top index of buffer1, which defines its size. Note that the 3 LSBs of this register are set to zero. This ensures that the buffer is 64-bit aligned as each buffer entry is 64 bits long.

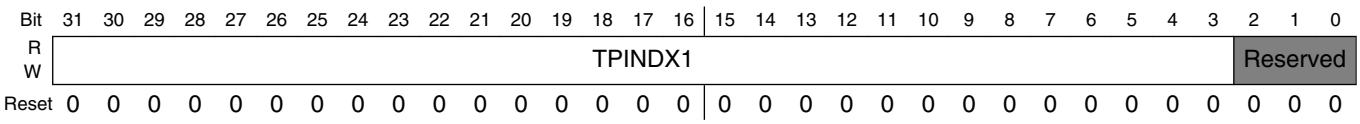
The size of buffer1 is the difference between BUF1IND and BUF0IND. The register value should be entered in bytes. For example, If BUF0IND = 0x100 then setting BUF1IND = 0x130 will set buffer1 size to 0x30 bytes.

It is the responsibility of the software to ensure that BUF1IND value is not greater than the overall size of the buffer. The hardware does not provide any protection against illegal programming.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 34h offset = 410A_5034h



QuadSPIx_BUF1IND field descriptions

Field	Description
31–3 TPINDX1	Top index of buffer 1.
Reserved	This field is reserved.

20.4.2.12 Buffer2 Top Index Register (QuadSPIx_BUF2IND)

This register specifies the top index of buffer2, which defines its size. Note that that the 3 LSBs of this register are set to zero. This ensures that the buffer is 64-bit aligned as each buffer entry is 64 bits long.

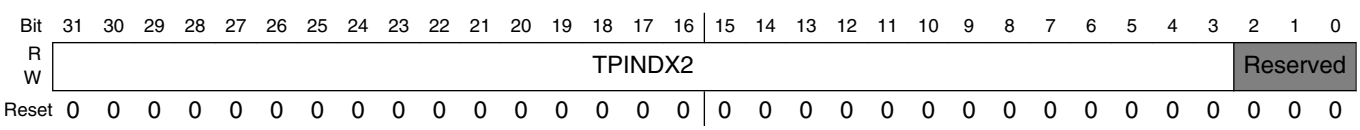
The size of buffer2 is the difference between the BUF2IND and BUF1IND. The register value should be entered in bytes. For example, if BUF1IND = 0x130 then setting BUF2IND = 0x180 will set buffer2 size to 0x50 bytes.

It is the responsibility of the software to ensure that BUF2IND value is not greater than the overall size of the buffer. The hardware does not provide any protection against illegal programming.

Write:

- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 38h offset = 410A_5038h



QuadSPIx_BUF2IND field descriptions

Field	Description
31–3 TPINDX2	Top index of buffer 2.
Reserved	This field is reserved.

20.4.2.13 Serial Flash Address Register (QuadSPIx_SFAR)

The module automatically translates this address on the memory map to the address on the flash itself. When operating in 24-bit mode, only bits 23-0 are sent to the flash. In 32-bit mode, bits 27-0 are used with bits 31-28 driven to 0 when QSPI_SFACR[CAS] is set to 0. Say, if QSPI_SFACR[CAS] is 3 then bits 26-3 are sent to flash as it page address in case flash is operating in 24-bit mode. Total number of address bits request by flash as it page and column address must not be more than 32 bit. Refer to Table 20-11 for the mapping between the access mode and the QSPI_SFAR content and to Normal Mode for details about the command triggering and command execution. The software should ensure that the serial flash address provided in the QSPI_SFAR register lies in the valid flash address range as defined in Table 20-11.

Write:

- *QSPI_SR[IP_ACC] = 0*

Address: 410A_5000h base + 100h offset = 410A_5100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SFADR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

QuadSPIx_SFAR field descriptions

Field	Description
SFADR	Serial Flash Address. The register content is used as byte address for all following IP Commands.

20.4.2.14 Serial Flash Address Configuration Register (QuadSPIx_SFACR)

This register contains the serial flash specific address requirements that must be configured according to the flash connected, for the controller to function properly. The module automatically translates the address QSPI_SFAR on the memory map or the incoming address on the AHB bus to the column address on the flash itself. Say, a flash needs 3 bits as its column address than only the lower 3 bits of QSPI_SFAR/AHB address are send to flash as its column address. The software should ensure that the serial flash address provided in the QSPI_SFAR register or the incoming AHB address lies in the valid flash address range.

Write:

- *QSPI_SR[IP_ACC] = 0*
- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 104h offset = 410A_5104h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															WA
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												CAS			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

QuadSPIx_SFACR field descriptions

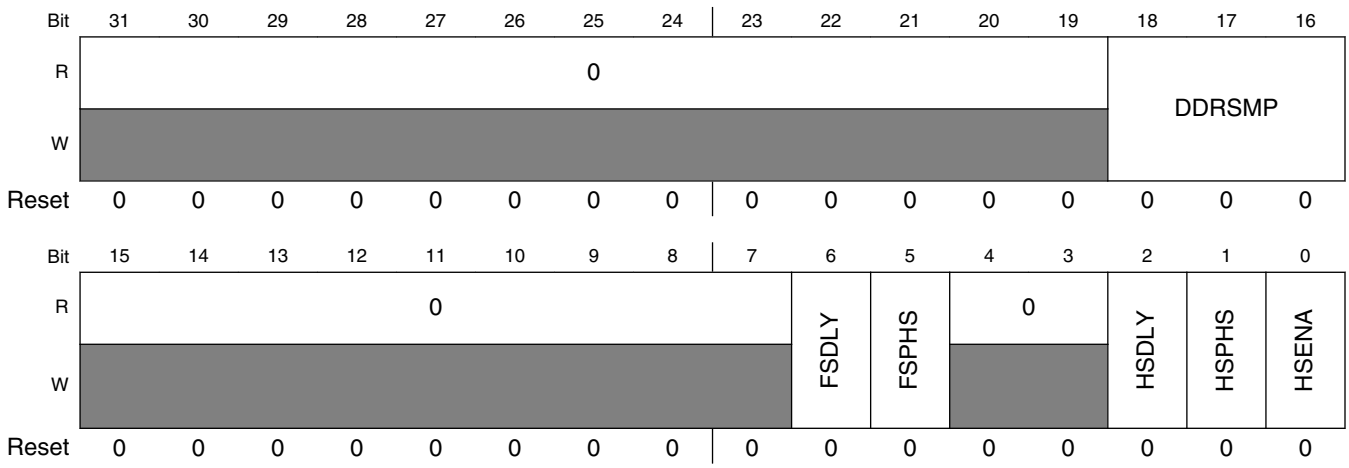
Field	Description
31–17 Reserved	This field is reserved.
16 WA	Word Addressable Defines whether the serial flash is a byte addressable flash or a word addressable flash. According to this bit configuration the address is re-mapped to the flash interface. Refer to Address scheme for details. 0 Byte addressable serial flash mode. 1 Word (2 byte) addressable serial flash mode.
15–4 Reserved	This field is reserved.
CAS	Column Address Space Defines the width of the column address. If the coulmn address is say [2:0] of QSPI_SFAR/AHB address, then CAS must be 3. If there is no column address separation in any serial flash this bit must be programmed to 0.

20.4.2.15 Sampling Register (QuadSPIx_SMPR)

The Sampling Register allows configuration of how the incoming data from the external serial flash devices are sampled in the QuadSPI module.

Write: Disabled Mode

Address: 410A_5000h base + 108h offset = 410A_5108h



QuadSPIx_SMPR field descriptions

Field	Description
31–19 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
18–16 DDRSMP	DDR Sampling point Select the sampling point for incoming data when serial flash is executing a DDR instruction. Refer to Figure 20-10 for details on the sampling points.
15–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 FSDLY	Full Speed Delay selection for SDR instructions. Select the delay with respect to the reference edge for the sample point valid for full speed commands. 0 One clock cycle delay 1 Two clock cycles delay.
5 FSPHS	Full Speed Phase selection for SDR instructions. Select the edge of the sampling clock valid for full speed commands. 0 Select sampling at non-inverted clock 1 Select sampling at inverted clock.
4–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 HSDLY	Half Speed Delay selection for SDR instructions.

Table continues on the next page...

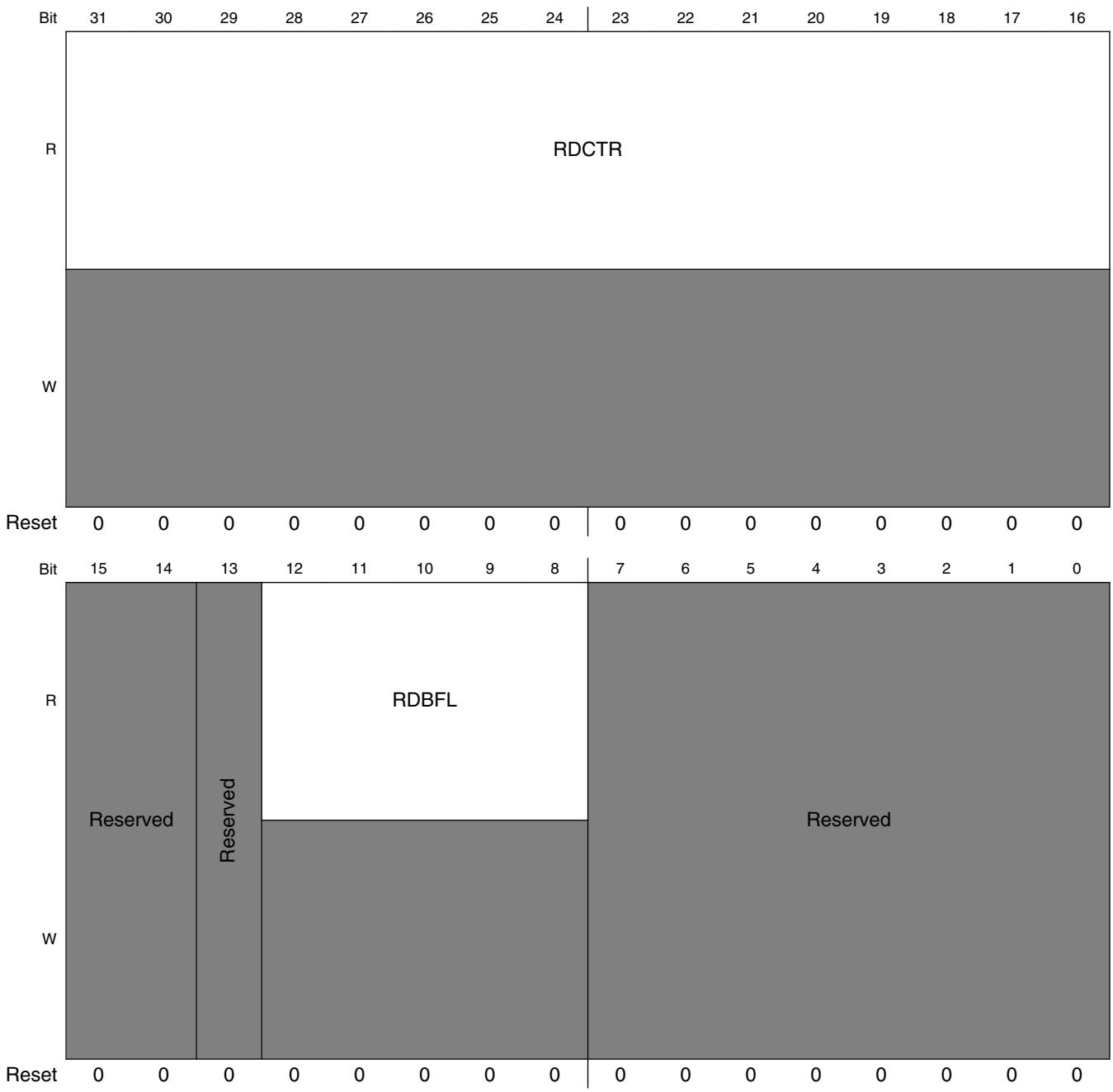
QuadSPIx_SMPR field descriptions (continued)

Field	Description
	Only relevant when HSENA bit is set. Select the delay with respect to the reference edge for the sample point valid for half speed commands. 0 One clock cycle delay 1 Two clock cycle delay
1 HSPHS	Half Speed Phase selection for SDR instructions. Only relevant when HSENA bit is set. Select the delay with respect to the reference edge for the sample point valid for half speed commands. 0 Select sampling at non-inverted clock 1 Select sampling at inverted clock
0 HSENA	Half Speed serial flash clock Enable This bit enables the divide by 2 of the clock to the external serial flash device for all commands, only in SDR. Refer to Serial Flash Clock Frequency Limitations for details. 0 Disable divide by 2 of serial flash clock for half speed commands 1 Enable divide by 2 of serial flash clock for half speed commands

20.4.2.16 RX Buffer Status Register (QuadSPIx_RBSR)

This register contains information related to the receive data buffer.

Address: 410A_5000h base + 10Ch offset = 410A_510Ch



QuadSPIx_RBSR field descriptions

Field	Description
31–16 RDCTR	Read Counter. Indicates how many entries of 4 bytes have been removed from the RX Buffer. For example, a value of 0x2 would indicate 8 bytes have been removed. It is incremented by the number (QSPI_RBCT[WMRK] + 1) on RX Buffer POP event. The RX Buffer can be popped using DMA or pop flag QSPI_FR[RBDF]. The QSPI_RSER[RBDDE] defines which pop has to be done. For further details, refer to AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB15) and "Data Transfer from the QuadSPI Module Internal Buffers" section in Flash Read.
15–14 Reserved	This field is reserved.
13 Reserved	This field is reserved.
12–8 RDBFL	RX Buffer Fill Level. Indicates how many entries of 4 bytes are still available in the RX Buffer.For example, a value of 0x2 would indicate 8bytes are available.
Reserved	This field is reserved.

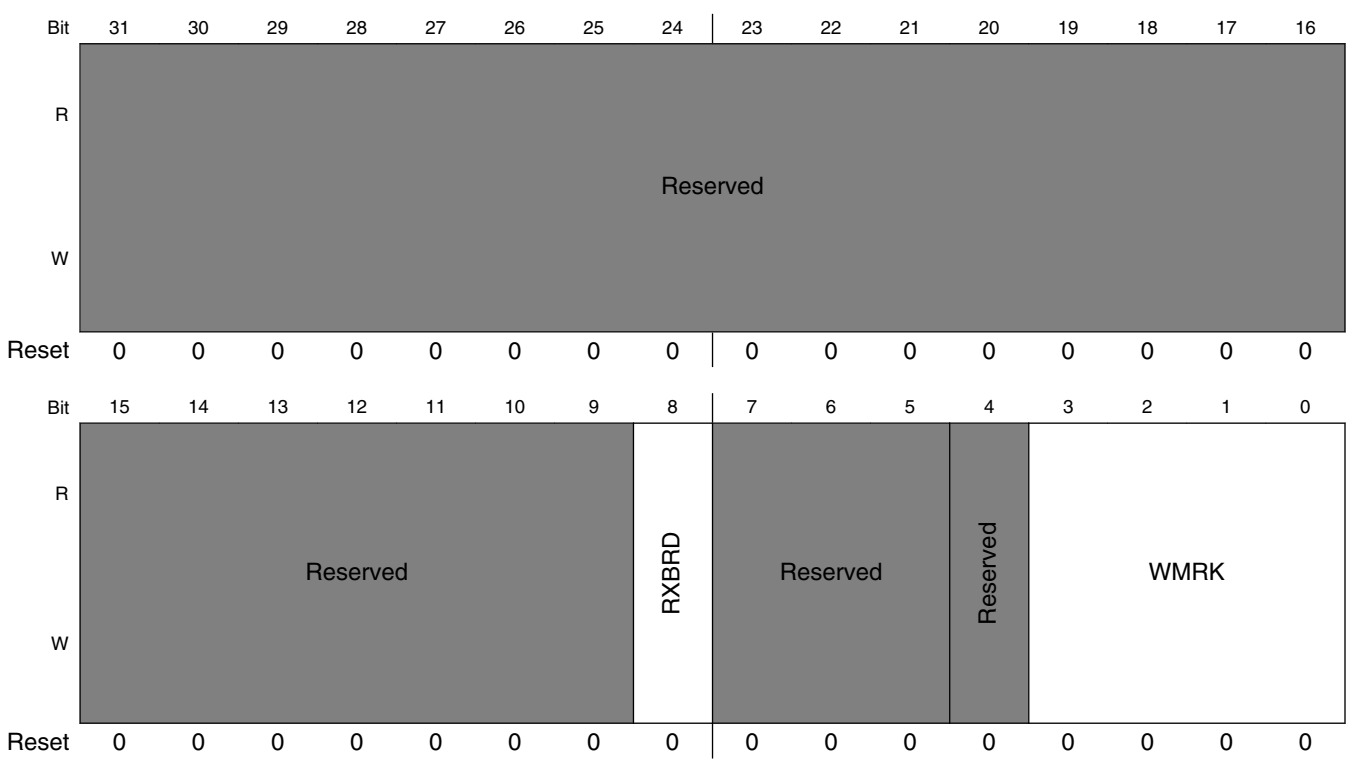
20.4.2.17 RX Buffer Control Register (QuadSPIx_RBCT)

This register contains control data related to the receive data buffer.

Write:

- *QSPI_SR[IP_ACC] = 0*

Address: 410A_5000h base + 110h offset = 410A_5110h



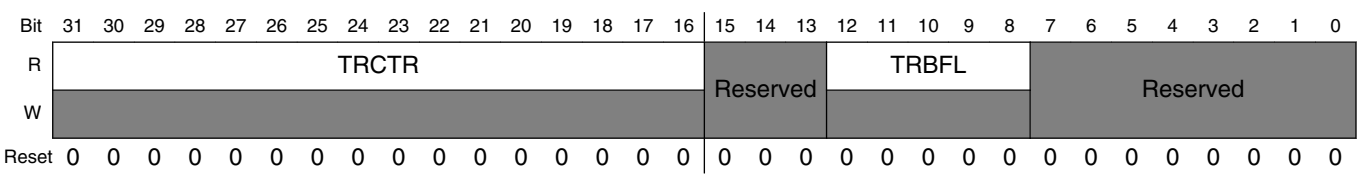
QuadSPIx_RBCT field descriptions

Field	Description
31–9 Reserved	This field is reserved.
8 RXBRD	RX Buffer Readout. This field specifies the access scheme for the RX Buffer readout. 0 RX Buffer content is read using the AHB Bus registers QSPI_ARDB0 to QSPI_ARDB15 . For details, refer to Exclusive Access to Serial Flash for AHB Commands. 1 RX Buffer content is read using the IP Bus registers QSPI_RBDR0 to QSPI_RBDR15 .
7–5 Reserved	This field is reserved.
4 Reserved	This field is reserved.
WMRK	RX Buffer Watermark. This field determines when the readout action of the RX Buffer is triggered. When the number of valid entries in the RX Buffer is equal to or greater than the number given by (WMRK+1) the QSPI_SR[RXWE] flag is asserted.The value should be entered as the number of 4-byte entries minus 1. For example, a value of 0x0 would set the watermark to 4 bytes, 1 to 8 bytes, 2 to 12 bytes, and so on. For details, refer to DMA Usage.

20.4.2.18 TX Buffer Status Register (QuadSPIx_TBSR)

This register contains information related to the transmit data buffer.

Address: 410A_5000h base + 150h offset = 410A_5150h



QuadSPIx_TBSR field descriptions

Field	Description
31–16 TRCTR	Transmit Counter. This field indicates how many entries of 4 bytes have been written into the TX Buffer by host accesses. It is reset to 0 when a 1 is written to QSPI_MCR[CLR_TXF]. It is incremented on each write access to the QSPI_TBDR register when another word has been pushed onto the TX Buffer. When it is not cleared the TRCTR field wraps around to 0. Refer to TX Buffer Data Register (QuadSPI_TBDR) for details.
15–13 Reserved	This field is reserved.
12–8 TRBFL	TX Buffer Fill Level. The TRBFL field contains the number of entries of 4 bytes each available in the TX Buffer for the QuadSPI module to transmit to the serial flash device.
Reserved	This field is reserved.

20.4.2.19 TX Buffer Data Register (QuadSPIx_TBDR)

The QSPI_TBDR register provides access to the circular TX Buffer of depth 64 bytes . This buffer provides the data written into it as write data for the page programming commands to the serial flash device. Refer to Table 20-17 for the byte ordering scheme. A write transaction on the flash with data size of less than 32 bits will lead to the removal of four data entry from the TX buffer. The valid bits will be used and the rest of the bits will be discarded.

Write:

- *QSPI_SR[TXFULL] = 0*

32-bit write access required

Address: 410A_5000h base + 154h offset = 410A_5154h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXDATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

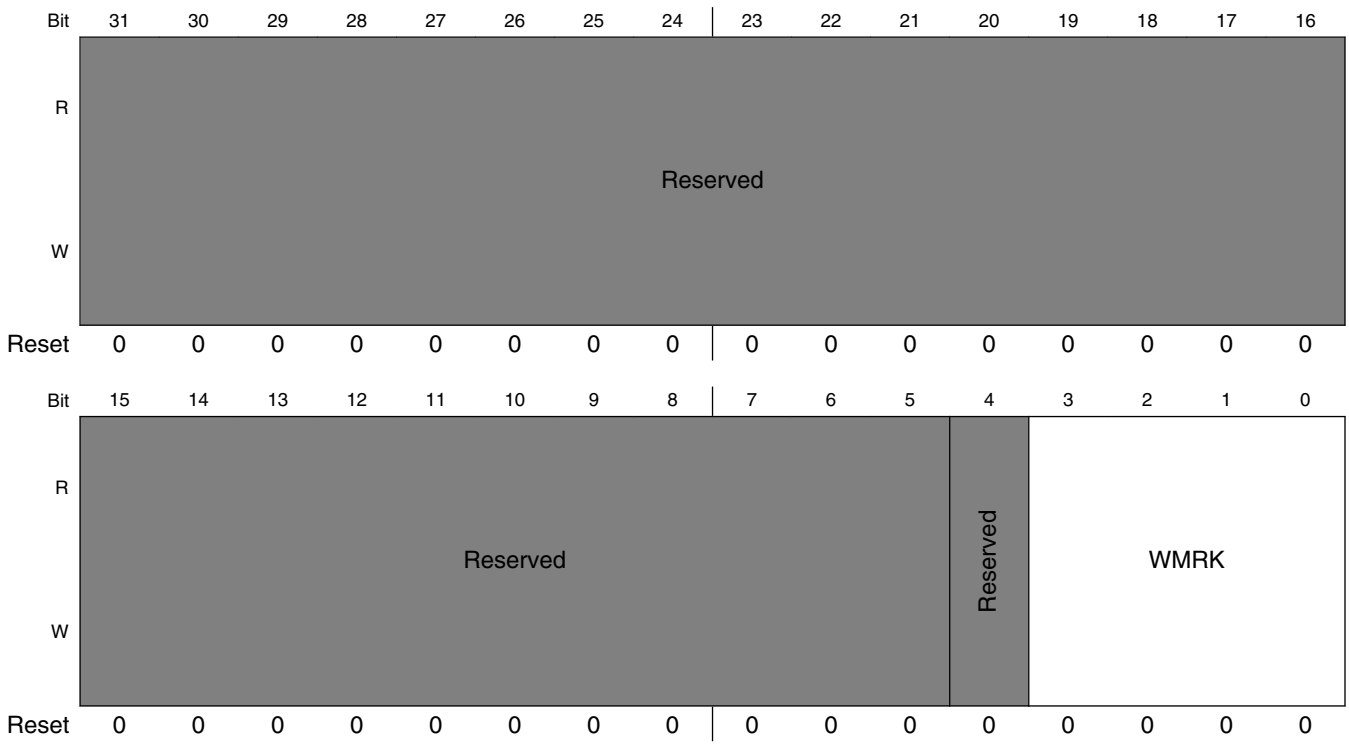
QuadSPIx_TBDR field descriptions

Field	Description
TXDATA	TX Data On write access the data is written into the next available entry of the TX Buffer and the QPSI_TBSR[TRBFL] field is updated accordingly. On a read access, the last data written to the register is returned.

20.4.2.20 Tx Buffer Control Register (QuadSPIx_TBCT)

This register contains control information for transmit data buffer.

Address: 410A_5000h base + 158h offset = 410A_5158h



QuadSPIx_TBCT field descriptions

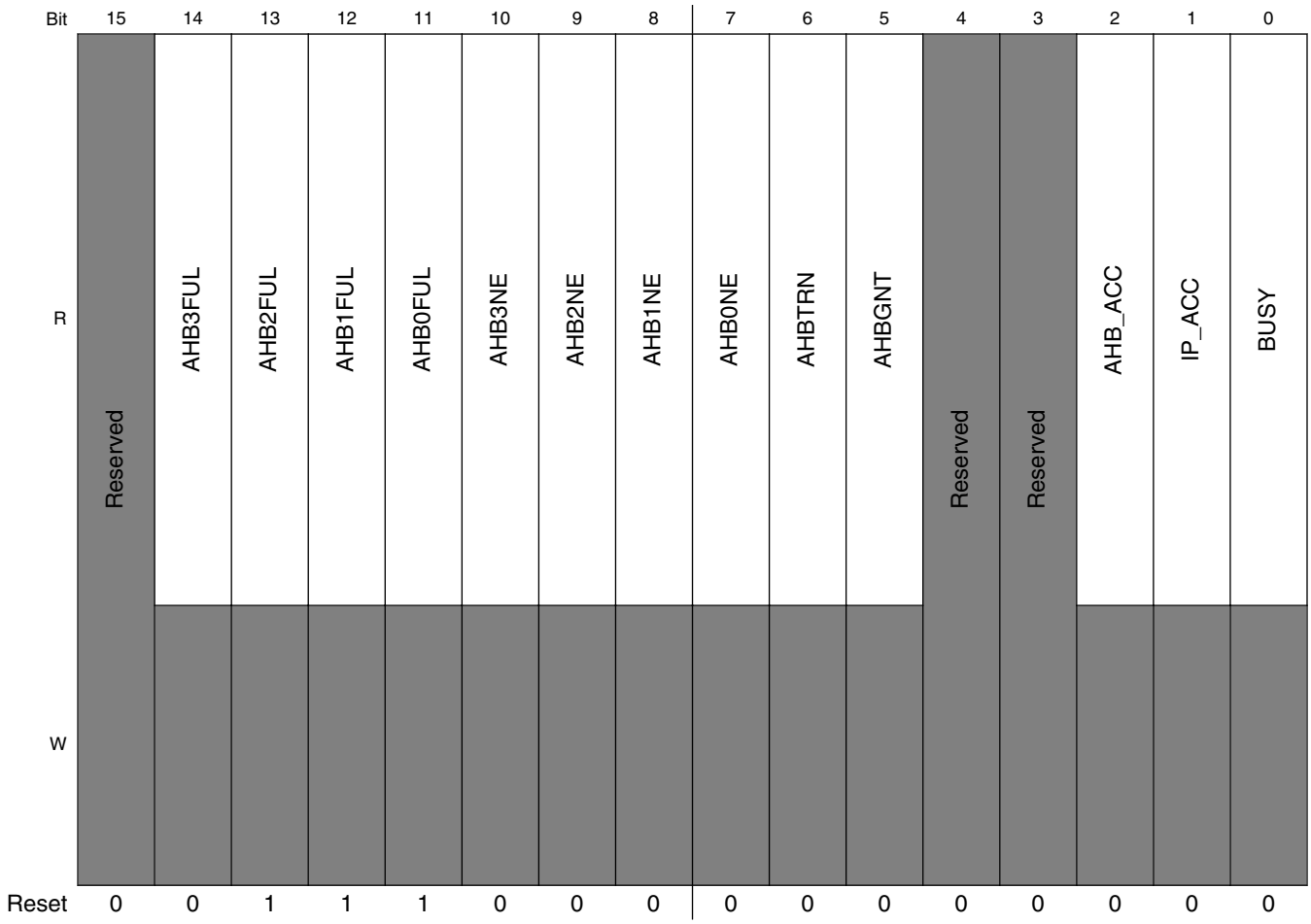
Field	Description
31–5 Reserved	This field is reserved.
4 Reserved	This field is reserved.
WMRK	Determines the watermark for the TX Buffer. When the number of available space in TX Buffer is greater than the number given by (WMRK+1), QSPI_SR[TXWA] is asserted. The values should be entered as the number of 4Bytes entries minus 1. For example, a value of 0x0 would set the watermark to 4 bytes, 1 to 8 bytes, 2 to 12 Bytes, and so on.For details, refer to DMA Usage.

20.4.2.21 Status Register (QuadSPIx_SR)

The QSPI_SR register provides all available status information about SFM command execution and arbitration, the RX Buffer, TX Buffer, and the AHB Buffer.

Address: 410A_5000h base + 15Ch offset = 410A_515Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPSMP			Reserved	TXFULL	TXDMA	TXWA	TXEDA	RXDMA	Reserved			RXFULL	Reserved		RXWE
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0



QuadSPIx_SR field descriptions

Field	Description
31–29 DLPSMP	Data learning pattern sampling point. The sampling point found by the controller with the data learning pattern. • This is used for DDR only. • If the learning fails, this field will return garbage and DLPFF bit will be set. • In case of Data learning with DQS this field will return the reset value as sampling point match is not found in case of DQS. For details, refer to Data Learning.
28 Reserved	This field is reserved.
27 TXFULL	TX Buffer Full. Asserted when no more data can be stored.
26 TXDMA	TXDMA Asserted when TXFIFO fill via DMA is active i.e. DMA is requested or running
25 TXWA	TX Buffer watermark Available Asserted when the number of available spaces in TX buffer is greater than or equal to the value give by QSPI_TBCT[WMRK].

Table continues on the next page...

QuadSPIx_SR field descriptions (continued)

Field	Description
24 TXEDA	Tx Buffer Enough Data Available Asserted when TX Buffer contains enough data for any pop operation to take place. There must be at least 128 bit data available in TX FIFO for any pop operation; otherwise, QSPI_FR[TBUF] will be set.
23 RXDMA	RX Buffer DMA. Asserted when RX Buffer read out via DMA is active i.e DMA is requested or running.
22–20 Reserved	This field is reserved.
19 RXFULL	RX Buffer Full. Asserted when the RX Buffer is full, i.e. that QSPI_RBSR[RDBFL] field is equal to 32.
18–17 Reserved	This field is reserved.
16 RXWE	RX Buffer Watermark Exceeded. Asserted when the number of valid entries in the RX Buffer exceeds the number given in the QSPI_RBCT[WMRK] field.
15 Reserved	This field is reserved.
14 AHB3FUL	AHB 3 Buffer Full. Asserted when AHB 3 buffer is full.
13 AHB2FUL	AHB 2 Buffer Full. Asserted when AHB 2 buffer is full.
12 AHB1FUL	AHB 1 Buffer Full. Asserted when AHB 1 buffer is full.
11 AHB0FUL	AHB 0 Buffer Full. Asserted when AHB 0 buffer is full.
10 AHB3NE	AHB 3 Buffer Not Empty. Asserted when AHB 3 buffer contains data.
9 AHB2NE	AHB 2 Buffer Not Empty. Asserted when AHB 2 buffer contains data.
8 AHB1NE	AHB 1 Buffer Not Empty. Asserted when AHB 1 buffer contains data.
7 AHB0NE	AHB 0 Buffer Not Empty. Asserted when AHB 0 buffer contains data.
6 AHBTRN	AHB Access Transaction pending. Asserted when there is a pending request on the AHB interface. Refer to the AMBA specification for details.
5 AHBGNT	AHB Command priority Granted: Asserted when another module has been granted priority of AHB Commands against IP Commands. For details refer to Command Arbitration.
4 Reserved	This field is reserved.
3 RESERVED	This field is reserved.
2 AHB_ACC	AHB Access. Asserted when the transaction currently executed was initiated by AHB bus.
1 IP_ACC	IP Access. Asserted when transaction currently executed was initiated by IP bus.
0 BUSY	Module Busy. Asserted when module is currently busy handling a transaction to an external flash device.

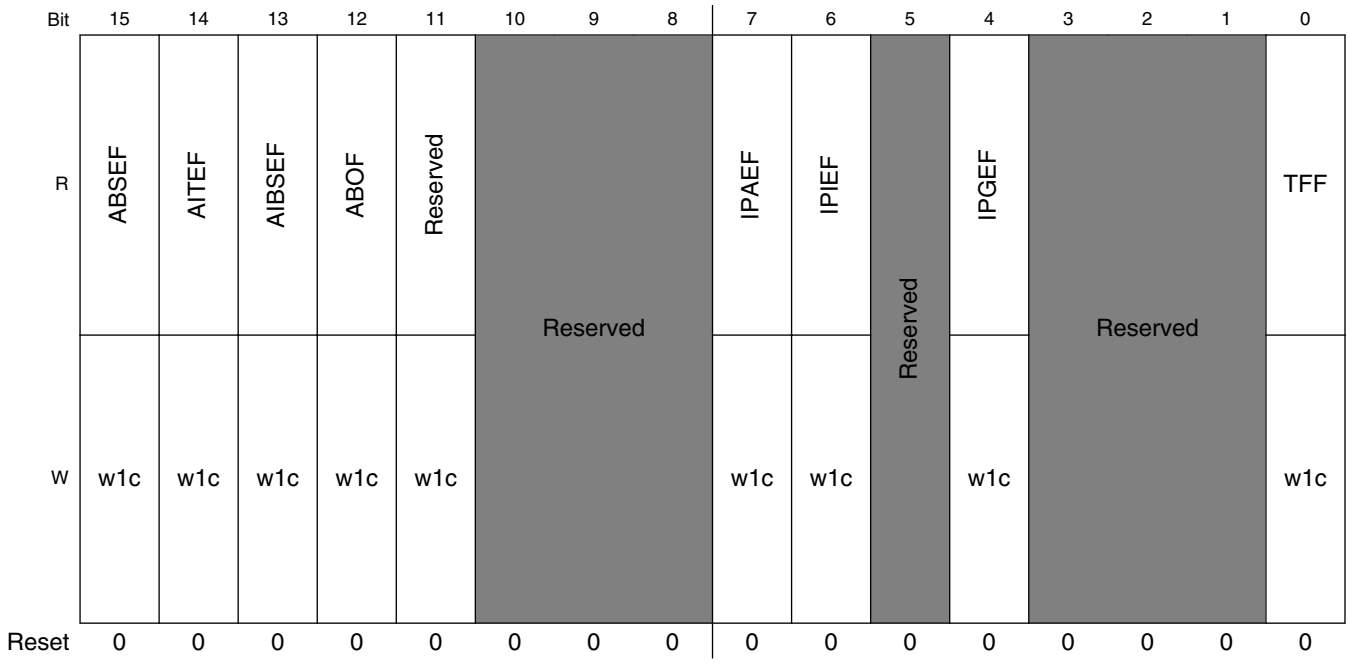
20.4.2.22 Flag Register (QuadSPIx_FR)

The QSPI_FR register provides all available flags about SFM command execution and arbitration which may serve as source for the generation of interrupt service requests. Note that the error flags in this register do not relate directly to the execution of the transaction in the serial flash device itself but only to the behavior and conditions visible in the QuadSPI module.

Write: Enabled Mode

Address: 410A_5000h base + 160h offset = 410A_5160h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFF	Reserved		Reserved	TBFF	TBUF	Reserved		ILLINE	Reserved					RBOF	RBDF
W	w1c				w1c	w1c			w1c						w1c	w1c
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0



QuadSPIx_FR field descriptions

Field	Description
31 DLPFF	Data Learning Pattern Failure Flag. Set when DATA_LEARN instruction was encountered in a sequence but no sampling point was found for the data learning patternin case only 8 bit data learning is requested for non DQS mode. The controller automatically starts sampling using the value in QSPI_SMPR[DDRSMP].If more than 8 bits data learning are requested with QSPI_MCR[DQS_EN] set to 0, and the sampling point found after first 8 bit match doesn't remain the same for the whole instruction duration, this flag is set. In case of Data learn with DQS this flag is set whenever the incoming data from flash on DQS edges doesn't match the pattern in QSPI_DLPR. For details, refer to Data Learning.
30–29 RESERVED	This field is reserved.
28 Reserved	This field is reserved.
27 TBFF	TX Buffer Fill Flag. Before writing to the TX buffer, this bit should be cleared. Then this bit has to be read back. If the bit is set, the TX Buffer can take more data. If the bit remains cleared, the TX buffer is full. Refer to Tx Buffer Operation for details.
26 TBUF	TX Buffer Underrun Flag. Set when the module tried to pull data although TX Buffer was emptyor the buffer contains less than 128 bits of data. The application must ensure that the buffer never goes empty during a transaction except for the last data fetch. The IP Command leading to the TX Buffer underrun is continued (data sent to the serial flash device is all F in case of valid TX underrun). Here valid 'underrun' means that, it should have occurred during the transacting such that few bytes (i.e., less than 4 bytes) are left in FIFO and remaining are filled with “FFFFh". Software should start TX transaction only when 128-bits are written in TX buffer. The application must clear the TX Buffer in response to this event by writing a '1' to the QSPI_MCR[CLR_TXF].
25–24 Reserved	This field is reserved.
23 ILLINE	Illegal Instruction Error Flag. Set when an illegal instruction is encountered by the controller in any of the sequences. As soon as this flag is set, user should assert QSPI_MCR[SWRSTSD], QSPI_MCR[SWRSTHD] and after software resets de-assertion in normal mode, QSPI_MCR[CLR_TXF] and QSPI_MCR[CLR_RXF] should be set to clear any remaining data in buffers i.e., reset to flash and

Table continues on the next page...

QuadSPIx_FR field descriptions (continued)

Field	Description
	AHB domain after re-configuring the correct sequence instruction. Refer to Table 20-15 for a list of legal instructions.
22–18 Reserved	This field is reserved.
17 RBOF	RX Buffer Overflow Flag. Set when not all the data read from the serial flash device could be pushed into the RX Buffer. The IP Command leading to this condition is continued until the number of bytes according to the QSPI_IPCR[IDATSZ] field has been read from the serial flash device. The content of the RX Buffer is not changed.
16 RBDF	RX Buffer Drain Flag. Will be set if the QuadSPI_SR[RXWE] status bit is asserted. Writing 1 into this bit triggers one of the following actions: • If the RX Buffer has up to QuadSPI_RBCT[WMRK] valid entries then the flag is cleared. • If the RX Buffer has more than QuadSPI_RBCT[WMRK] valid entries and the QuadSPI_RSER[RBDDE] bit is not set (flag driven mode) a RX Buffer POP event is triggered. The flag remains set if the RX Buffer contains more than QuadSPI_RBCT[WMRK] valid entries after the RX Buffer POP event is finished. The flag is cleared if the RX Buffer contains less than or equal to QuadSPI_RBCT[WMRK] valid entries after the RX Buffer POP event is finished. Refer to "Receive Buffer Drain Interrupt or DMA Request" section in Normal Mode Interrupt and DMA Requests, for details.
15 ABSEF	AHB Sequence Error Flag. Set when the execution of an AHB Command is started with a WRITE or WRITE_DDR Command in the sequence pointed to by the QSPI_BUFxCR register. (QSPI_BUFxCR implies any one of QSPI_BUF0CR/QSPI_BUF1CR/QSPI_BUF2CR/QSPI_BUF3CR.) Communication with the serial flash device is terminated before the execution of WRITE/WRITE_DDR command by the QuadSPI module. The AHB bus request which triggered this command is answered with an ERROR response.
14 AITEF	AHB Illegal transaction error flag. Set whenever there is no response generated from QSPI to AHB bus in case of illegal transaction and the watchdog timer expires.The timer value is taken as parameter.
13 AIBSEF	AHB Illegal Burst Size Error Flag. Set whenever the total burst size (size x beat) of an AHB transaction is greater than the prefetch data size. The prefecth data size is defined by QSPI_BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field in case ADATSZ = 0. Refer to HBURST Support for more details on HBURST feature.
12 ABOF	AHB Buffer Overflow Flag. Set when the size of the AHB access exceeds the size of the AHB buffer. This condition can occur only if the QSPI_BUFxCR[ADATSZ] field is programmed incorrectly. The AHB Command leading to this condition is continued until the number of entries according to the QSPI_BUFxCR[ADATSZ] field has been read from the serial flash device. The content of the AHB Buffer is not changed.
11 Reserved	This field is reserved.
10–8 Reserved	This field is reserved.
7 IPAEF	IP Command Trigger during AHB Access Error Flag. Set when the following condition occurs: • A write access occurs to the QSPI_IPCR[SEQID] field and the QSPI_SR[AHB_ACC] bit is set. Any command leading to the assertion of the IPAEF flag is ignored.

Table continues on the next page...

QuadSPIx_FR field descriptions (continued)

Field	Description
6 IPIEF	IP Command Trigger could not be executed Error Flag. Set when the QSPI_SR[IP_ACC] bit is set (i.e. an IP triggered command is currently executing) and any of the following conditions occurs: • Write access to the QSPI_IPCR register. Any command leading to the assertion of the IPIEF flag is ignored • Write access to the QSPI_SFAR register. • Write access to the QSPI_RBCT register.
5 Reserved	This field is reserved.
4 IPGEF	IP Command Trigger during AHB Grant Error Flag. Set when the following condition occurs: • A write access occurs to the QSPI_IPCR[SEQID] field and the QSPI_SR[AHBGNT] bit is set. Any command leading to the assertion of the IPGEF flag is ignored.
3–1 Reserved	This field is reserved.
0 TFF	IP Command Transaction Finished Flag. Set when the QuadSPI module has finished a running IP Command. If an error occurred the related error flags are valid, at the latest, in the same clock cycle when the TFF flag is asserted.

20.4.2.23 Interrupt and DMA Request Select and Enable Register (QuadSPIx_RSER)

The QuadSPI_RSER register provides enables and selectors for the interrupts in the QuadSPI module.

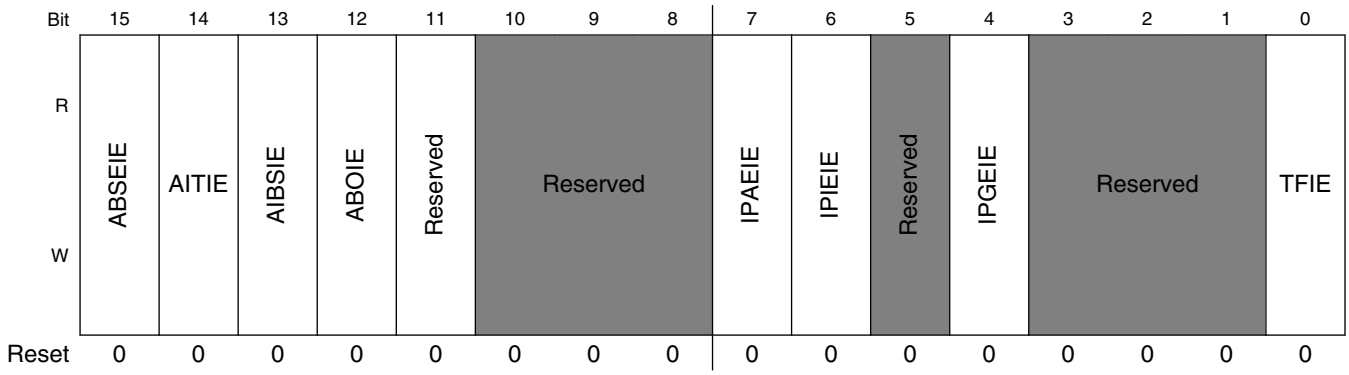
NOTE

Each flag of the QuadSPI_FR register enabled as source for an interrupt prevents the QuadSPI module from entering Stop Mode or Module Disable Mode when this flag is set.

Write: Anytime

Address: 410A_5000h base + 164h offset = 410A_5164h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DLPFIE	Reserved		Reserved	TBFIE	TBUIE	TBFDE	Reserved	ILLINIE	Reserved	RBDDE	Reserved			RBOIE	RBDIE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



QuadSPIx_RSER field descriptions

Field	Description
31 DLPFIE	Data Learning Pattern Failure Interrupt enable . Triggered by DLPFF flag in QSPI_FR register 0 No DLPFF interrupt will be generated 1 DLPFF interrupt will be generated
30–29 Reserved	This field is reserved.
28 Reserved	This field is reserved.
27 TBFIE	TX Buffer Fill Interrupt Enable 0 No TBFF interrupt will be generated 1 TBFF interrupt will be generated
26 TBUIE	TX Buffer Underrun Interrupt Enable 0 No TBUF interrupt will be generated 1 TBUF interrupt will be generated
25 TBFDE	TX Buffer Fill DMA Enable Enables generation of DMA requests for TX Buffer fill. When this is set, DMA requests are generated as long as the QuadSPI_SR[TXWA] status bit is set. **NOTE:** • Once set, writting a zero does not impact DMA transfers, even though the bit gets written a zero. • After QuadSPI_MCR[SWRSTHD] is used to do software reset for AHB Domain, this bit takes effect and needs to be written a 1 to re-enable DMA transfers. 0 No DMA request will be generated 1 DMA request will be generated
24 Reserved	This field is reserved.
23 ILLINIE	Illegal Instruction Error Interrupt Enable. Triggered by ILLINE flag in QSPI_FR 0 No ILLINE interrupt will be generated 1 ILLINE interrupt will be generated
22 Reserved	This field is reserved.
21 RBDDE	RX Buffer Drain DMA Enable: Enables generation of DMA requests for RX Buffer Drain. When this bit is set DMA requests are generated as long as the QSPI_SR[RXWE] status bit is set.

Table continues on the next page...

QuadSPIx_RSER field descriptions (continued)

Field	Description
	0 No DMA request will be generated 1 DMA request will be generated
20–18 Reserved	This field is reserved.
17 RBOIE	RX Buffer Overflow Interrupt Enable 0 No RBOF interrupt will be generated 1 RBOF interrupt will be generated
16 RBDIE	RX Buffer Drain Interrupt Enable: Enables generation of IRQ requests for RX Buffer Drain. When this bit is set the interrupt is asserted as long as the QuadSPI_SR[RBDF] flag is set. 0 No RBDF interrupt will be generated 1 RBDF Interrupt will be generated
15 ABSEIE	AHB Sequence Error Interrupt Enable: Triggered by ABSEF flags of QSPI_FR 0 No ABSEF interrupt will be generated 1 ABSEF interrupt will be generated
14 AITIE	AHB Illegal transaction interrupt enable. 0 No AITEF interrupt will be generated 1 AITEF interrupt will be generated
13 AIBSIE	AHB Illegal Burst Size Interrupt Enable 0 No AIBSEF interrupt will be generated 1 AIBSEF interrupt will be generated
12 ABOIE	AHB Buffer Overflow Interrupt Enable 0 No ABOF interrupt will be generated 1 ABOF interrupt will be generated
11 Reserved	This field is reserved.
10–8 Reserved	This field is reserved.
7 IPAEIE	IP Command Trigger during AHB Access Error Interrupt Enable 0 No IPAEF interrupt will be generated 1 IPAEF interrupt will be generated
6 IPIEIE	IP Command Trigger during IP Access Error Interrupt Enable 0 No IPIEF interrupt will be generated 1 IPIEF interrupt will be generated
5 Reserved	This field is reserved.
4 IPGEIE	IP Command Trigger during AHB Grant Error Interrupt Enable 0 No IPGEF interrupt will be generated 1 IPGEF interrupt will be generated
3–1 Reserved	This field is reserved. Reserved.

Table continues on the next page...

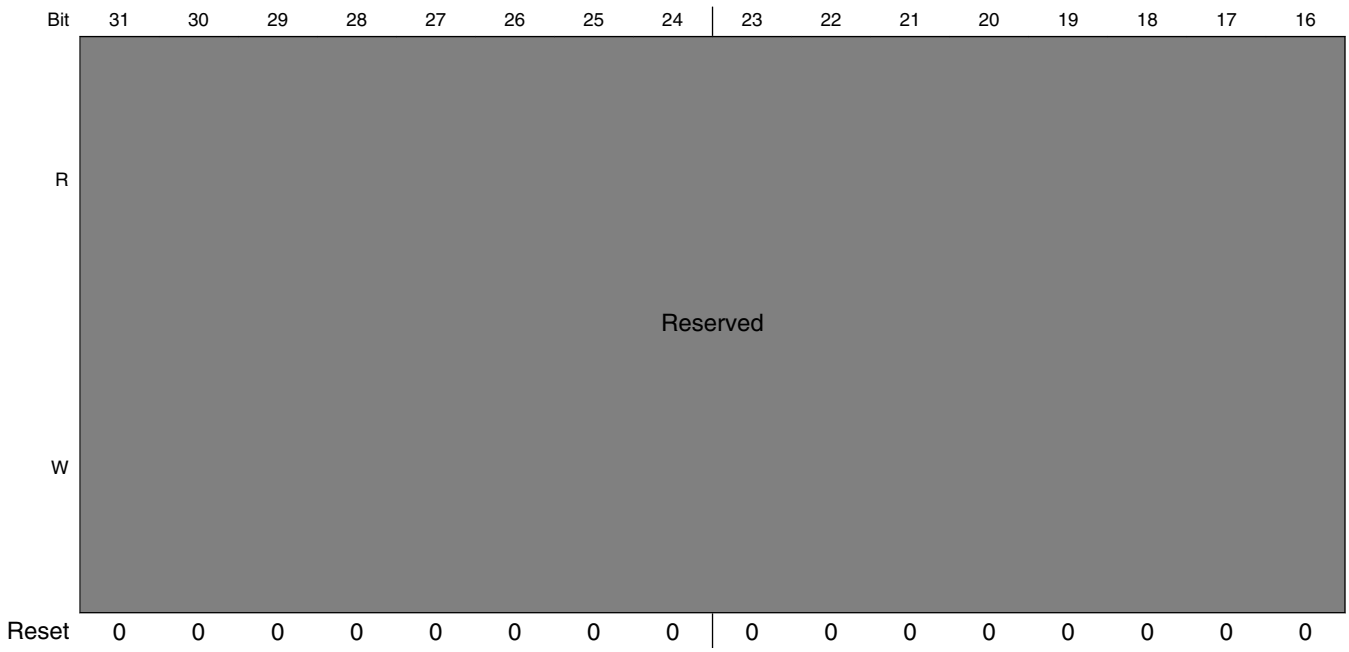
QuadSPIx_RSER field descriptions (continued)

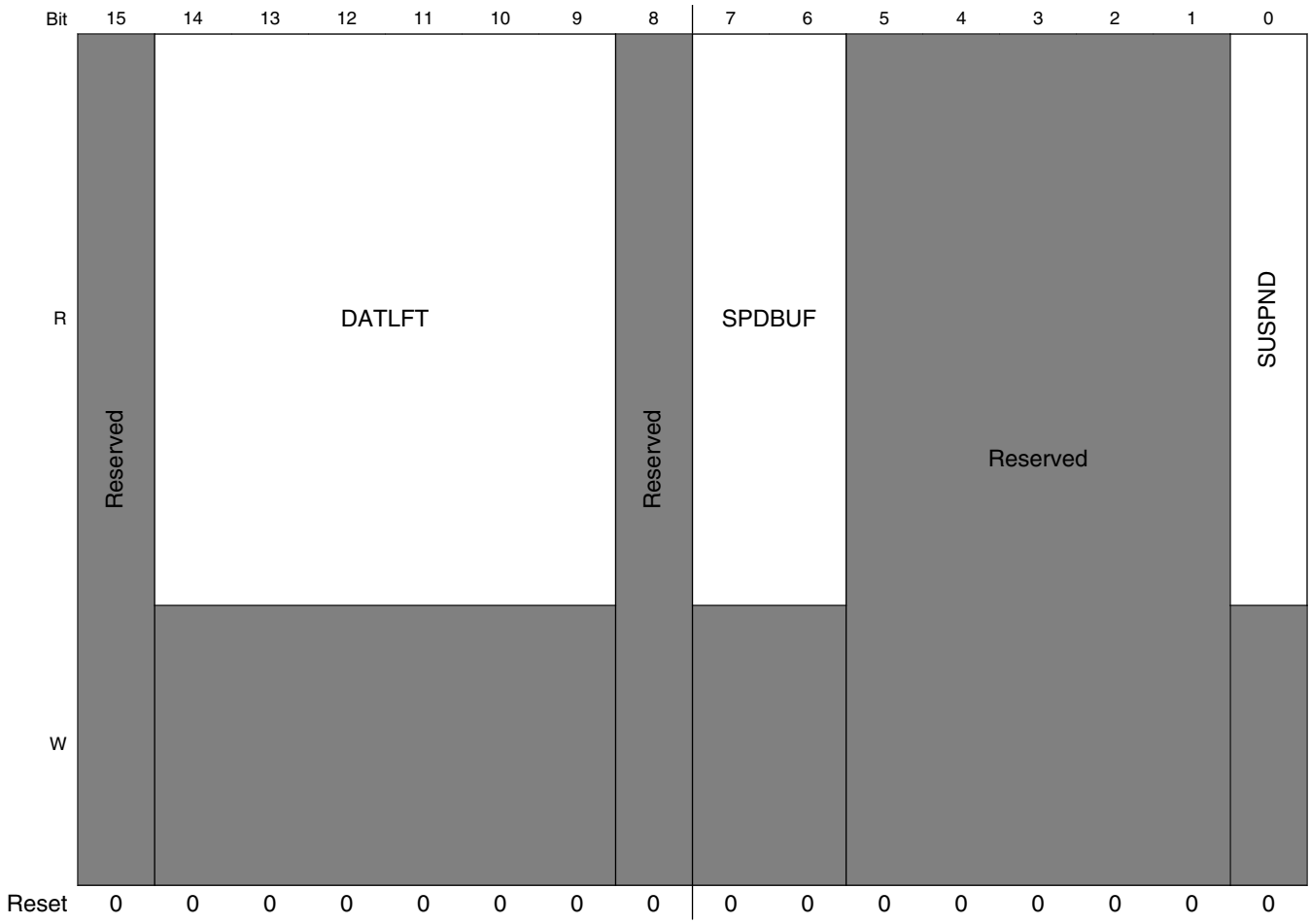
Field	Description
0 TFIE	Transaction Finished Interrupt Enable 0 No TFF interrupt will be generated 1 TFF interrupt will be generated

20.4.2.24 Sequence Suspend Status Register (QuadSPIx_SPNDST)

The sequence suspend status register provides information specific to any suspended sequence. An AHB sequence may be suspended when a high priority AHB master makes an access before the AHB sequence completes the data transfer requested.

Address: 410A_5000h base + 168h offset = 410A_5168h





QuadSPIx_SPNDST field descriptions

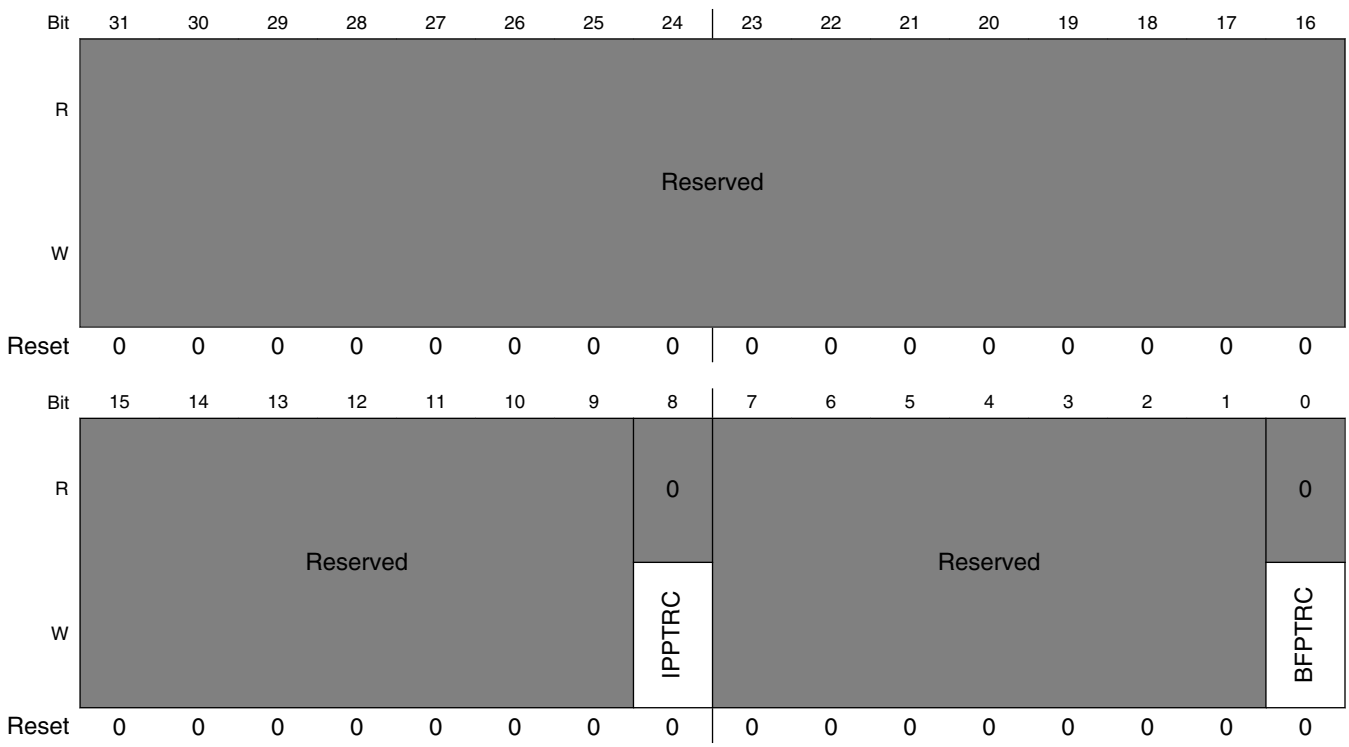
Field	Description
31–16 Reserved	This field is reserved.
15 Reserved	This field is reserved.
14–9 DATLFT	Data left: Provides information about the amount of data left to be read in the suspended sequence. Valid only when SUSPND is set to 1'b1. Value in terms of 64 bits or 8 bytes
8 Reserved	This field is reserved.
7–6 SPDBUF	Suspended Buffer: Provides the suspended buffer number. Valid only when SUSPND is set to 1'b1
5–1 Reserved	This field is reserved.
0 SUSPND	When set, it signifies that a sequence is in suspended state

20.4.2.25 Sequence Pointer Clear Register (QuadSPIx_SPTRCLR)

The sequence pointer clear register provides bits to reset the IP and Buffer sequence pointers. The sequence pointer contains the index of which instruction within the LUT entry is to be executed next. For example, if the LUT entry ends on a JMP_ON_CS value of 2, the index will be stored as 2.

The software should reset the sequence pointers whenever the sequence ID is changed by updating the SEQID field in QSPI_IPCR or QSPI_BFGENCR.

Address: 410A_5000h base + 16Ch offset = 410A_516Ch



QuadSPIx_SPTRCLR field descriptions

Field	Description
31–9 Reserved	This field is reserved.
8 IPPTRC	IP Pointer Clear: 1: Clears the sequence pointer for IP accesses as defined in QuadSPI_IPCR This is a self-clearing field. This bit should be programmed two times to clear the sequence pointers.
7–1 Reserved	This field is reserved. Reserved.
0 BFPTRC	Buffer Pointer Clear: 1: Clears the sequence pointer for AHB accesses as defined in QuadSPI_BFGENCR. This is a self-clearing field. This bit should be programmed two times to clear the sequence pointers.

20.4.2.26 Serial Flash A1 Top Address (QuadSPIx_SFA1AD)

The QSPI_SFA1AD register provides the address mapping for the serial flash A1.The difference between QSPI_SFA1AD[TPADA1] and QSPI_AMBA_BASE defines the size of the memory map for serial flash A1.

Write:

- *QSPI_SR[IP_ACC] = 0*
- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 180h offset = 410A_5180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TPADA1																						Reserved									
Reset	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0	0	0	0	0	0	0	0	0	0

* Notes:

- TPADA1 field: See the module configuration for the device specific reset values.

QuadSPIx_SFA1AD field descriptions

Field	Description
31–10 TPADA1	Top address for Serial Flash A1. In effect, TPADxx is the first location of the next memory.
Reserved	This field is reserved.

20.4.2.27 Serial Flash A2 Top Address (QuadSPIx_SFA2AD)

The QSPI_SFA2AD register provides the address mapping for the serial flash A2.The difference between QSPI_SFA2AD[TPADA2] and QSPI_SFA1AD[TPADA1] defines the size of the memory map for serial flash A2.

Write:

- *QSPI_SR[IP_ACC] = 0*
- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 184h offset = 410A_5184h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TPADA2																						Reserved									
Reset	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0	0	0	0	0	0	0	0	0	0

* Notes:

- TPADA2 field: See the module configuration for the device specific reset values.

QuadSPIx_SFA2AD field descriptions

Field	Description
31–10 TPADA2	Top address for Serial Flash A2. In effect, TPxxAD is the first location of the next memory.
Reserved	This field is reserved.

20.4.2.28 Data Learn Pattern Register (QuadSPIx_DLPR)

The QSPI_DLPR register contains the information of the data to be used for Data Learning.

Write:

- *QSPI_SR[IP_ACC] = 0*
- *QSPI_SR[AHB_ACC] = 0*

Address: 410A_5000h base + 190h offset = 410A_5190h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DLPV																															
Reset	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	0	1	1	0	1	0	0	0	1	0	0	0	0	1	1

QuadSPIx_DLPR field descriptions

Field	Description
DLPV	Data Learning Pattern Value: This value is used for data learning in DDR and DQS mode. If programmer wants to do data learn for more than 32 bit than the same value in the register is repeated. Say if 64 bit data learning is requested by any flash and the value of DLPR is aa55_3443 then the 64 bit value will be aa55_3443_aa55_3443. If 8 bit data learning was enabled by programming in seq_operand fields of DATA_LEARN instruction to 1 , the bits [7:0] are used as data_learning pattern. For details, refer to Data Learning

20.4.2.29 RX Buffer Data Register (QuadSPIx_RBDR)

The QuadSPI_RBDR registers provide access to the individual entries in the RX Buffer. Refer to Table 20-17 for the byte ordering scheme.

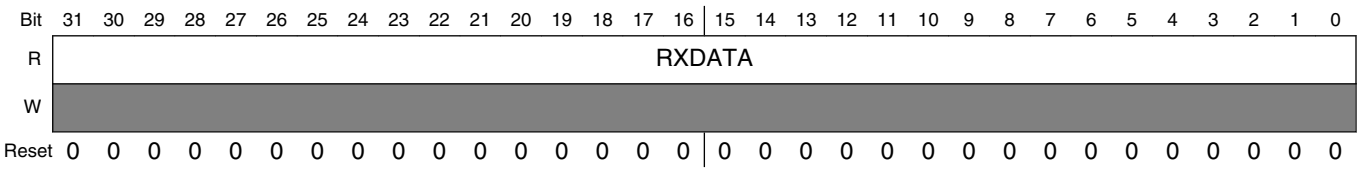
QuadSPI_RBDR0 corresponds to the actual position of the read pointer within the RX Buffer. The number of valid entries available depends from the number of RX Buffer entries implemented and from the number of valid buffer entries available in the RX Buffer.

Example 1, RX Buffer filled completely with 16 words: In this case the address range for valid read access extends from QuadSPI_RBDR0 to QuadSPI_RBDR15 .

Example 2, RX Buffer filled with 5 valid words: RX Buffer fill level QuadSPI_RBSR[RDBFL] is 5. In this case an access to QuadSPI_RBDR4 provides the last valid entry.

Any access beyond the range of valid RX Buffer entries provides undefined results.

Address: 410A_5000h base + 200h offset + (4d × i), where i=0d to 15d



QuadSPIx_RBDR*n* field descriptions

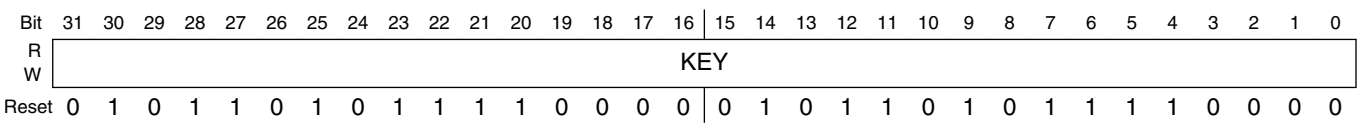
Field	Description
RXDATA	RX Data. The RXDATA field contains the data associated with the related RX Buffer entry. Data format and byte ordering is given in Byte Ordering of Serial Flash Read Data .

20.4.2.30 LUT Key Register (QuadSPIx_LUTKEY)

The LUT Key register contains the key to lock and unlock the Look-up-table. Refer to Look-up Table for details.

Write: Anytime

Address: 410A_5000h base + 300h offset = 410A_5300h



QuadSPIx_LUTKEY field descriptions

Field	Description
KEY	The key to lock or unlock the LUT. The KEY is 0x5AF05AF0. The read value is always 0x5AF05AF0

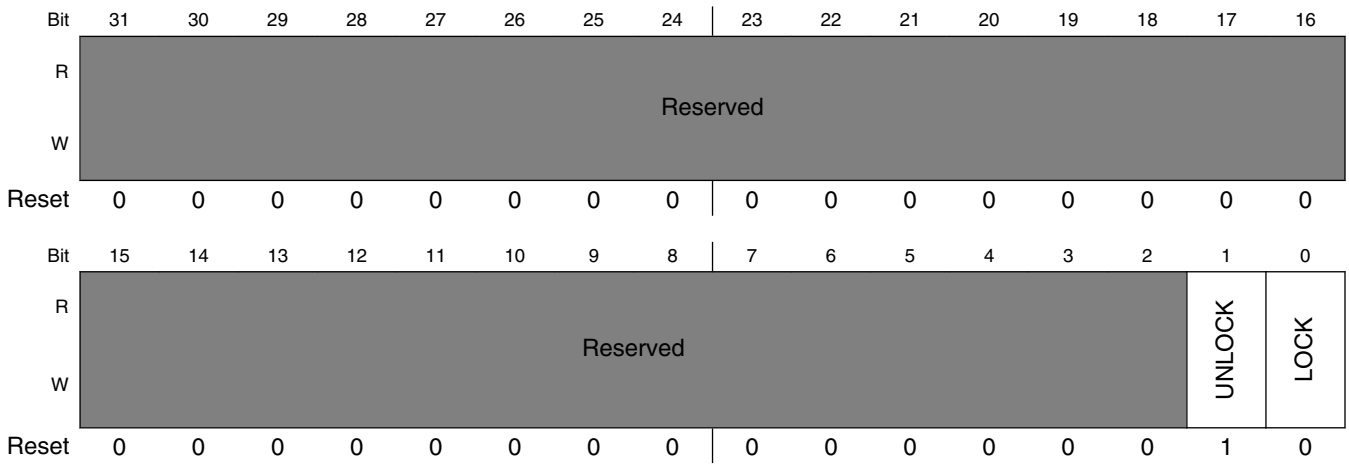
20.4.2.31 LUT Lock Configuration Register (QuadSPIx_LCKCR)

The LUT lock configuration register is used along with QSPI_LUTKEY register to lock or unlock the LUT. This register has to be written immediately after QSPI_LUTKEY register for the lock or unlock operation to be successful. Refer to Look-up Table for details. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

Write: Just after writing the LUT Key Register

(QSPI_LUTKEY)

Address: 410A_5000h base + 304h offset = 410A_5304h



QuadSPIx_LCKCR field descriptions

Field	Description
31–2 Reserved	This field is reserved.
1 UNLOCK	Unlocks the LUT when the following two conditions are met: 1. This register is written just after the LUT Key Register (QuadSPI_LUTKEY) 2. The LUT key register was written with 0x5AF05AF0 key
0 LOCK	Locks the LUT when the following condition is met: 1. This register is written just after the LUT Key Register (QuadSPI_LUTKEY) 2. The LUT key register was written with 0x5AF05AF0 key

20.4.2.32 Look-up Table register (QuadSPIx_LUT*n*)

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence pre-programmed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers are a look-up-table for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash transaction. There are a total of 64 LUT registers. These 64 registers are divided into groups of 4 registers that make a valid sequence. Therefore, QSPI_LUT[0], QSPI_LUT[4], QSPI_LUT[8] ..... QSPI_LUT[60] are the starting registers of a valid sequence. Each of these sets of 4 registers can have a maximum of 8 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT63. A maximum of 16 sequences can be defined at one time. Look-up Table describes the LUT registers in detail.

NOTE

The reset values for LUT0 and LUT1 are 0818_0403h and 2400_1C08h, respectively.

Write: Once the LUT is unlocked

Address: 410A_5000h base + 310h offset + (4d × i), where i=0d to 63d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*

* Notes:
- The reset values for LUT0 and LUT1 are 0818_0403h and 2400_1C08h respectively.

QuadSPIx_LUT*n* field descriptions

Field	Description
31–26 INSTR1	Instruction 1
25–24 PAD1	Pad information for INSTR1. 00 1 Pad 01 2 Pads 10 4 Pads 11 8 Pads

Table continues on the next page...

QuadSPIx_LUT*n* field descriptions (continued)

Field	Description
23–16 OPRND1	Operand for INSTR1.
15–10 INSTR0	Instruction 0
9–8 PAD0	Pad information for INSTR0. 00 1 Pad 01 2 Pads 10 4 Pads 11 8 Pads
OPRND0	Operand for INSTR0.

20.4.3 Serial Flash Address Assignment

The serial flash address assignment may be modified by writing into Serial Flash A1 Top Address (QuadSPI_SFA1AD) and Serial Flash A2 Top Address (QuadSPI_SFA2AD) for device A . The following table shows how different access modes are related to the address specified for the next SFM Command. Note that this address assignment is valid for both IP and AHB commands.

Table 20-11. Serial Flash Address Assignment

Parameter	Function	Access Mode
QSPI_AMBA_BASE ((31:10) - 22 bits)	QuadSPI AHB base address	
TOP_ADDR_MEMA1(TPADA1)	Top address for the external flash A1 (first device of the dual die flash A, or the first of the two independent flashes sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA1 and QSPI_AMBA_BASE will be routed to **Serial Flash A1**
TOP_ADDR_MEMA2(TPADA2)	Top address for the external flash A2 (second device of the dual die flash A, or the second of the two independent flashes sharing the IOFA).	Any access to the address space between TOP_ADDR_MEMA2 and TOP_ADDR_MEMA1 will be routed to **Serial Flash A2**

20.5 Flash memory mapped AMBA bus

QSPI_AMBA_BASE defines the address to be used as start address of the serial flash device as defined by the system memory map. Note that this may be a remapping of the physical address of the serial flash in the system. Refer to the system address map. Note that this may be a remapping of the physical address of the serial flash in the system. Refer to the system address map.

Table 20-12. QuadSPI AMBA Bus Memory Map

Address	Register Name
Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A	
QSPI_AMBA_BASE to (TOP_ADDR_MEMA2 - 0x01)	Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A Refer to Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A for details and to Table 20-17 and Table 20-18 for information about the byte ordering.
AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB15)	
QSPI_ARDB_BASE to… (32 * 4 Byte) QSPI_ARDB_BASE + 0x0000_01FF	AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB15) Refer to Table 20-17 for information about the byte ordering.

Note

Any read access to non-implemented addresses will provide undefined results.

In 'Individual Flash Modes', the 3/4 address bytes (as programmed in the instruction/operand in the sequence) available for the flash address is determined by SFADR [23:0] or SFADR [31:0]as given in the table above.

20.5.1 AHB Bus Access Considerations

It has to be noted that all logic in the QuadSPI module implementing the AHB Bus access is designed to read the content of an external serial flash device. Therefore, the following restrictions apply to the QuadSPI module with respect to accesses to the AHB bus.

- At present, the QuadSPI does not support AHB writes so any write access is answered with the ERROR condition according to the AMBA AHB Specification. No write occurs.

- Any AHB Command resulting in the assertion of the QSPI_FR[ABSEF] flag is answered with the ERROR condition according to the AMBA_AHB specification. The resulting AHB Command is ignored.
- AHB Bus access types fully supported are NONSEQ and BUSY.
- AHB access type SEQ is treated in the same way like NONSEQ. Refer to the AMBA AHB Specification for further details.

20.5.2 Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A

Starting with address QSPI_AMBA_BASE the content of the first external serial flash devices is mapped into the address space of the device containing the QuadSPI module. Serial flash address byte address 0x0 corresponds to bus address QSPI_AMBA_BASE with increasing order. Assuming that a dual-die flash is connected on the first set of external pads, the address space is divided into two parts, one for each device of the dual die package. Refer to the following table for the address mapping. The byte ordering for 32 bit access is given in Table 20-17 and for 64 bit read access the byte ordering is given in Table 20-18.

Table 20-13. Memory Mapped Individual Flash Mode - Flash A Address Scheme

Memory Mapped Address 32 Bit Access	Memory Mapped Address 64 Bit Access	Serial Flash Byte Address	Flash Device
QSPI_AMBA_BASE + 0x00	QSPI_AMBA_BASE + 0x00	0x00_0000 to 0x00_0003	A1
QSPI_AMBA_BASE + 0x04		0x00_0004 to 0x00_0007	
…	…	…	
TOP_ADDR_MEMA1 - 0x08	TOP_ADDR_MEMA1 - 0x08	(TOP_ADDR_MEMA1- 0x08) to (TOP_ADDR_MEMA1 - 0x04 -0x01)	
TOP_ADDR_MEMA1 - 0x04		(TOP_ADDR_MEMA1 - 0x04) to (TOP_ADDR_MEMA1 - 0x01)	
TOP_ADDR_MEMA1 + 0x00	TOP_ADDR_MEMA1 + 0x00_0000	0x00_0000 to 0x00_0003	A2
TOP_ADDR_MEMA1 + 0x04		0x00_0004 to 0x00_0007	
…..	…	…	
TOP_ADDR_MEMA2 - 0x08	TOP_ADDR_MEMA2 - 0x08	(TOP_ADDR_MEMA2 - 0x08) to (TOP_ADDR_MEMA2 - 0x04 - 0x01)	
TOP_ADDR_MEMA2 - 0x04		(TOP_ADDR_MEMA2 - 0x04) to (TOP_ADDR_MEMA2 - 0x01)	

The available address range depends from the size of the external serial flash device. Any access beyond the size of the external serial flash provides undefined results.

For details concerning the read process refer to Flash Read.

20.5.3 AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB15)

NOTE

See the System Memory map in this document for the base address of the QSPI AHB RX Data Buffer.

memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	AHB RX Data Buffer register (ARDB0)	32	R/W	0000_0000h	20.5.3.1/468
4	AHB RX Data Buffer register (ARDB1)	32	R/W	0000_0000h	20.5.3.1/468
8	AHB RX Data Buffer register (ARDB2)	32	R/W	0000_0000h	20.5.3.1/468
C	AHB RX Data Buffer register (ARDB3)	32	R/W	0000_0000h	20.5.3.1/468
10	AHB RX Data Buffer register (ARDB4)	32	R/W	0000_0000h	20.5.3.1/468
14	AHB RX Data Buffer register (ARDB5)	32	R/W	0000_0000h	20.5.3.1/468
18	AHB RX Data Buffer register (ARDB6)	32	R/W	0000_0000h	20.5.3.1/468
1C	AHB RX Data Buffer register (ARDB7)	32	R/W	0000_0000h	20.5.3.1/468
20	AHB RX Data Buffer register (ARDB8)	32	R/W	0000_0000h	20.5.3.1/468
24	AHB RX Data Buffer register (ARDB9)	32	R/W	0000_0000h	20.5.3.1/468
28	AHB RX Data Buffer register (ARDB10)	32	R/W	0000_0000h	20.5.3.1/468
2C	AHB RX Data Buffer register (ARDB11)	32	R/W	0000_0000h	20.5.3.1/468
30	AHB RX Data Buffer register (ARDB12)	32	R/W	0000_0000h	20.5.3.1/468
34	AHB RX Data Buffer register (ARDB13)	32	R/W	0000_0000h	20.5.3.1/468
38	AHB RX Data Buffer register (ARDB14)	32	R/W	0000_0000h	20.5.3.1/468
3C	AHB RX Data Buffer register (ARDB15)	32	R/W	0000_0000h	20.5.3.1/468

20.5.3.1 AHB RX Data Buffer register (ARDB)

The AHB RX Data Buffer register 0 to 15 can be used to read the buffer content of the RX Buffer from successive addresses. QSPI_ARDB0 corresponds to the RX Buffer register entry corresponding to the current value of the read pointer with increasing order.

The increment of the read pointer depends from the access scheme (DMA or flag-driven). Refer to "Data Transfer from the QuadSPI Module Internal Buffers" section in Flash Read section, RX Buffer, data read via register interface and AHB read, for the description of successive accesses to the RX Buffer content. Refer also to Byte Ordering of Serial Flash Read Data for the byte ordering scheme.

Valid address range accessible in the QSPI_ARDBn range depends from the number of RX Buffer entries implemented and from the number of valid buffer entries available in the RX Buffer.

- Example 1, RX Buffer filled completely with 16 words: In this case the address range for valid read access extends from QSPI_ARDB0 to QSPI_ARDB15.
- Example 2, RX Buffer filled with 5 valid words, RX Buffer fill level QSPI_RBSR[RDBFL] is 5. In this case an access to QSPI_ARDB4 provides the last valid entry.

Address: 0h base + 0h offset + (4d × i), where i=0d to 15d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ARXD																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ARDB*n* field descriptions

Field	Description
ARXD	ARDB provided RX Buffer Data. Byte order (endianness) is identical to the RX Buffer Data Registers.

20.6 Interrupt Signals

The interrupt request lines of the QuadSPI module are mapped to the internal flags according to the following table.

Table 20-14. Assignment of Interrupt Request Lines

IRQ/DMA line	QSPI_FR Flag	Interrupt Description
ipi_int_tfff	TBFF	TX Buffer Fill
ipi_int_tcf	TFF	Peripheral Command Transaction Finished
ipi_int_rfdf	RBDF	RX Buffer Drain
ipi_int_overrun		Buffer Overflow/Underrun Error Logical OR from:
	RBOF	RX Buffer Overrun
	TBUF	TX Buffer Underrun

Table continues on the next page...

Table 20-14. Assignment of Interrupt Request Lines (continued)

IRQ/DMA line	QSPI_FR Flag	Interrupt Description
	ABOF	AHB Buffer Overflow
ipi_int_cerr		Serial Flash Command Error Logical OR from:
	IPAEF	Peripheral access while AHB busy Error
	IPIEF	Peripheral Command could not be triggered Error
	IPGEF	Peripheral access while AHB Grant Error
ipi_int_ored	DLPFF, TBFF, TFF, ILLINE, RBDF, RBOF, TBUF, ABSEF, ABOF, IPAEF, IPIEF, IPGEF, AITEF, AIBSEF	Logical OR from all the QSPI_FR flags mentioned

20.7 Functional Description

This section provides the functional information of the QuadSPI module.

20.7.1 Serial Flash Access Schemes

Note

In the Individual Flash Mode, all supported commands are available.

20.7.2 Normal Mode

This mode is used to allow communication with an external serial flash device. Compared to the standard SPI protocol, this communication method uses up to 4 bidirectional data lines operating at high data rates. The communication to the external serial flash device consists of an instruction code and optional address, mode, dummy and data transfers. The flexible programmable core engine described below is immune to a wide variety of command/protocol differences in the serial flash devices provided by various flash vendors.

20.7.2.1 Programmable Sequence Engine

The core of the QuadSPI module is a programmable sequence engine that works on "instruction-operand" pairs. The core controller executes each programmed instruction sequentially. The complete list of instructions and the corresponding operands is given in the following table.

Table 20-15. Instruction set

Instruction	Instruction encoding	Pins	Operand	Action on Serial Flash(es)
CMD	6'd1	N=2'd{0,1,2,3} 2'd0 - One pad 2'd1 - Two pads 2'd2 - Four pads 2'd3- Eight pads	8 bit command value	Provide the serial flash with operand on the number of pads specified
ADDR	6'd2		Number of address bits to be sent (for example, 8'd24 => 24 address bits required)	Provide the serial flash with address cycles according to the operand on the number of pads specified. The actual address to be provided will be derived from the incoming address in case of AHB initiated transactions and the value of SFAR in case of IPS initiated transactions , if QSPI_SFACR[CAS] is set to 0, else the actual address will take CAS into consideration.
DUMMY	6'd3		Number of dummy clock cycles (should be <= 64 cycles)	Provide the serial flash with dummy cycles as per the operand. The PAD information defines the number of pads in input mode. (for example, one pad implies that pad 1 is not driven, rest all are driven)
MODE	6'd4		8 bit mode value	Provide the serial flash with 8 bit operand on the number of pads specified
MODE2	6'd5	N=2'd{0,1}	2 bit mode value	Provide the serial flash with 2 bit operand on the number of pads[1] specified
MODE4	6'd6	N=2'd{0,1,2}	4 bit mode value	Provide the serial flash with 4 bit operand on the number of pads[2] specified
READ	6'd7	N=2'd{0,1,2,3} 2'd0 - One pad 2'd1 - Two pads 2'd2 - Four pads 2'd3- Eight pads	Read data size in bytes (the user's application should ensure that data size is a multiple of 8 bytes)	Read data from flash on the number of pads specified. The data size may be overwritten by writing to the ADATSZ field of the QSPI_BUFxCR registers for AHB initiated transactions and IDATSZ field of IP Configuration Register (QuadSPI_IPCR) for IP initiated transactions.
WRITE	6'd8		Write data size in bytes	Write data on number of pads sepcified. The data size may be overwritten by writing to the IDATSZ field of IP Configuration Register (QuadSPI_IPCR) register
JMP_ON_CS	6'd9	NA	Instruction number	Every time the chip select (CS) is deasserted, jump to the instruction pointed to by the operand. This instruction allows the programmer to specify the behavior of the controller when a new read transaction is initiated following a CS deassertion.
ADDR_DDR	6'd10	N=2'd{0,1,2,3} 2'd0 - One pad 2'd1 - Two pads	Number of address bits to be sent (for example,	Provide the serial flash with address cycles according to the operand on the number of pads specified at each clock edge of serial flash clock. The actual address to be provided will be derived from the incoming address in case of AHB initiated

Table continues on the next page...

Table 20-15. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on Serial Flash(es)
		2'd2 - Four pads 2'd3- Eight pads	8'd24 => 24 address bits required)	transactions and the value of QSPI_SFAR in case of IPS initiated transactions , if QSPI_SFACR[CAS] is set to 0, else the actual address will take CAS into consideration.
MODE_DDR	6'd11		8 bit mode value	Provide the serial flash with 8 bit operand on the number of pads specified at each clock edge of serial flash.
MODE2_DDR	6'd12	N=2'd{0}	2 bit mode value	Provide the serial flash with 2 bit operand on the number of pads specified at each clock edge of serial flash[3]
MODE4_DDR	6'd13	N=2'd{0,1}	4 bit mode value	Provide the serial flash with 4 bit operand on the number of pads specified at each clock edge of serial flash[4].
READ_DDR	6'd14	N=2'd{0,1,2,3} 2'd0 - One pad 2'd1 - Two pads 2'd2 - Four pads 2'd3- Eight pads	Read data size in bytes (the user's application should ensure that data size is in multiple of 8 bytes)	Read data from flash on the number of pads specified at each clock edge of serial flash. The data size may be overwritten by writing to the ADATSZ field of the QSPI_BUFxCR registers for AHB initiated transactions and IDATSZ field of IP Configuration Register (QuadSPI_IPCR) for IP initiated transactions
WRITE_DDR	6'd15		Write data size in bytes	Write data on the number of pads specified at each clock edge of serial flash. The data size may be overwritten by writing to the IDATSZ field of IP Configuration Register (QuadSPI_IPCR) register
DATA_LEARN	6'd16		Number of data in bytes to be used for data learning. Say if operand is 1 than 8 bit data_learning is enabled.	Finds the correct sampling point in case of only DDR operations. When this instruction is encountered, the QSPI_SMPR[DDRSMP] values are ignored and the controller finds the correct sampling point on its own by data learning. But this feature of sampling point is valid for only DDR modes. If DQS mode is enabled, then data learn instruction just matches the incoming data from flash and if it does not matches the QSPI_DLPV then Flag in QSPI_FR[DLPFF] is set.
CMD_DDR	6'd17	N=2'd{0,1,2,3} 2'd0 - One pad 2'd1 - Two pads 2'd2 - Four pads 2'd3- Eight pads	8 bit command value	Provide the serial flash with the operand with number of pads specified at each clock edge of the serial flash
CADDR	6'd18		Number of Column address bits to be sent(8'd8 means 8 bits of column address is to be sent)	Provide the serial flash with column address cycles according to the operand on the number of pads specified. The actual address to be provided to flash will depend on value of QSPI_SFACR[CAS]. For example, if QSPI_SFACR[CAS] is 3, then the address to flash will be [2:0] of incoming address in case of AHB and the value of QSPI_SFAR in case of IP. This will be appended with zero if QSPI_SFACR[CAS] is less than number of pads for a Flash.
CADDR_DDR	6'd19		Number of Column address bits to be sent(8'd8 means 8 bits of column address is to be sent)	Provide the serial flash with column address cycles according to the operand on the number of pads specified at each clock edge of the serial flash. The actual address to be provided to flash will depend on value of QSPI_SFACR[CAS]. For example, if CAS is 3, then the address to flash will be [2:0] of incoming address in case of AHB and the value of QSPI_SFAR in case of IP. This will be appended with zero if CAS is less than number of pads for a Flash.

Table continues on the next page...

Table 20-15. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on Serial Flash(es)
STOP	8'd0	NA	NA	Stop execution; deassert CS

1. For a one pad instruction, MODE2 will take 2 serial flash clock cycles on the flash interface.
2. For a one pad instruction, MODE4 will take 4 serial flash clock cycles on the flash interface. For a 4 pad instruction, MODE4 will take 1 serial flash clock cycle on the flash interface.
3. For a one pad instruction, MODE2_DDR will take 1 serial flash clock cycle on the flash interface.
4. For a one pad instruction, MODE4_DDR will take 2 serial flash clock cycles on the flash interface. For a 4 pad instruction MODE4_DDR will take half a cycle on the serial flash interface.

The programmable sequence engine allows the user to configure the QuadSPI module according to the serial flash connected on board. The flexible structure is easily adaptable to new command/protocol changes from different vendors.

20.7.2.2 Flexible AHB buffers

In order to reduce the latency of the reads for AHB masters, the data read from the serial flash is buffered in flexible AHB buffers. There are four such flexible buffers. The size of each of these buffers is configurable with the minimum size being 0 Bytes and maximum size being the size of the complete buffer instantiated.The size of buffer 0 is defined as being from 0 to QSPI_BUF0IND. The Size of buffer 1 is from QSPI_BUF0IND to QSPI_BUF1IND, buffer2 is from QSPI_BUF1IND to QSPI_BUF2IND and buffer 3 is from QSPI_BUF2IND to the size of the complete buffer, which is given in the chip-specific QuadSPI information.

Each flexible AHB buffer is associated with the following

1. An AHB master. Optionally, buffer3 may be configured as an "all master" buffer by setting the QSPI_BUF3CR[ALLMST] bit. When buffer3 is configured in such a way, any access from a master not associated with any other buffer is routed to buffer3.
2. A datasize field representing the amount of data to be fetched from the flash on every "missed" access.

The master port number of every incoming request is checked and the data is returned/ fetched into the corresponding associated buffer. Every "missed" access to the buffer causes the controller to clear the buffer and fetch QSPI_BUFxCR[ADATSZ] amount of data from the serial flash.As such, there is no benefit in configuring a buffer size of greater than ADATSZ, as the locations greater than ADATSZ will never be used. For any AHB access, the sequence pointed to by the QSPI_BFGENCR[SEQID] field is used for the flash transaction initiated. The data is returned to the master as soon as the requested

amount is read from the serial flash. The controller however, continues to prefetch the rest of the data in anticipation of a next consecutive request. Figure 20-3 shows the flexible AHB buffers.

The QSPI_BFGENCR[SEQID] field points to an index of the LUT. Refer to Look-up Table for details.

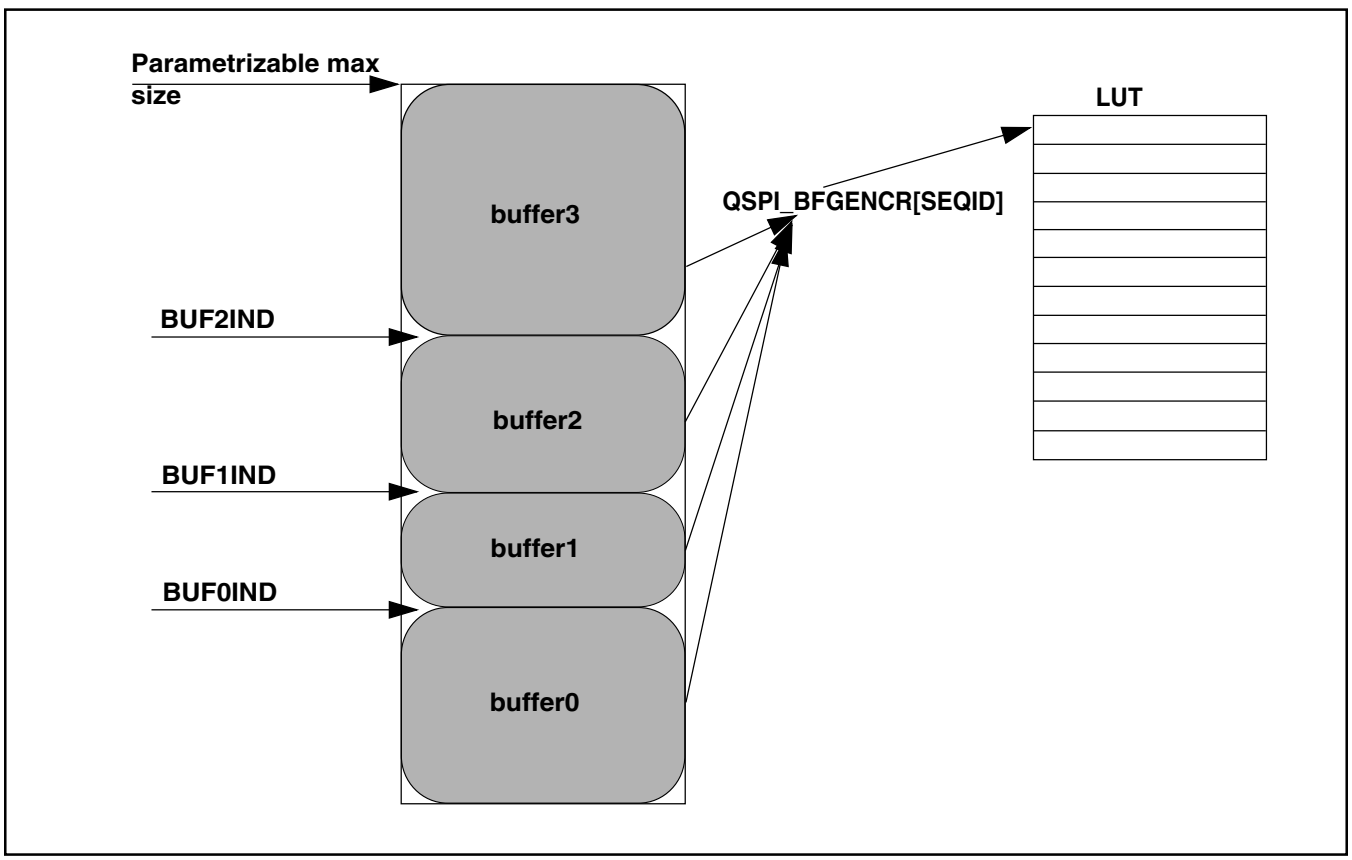


Figure 20-3. Flexible AHB Buffers

Buffer0 may optionally be configured to be associated with a high priority master by setting the QSPU_BUF0CR[HP_EN] bit. An access by a high priority master suspends any ongoing prefetch to any of the other buffers. The ongoing prefetch is suspended and the high priority master is serviced first. Once the high priority masters access completes, the suspended transaction is resumed (before any other AHB access is entertained). The status of the suspended buffer can be read from Sequence Suspend Status Register (QuadSPI_SPNDST).

20.7.2.3 Suspend-Abort Mechanism

Any low priority AHB access can be suspended by a high priority AHB master request. The ongoing transaction is suspended at 64 bit boundary. The suspended transaction is restarted after the high priority master is served and the high priority transaction including data prefetch is completed. While a transaction is in suspended state, it may be aborted if a transaction by the same suspended master is made to a location which is different from the location of the suspended transaction.

Any ongoing transaction is aborted if a request from the same master arrives for a location other than the location at which the transaction is going on. The abort can happen at any point of time.

20.7.2.4 HBURST Support

QuadSPI controller supports HBURST and HSIZE on the AHB interface. HBURST indicates if the transfer forms part of a burst. Four, eight and sixteen beat bursts are supported and the burst may be either incrementing or wrapping. HSIZE indicates the size of the transfer. 8, 16, 32 and 64 bit data size are supported. In case of WRAP accesses, QuadSPI generates aligned accesses to Serial Flash if there is no buffer hit for any incoming non-sequential AHB access. In case there is a buffer hit, the incoming address in the haddr line is latched as it is. If the total burst size is more than the data prefetch size an error response is generated and QSPI_FR[AIBSEF] is set. The data perfetch size can be defined by QSPI_BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field when ADATSZ is programmed as zero. A few examples are shown in the figure below:

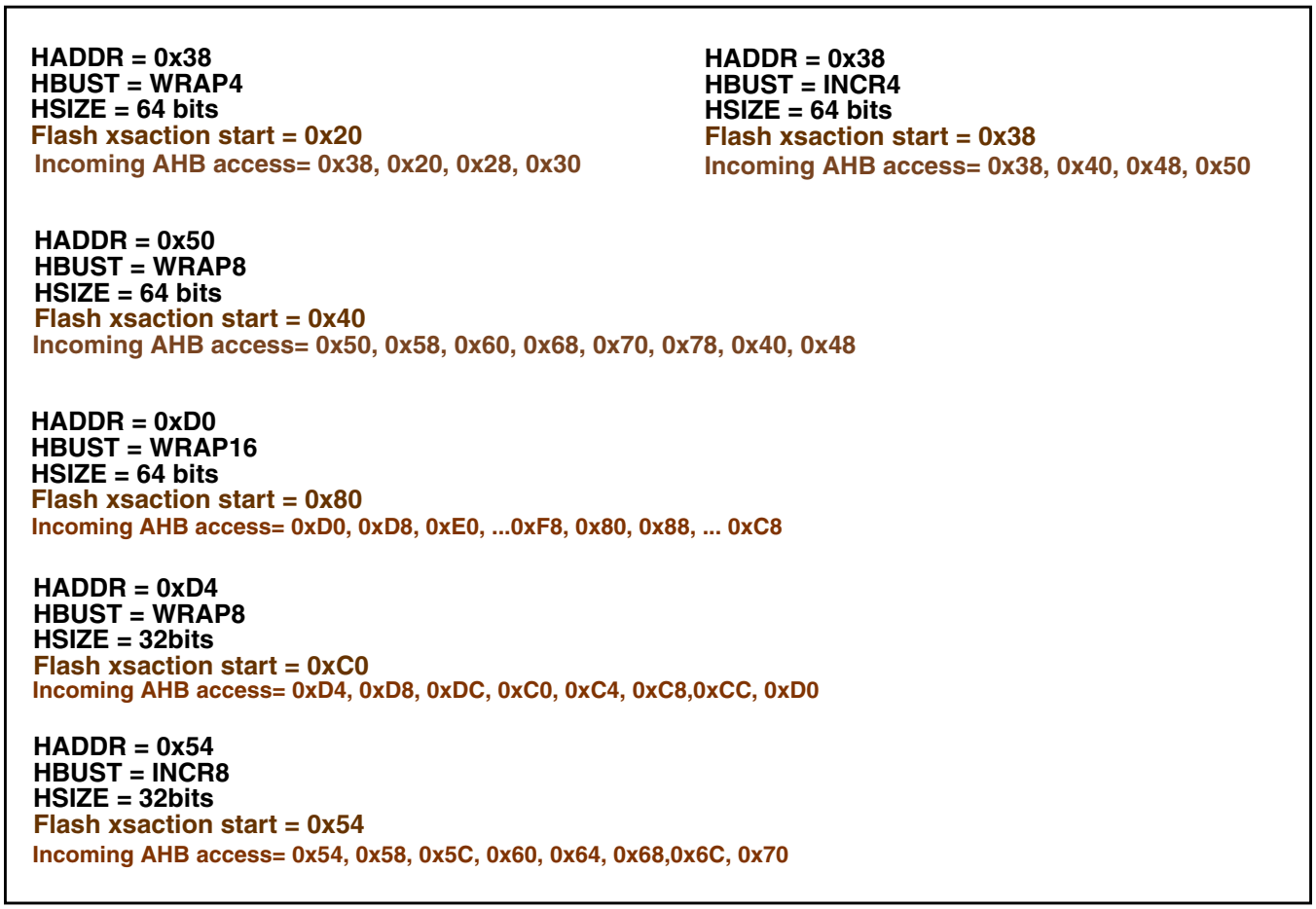


Figure 20-4. QuadSPI HBURST support

NOTE

The software must take care that the prefetch size should never be set less than the minimum data needed by any external interface to start processing.

NOTE

Whenever a core access QuadSPI memory with cache enabled, prefetch size must be configured equal or more than the cache line size, otherwise QSPI_FR[AIBSEF] error gets set.

20.7.2.5 Look-up Table

The Look-up-table or LUT consists of a number of pre-programmed sequences. Each sequence is basically a sequence of instruction-operand pairs which when executed sequentially generates a valid serial flash transaction. Each sequence can have a maximum of 8 instruction-operand pairs. The LUT can hold a maximum of 16 sequences. The figure below shows the basic structure of the sequence in the LUT.

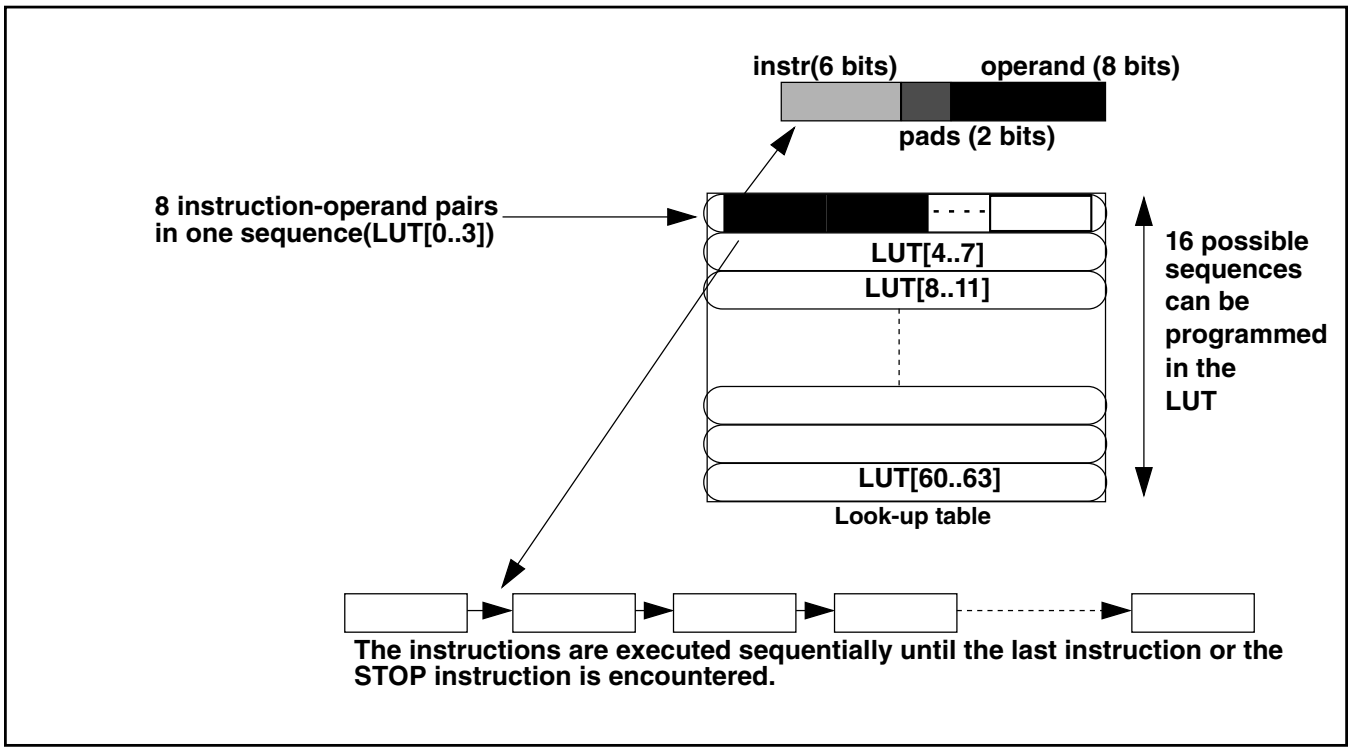


Figure 20-5. LUT and sequence structure

At reset, the index 0 of the look-up-table (LUT[0..3]) is programmed with a basic read sequence as given in Table 20-16. After reset the complete LUT may be reprogrammed according to the device connected on board.In order to protect its contents during a code runover the LUT may be locked, after which a write to the LUT will not be successful until it has been unlocked again.The key for locking or unlocking the LUT is **0x5AF05AF0**. The process for locking and un-locking the LUT is as follows:

Locking the LUT

1. Write the key **(0x5AF05AF0)** in to the LUT Key Register (QuadSPI_LUTKEY).
2. Write 0b01 to the LUT Lock Configuration Register (QuadSPI_LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write into this register locks the LUT.

Unlocking the LUT

1. Write the key **(0x5AF05AF0)** into the LUT Key Register (QuadSPI_LUTKEY)
2. Write 0b10 to the LUT Lock Configuration Register (QuadSPI_LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write into this register unlocks the LUT.

The lock status of the LUT can be read from QSPI_LCKCR[UNLOCK] and QSPI_LCKCR[LOCK] bit.

Some example sequences are defined in Example Sequences. The reset sequence at LUT index 0 is given in the following table.

Table 20-16. Reset sequence

Instruction	Pad	Operand	Comment
CMD	0x00	0x03	Read Data byte command on one pad
ADDR	0x00	0x18	24 Addr bits to be sent on one pad
READ	0x00	0x08	Read 64 bits
JMP_ON_CS	0x00	0x00	Jump to instruction 0 (CMD)

20.7.2.6 Issuing Serial Flash Memory (SFM) Commands

Each access to the external device follows the same sequence:

1. The user must pre-populate the LUT with the serial flash command sequences that are required for the flash device being used.
2. The QuadSPI module starts executing the instructions in the sequence one by one. The transaction starts and the status bit QSPI_SR[BUSY] is set.
3. Communication with the external serial flash device is started and the transaction is executed.
4. When the transaction is finished (all transmit- and receive operations with the external serial flash device are finished) the status bit QSPI_SR[BUSY] is reset. In case of an IP Command the QSPI_FR[TFF] flag is asserted.

Further details are given in below in Flash Programming and Flash Read.

You can trigger the processing of SFM commands in the QuadSPI module in one of the following ways:

- **Using IP commands**

  For IP Commands the required components need to be written into the following registers:

  - Write the serial flash address to be used by the instruction into QSPI_SFAR, refer to Serial Flash Address Register (QSPI_SFAR). For IP Commands not related to specific addresses, the base address of the related flash need to be

programmed.For example, for an instruction which does not require an address (i.e. write enable instruction) the SFAR should be programmed with the base address of the memory the command is to be sent to.

- Write the sequence ID and data size details in the IP Configuration Register (QSPI_IPCR).
- Note that the write into the QSPI_IPCR[SEQID] field must be the last step of the sequence. It is possible to combine all fields of the QSPI_IPCR into one single write. Refer to IP Configuration Register (QSPI_IPCR) for details.

Note that there are some conditions where no IP Command is executed after writing the QSPI_IPCR[SEQID] field and the write operation itself is ignored. They are described in Command Arbitration.

- **Using AHB commands**

  Any AHB memory mapped access is routed to one of the buffers depending on the master port number of the request. If the access is a "miss", a new serial flash transaction is started. The transaction is based on the sequence pointed to by the BFGENCR[SEQID] field as described in Flexible AHB buffers.

  An AHB access is termed memory mapped when the access is to the memory mapped serial flashes, as described in Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A.

  Again the possible error conditions are described in Command Arbitration .

20.7.2.7 Flash Programming

In all cases the memory sector to be written needs to be erased first. The programming sequence itself is then initiated in the following way:

1. Check QuadSPI_SR[BUSY] is de-asserted or '0'. Check that the TX buffer is empty. If it is required to discard the data present in the TX buffer (QSPI_SR[TXEDA]) bit is set, then the TX buffer must be cleared by writing '1' into the QSPI_MCR[CLR_TXF] bit.
2. Program the address related to the command in the QSPI_SFAR register. Program the QSPI_SFACR[CAS] if required to desired value else to 0. Program the QSPI_SFACR[WA] to 1 if the serial flash is a word addressable flash else to 0 in case serial flash is byte addressable.

3. Provide initial data for the program command into the circular buffer via register TX Buffer Data Register (QSPI_TBDR) . At least four word of data must be written into the TX Buffer up to a maximum of 16.
4. Program the QSPI_IPCR register to trigger the command. The QSPI_IPCR[SEQID] should point to an index of the LUT which has the flash program sequence pre-programmed.The IDATSZ field should be set to denote the size of the write.
5. Depending on the amount of data required, step 3 must be repeated until all the required data have been written into the QSPI_TBDR register.The QSPI_SR[TXFULL] can be used to check if the buffer is ready to receive more data. At any time, the QSPI_TBSR[TRCTR] field can be read to check how many words have been written actually into the TX Buffer.

Upon writing the QSPI_IPCR[SEQID] field (refer to step 4) the QuadSPI module will start to execute the programmed sequence. It is the responsibility of the software to ensure that a correct sequence is programmed into the LUT in accordance with the flash memory connected to the module. The data is fetched from the TX Buffer. It consists of **16** entries of 32-bits and is organized as a circular FIFO, whose read pointer is incremented by four after each fetch. When all data are transmitted, the QuadSPI module will return from 'busy' to 'idle'. However, this is not true for the external device since the internal programming is still ongoing. It is up to the user to monitor the relevant status information available from the serial flash device and to ensure that the programming is finished properly.

20.7.2.8 Flash Read

Host access to the data stored in the external serial flash device is done in two steps. First, the data must be read into the internal buffers and in the second step these internal buffers can be read by the host.

1. **Reading Serial Flash Data into the QuadSPI Module Internal Buffers**

   A read access to the external serial flash device can be triggered in two different ways:

   - **IP Command Read**: For **reading flash data into the RX Buffer** the user must provide the correct sequence ID in the QuadSPI_IPCR[SEQID] register. The sequence ID points to a sequence in the LUT. It is the responsibility of the software to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash device connected on board. The user should

program the Serial Flash Address Register (QSPI_SFAR) , QSPI_SFACR[CAS] and the IP Configuration Register (QSPI_IPCR) registers. All available read commands supported by the external serial flash are possible.

Optionally it is possible to clear the RX Buffer pointer prior to triggering the IP Command by writing a 1 into the QuadSPI_MCR[CLR_RXF] field.

From these inputs, the complete transaction is built when the QSPI_IPCR[SEQID] field is written. The transaction related to the read access starts and the requested number of bytes is fetched from the external serial flash device into the RX Buffer. Since the read access is triggered by an IP command, the IP_ACC status bit and the BUSY bit are both set (both are located in the Status Register (QSPI_SR)). A count of the number of entries currently in the Rx Buffer can be obtained from QSPI_RBSR[RDBFL].

The communication with the external serial flash is stopped when the specified number of bytes has been read (successful completion of the transaction).

- **AHB Command Read**: For **reading flash data into the AHB Buffer** the user must set up a read access by a master to the address range in the system memory map which the external serial flash devices are mapped to. The user should program the QSPI_SFACR[CAS], if required, to desired value else to 0. The user should also program the buffer registers corresponding to the AHB master initiating the request, this is depends on the configuration of the QSPI_RBCT[RXBRD]. The user should provide the correct sequence ID into the buffer generic configuration register (QSPI_BFGENCR). It is the responsibility of the software to ensure that a correct read sequence in programmed in the LUT in accordance with the serial flash device connected on board. Flash device selection and access mode are determined by the address accessed in the AHB address space associated to the QuadSPI module (refer to Memory Mapped Serial Flash Data - Individual Flash Mode on Flash A,

  On each AHB read access to the memory mapped area the valid data in the AHB Buffer is checked against the address requested in the actual read. When the AHB read request can't be served from the content of the AHB Buffer, the buffer is flushed and the sequence pointed to by the sequence ID is executed by the controller. The requested number of buffer entries defined in the QSPI_BUFxCR[ADATSZ] field is then fetched from the external serial flash device into the internal AHB Buffer. Since the read access is triggered via the AHB bus, the QSPI_SR[AHB_ACC] status bit is set driving in turn the QSPI_SR[BUSY] bit until the transaction is finished. The communication with the external serial flash is stopped when the specified number of entries has been filled.

2. Data Transfer from the QuadSPI Module Internal Buffers

The data read out from the external serial flash device by the QuadSPI module is stored in the internal buffers.The means of accessing the data from the buffer differs depending on which buffer the data has been loaded to. Refer to Block Diagram for details about the two available buffers, the RX Buffer and the AHB Buffer, in the QuadSPI module:

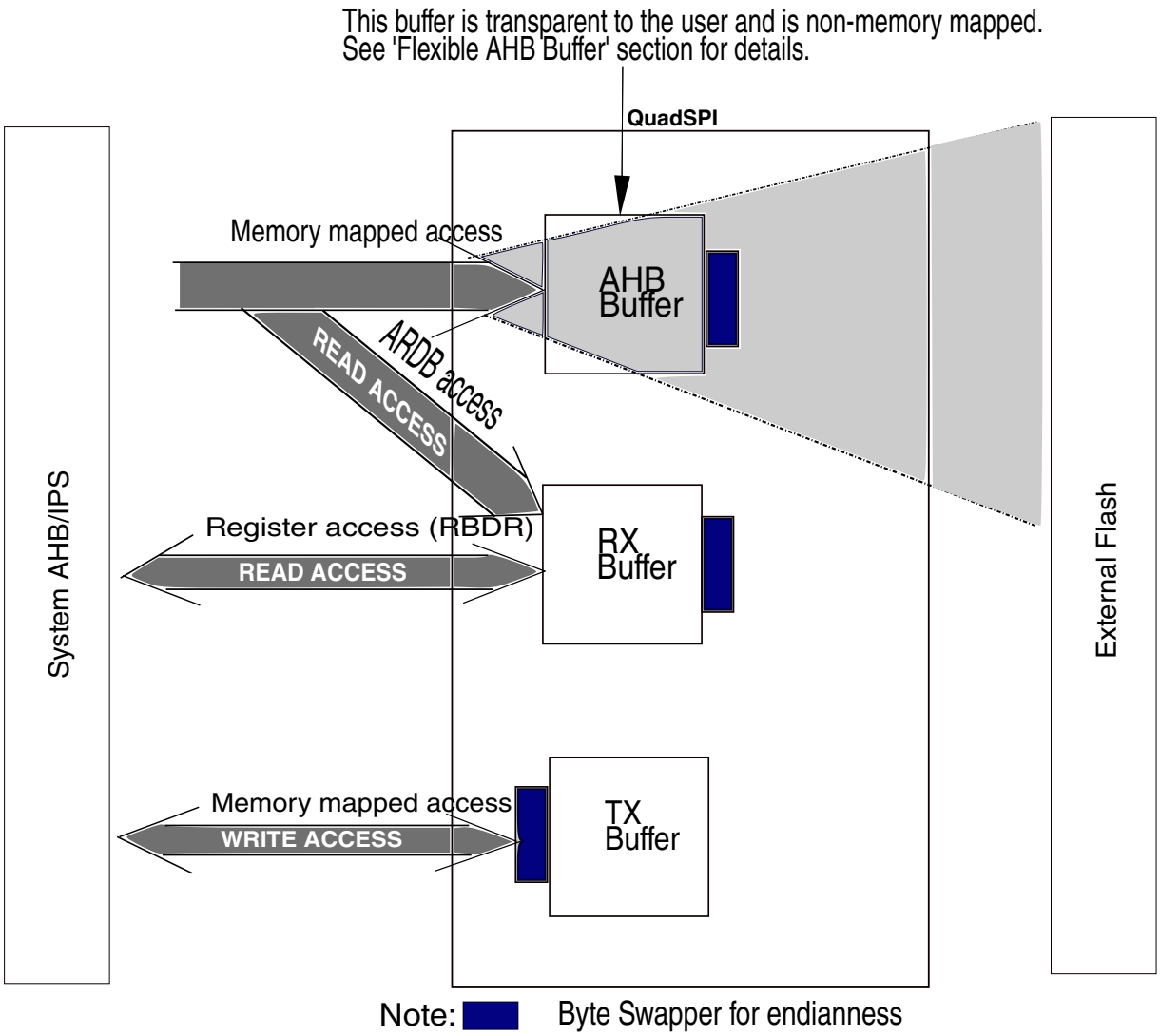


Figure 20-6. QuadSPI memory map

- The RX Buffer is implemented as FIFO of depth 16 entries of 4 bytes. Its content is accessible in two different address areas both referring to the identical data and the same physical memory.

  In the IPS address space in the area associated to QSPI_RBDR0 to QSPI_RBDR15

  In the AHB address space in the area associated to QSPI_ARDB0 to QSPI_ARDB15. Two successive entries are accessed with one single 64 bit AHB read operation.

RX Buffer operation can be summarized as follows: The QSPI_RBCT[WMRK] field determines at which fill level the RXWE bit is asserted and how many entries are removed from the RX Buffer on each Buffer POP operation. So the QSPI_SR[RXWE] bit indicates that the configured number of data entries is available in the RX Buffer and the QSPI_RBSR[RDBFL] field indicates how many valid entries are available in total. Note that the first entry (QSPI_RBDR0 or QSPI_ARDB0) always corresponds to the first valid entry in the RX Buffer. The software needs to manage the number of valid data bytes itself.

Further details can be found in RX Buffer Data Register (QuadSPI_RBDR) and in AHB RX Data Buffer (QSPI_ARDB0 to QSPI_ARDB15).

- **Flag-based Data Read of the RX Buffer** is done by polling the QSPI_SR[RXWE] bit. When it is asserted the valid entries can be read either via the IPS address space (QSPI_RBDRn) or the AHB address space (QSPI_ARDBn). A Buffer POP operation must be triggered by the application by writing a 1 into the QSPI_FR[RBDF] bit - this automatically updates the FIFO to point to the next entry as defined by RBCT[WMRK]. For example, if WMRK is set to 3, then the buffer will discard 16 bytes of data.
- **DMA controlled Data Read of the RX Buffer** is done by using the DMA module.The application must ensure that the DMA controller of the related device is programmed appropriately like it is described in DMA Usage.

  DMA controlled read out is triggered fully automatically by the assertion of the QSPI_SR[RXWE] bit. The related Buffer POP operation is also handled completely inside the QuadSPI module. Like in the case above, accessing the RX Buffer content either on QSPI_RBDRn or QSPI_ARDBn related addresses is equivalent.

- **AHB Buffer data read via memory mapped access:** This kind of access is done by reading one of the addresses assigned to the external serial flash device(s) within the range given in Table 20-12 table *under the condition that the data requested are already present in the AHB Buffer or it is currently being read from the serial flash device by the instruction in progress.* If this is not the case a memory mapped AHB command read is triggered as described above. If the requested data is already available in the AHB Buffer they are provided directly to the host.

  When AHB access are made to the flash memory mapped address, the data will be fetched and returned to the AHB interface. Till the data is being fetched the AHB interface would be stalled. As soon as the data from the requested address has been read by the QuadSPI module the AHB read access is served. So it is

possible to run sequential reads from the AHB buffer at arbitrary speed without the need to monitor any information about the availability of the data. Nevertheless this access scheme stalls the AHB bus for the time required to read the data from the serial flash device. A better way is (when it is known the access is sequential) to have a prefetch enabled (by programming the ADATSZ field) such that before the next sequential AHB access come, the data is already fetched into the buffer.

As long as the host restricts its accesses to the data already in the buffer and the data currently fetched from the serial flash, it is possible to run the host read from the AHB Buffer in parallel to the serial flash read into the AHB Buffer.

20.7.2.9 Byte Ordering of Serial Flash Read Data

In this paragraph the byte ordering of the serial flash data is given. The basic scheme is that the **first** byte read out of the serial flash device - which is addressed by the QSPI_SFAR[SFADR] field - corresponds to bit position QSPI_RBDR0[31:24] register for IP Command read. In contrast to that for AHB Command read the bytes are always positioned according to the byte ordering of the AHB bus.

- **Byte Ordering in Individual Flash Mode**

  The following table gives the byte ordering scheme of how the byte oriented data space of the serial flash device is mapped into one single 32 bit entry of the RX Buffer or the AHB Buffer. The table is valid within the following context:

  - Flash A in Individual Flash Mode
  - All AHB data read commands with access size of 32 bit

Table 20-17. Byte Ordering in Individual Flash Mode

Serial Flash Byte Numbering	3	2	1	0
Buffer Entry Bit Position [31:0] (32 Bit data width)	[31:24]	[23:16]	[15:8]	[7:0]

Note

For IP Commands the read size can be given in number of bytes. If this number is not a multiple of 4, then the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

For AHB Commands, reads, starting from an address not aligned to 32 bit boundaries, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- **Buffer Entry Ordering for 64 Bit Read Access**

  For read access via the AHB interface 64 bit access is possible. Each 64 bit access reads 2 32 bit entries simultaneously. The ordering of these 32 bit entries within the 64 bit word is given in the following table.

Table 20-18. 64 Bit Read Access Buffer Entry Ordering

AHB Read Data Bit Position [63:0]	[63:32]	[31:0]
Buffer Entry #	Odd (1, 3, 5, ...)	Even (0, 2, 4, ...)

20.7.2.10 Normal Mode Interrupt and DMA Requests

The QuadSPI module has different flags that can only generate interrupt requests and one flag that can generate interrupt as well as DMA requests. The following table lists the eight conditions. Note that the flags mentioned in the table are related to the Flag Register (QSPI_FR).

Table 20-19. Interrupt and DMA Request Conditions

Condition	Flag(QSPI_FR)	DMA
Data Learn patttern Failure	DLPFF	-
TX Buffer Fill	TBFF	-
TX Buffer Underrun	TBUF	-
Illegal Instruction Error	ILLINE	-
RX Buffer Drain	RBDF	X
RX Buffer Overflow	RBOF	-
AHB Buffer Overflow	ABOF	-
AHB Sequence Error	ABSEF	-
AHB Illegal Transaction Error	AITEF	-

Table continues on the next page...

Table 20-19. Interrupt and DMA Request Conditions (continued)

Condition	Flag(QSPI_FR)	DMA
AHB Illegal Burst Size Error	AIBSEF	-
IP Command Trigger during AHB Access Error	IPAEF	-
IP Command Trigger could not be executed Error	IPIEF	-
IP Access during AHB Grant Error	IPGEF	-
IP Command related Transaction Finished	TFF	-

Each condition has a flag bit in the Flag Register (QSPI_FR) and a Request Enable bit in the DMA Request Select and Enable Register (QSPI_RSER) . The RX Buffer Drain Flag (RBDF) has separate enable bits for generating IRQ and DMA requests. Note that not all flags have an individual IRQ line. Check the devices Interrupt Vector Table for more details.

- Transmit Buffer Fill Interrupt Request:

  The Transmit Buffer Fill IRQ indicates that the TX Buffer can accept new data. It is asserted if the QSPI_FR[TBFF] flag is asserted and if the corresponding enable bit (QSIP_RSER[TBFIE]) is set. Refer to TX Buffer Operation, for details about the assertion of the QSPI_FR[TBFF] flag.

  Aside from IRQ it is possible to handle TX Buffer fill by DMA. If the QSPI_RSER[TBFDE] bit is set, a DMA request will be triggered when the number of available space in the TX Buffer is more than the QSPI_TBCT[WMRK] valid entries and QSPI_SR[TXWA] is set. The application must set the environment appropriately (eg. the DMA controller) for the DMA transfer.

- Receive Buffer Drain Interrupt or DMA Request:

  The Receive Buffer Drain IRQ derived from the QSPI_FR[RBDF] flag indicates that the RX Buffer of the QuadSPI module has data available from the serial flash device to be read by the host. It remains set as long as the QSPI_RBSR[RXWE] bit is set. The QSPI_RSER[RBDIE] bit enables the related IRQ.

  Aside from the IRQ it is possible to handle RX Buffer drain by DMA. If the QSPI_RSER[RBDDE] bit is set, a DMA request will be triggered when the RX Buffer contains more than QSPI_RBCT[WMRK] valid entries.The application must set the environment appropriately (for example, the DMA controller) for the DMA transfers.

- Buffer Overflow/Underrun Interrupt Request:

The Buffer Overflow/Underrun IRQ is a combination of the following flags (all located in the QSPI_FR register with the related enable bits in the QSPI_RSER register):

- TBUF - TX Buffer Underrun, enabled by TBUIE
- RBOF - RX Buffer Overflow, enabled by RBOIE
- ABOF - AHB Buffer Overflow, enabled by ABOIE

  The Transmit Buffer Underrun indicates that an underrun condition in the TX Buffer has occurred. It is generated when a write instruction is triggered whilst the Tx Buffer is empty and the QSPI_RSER[TBUIE] bit is set.

  The Receive Buffer Overflow indicates that an overflow condition in the RX Buffer has occurred. It is generated when the RX Buffer is full, an additional read transfer attempts to write into the RX Buffer and the QSPI_RSER[RBOIE] bit is set.

  The AHB Buffer Overflow indicates that an overflow condition in the AHB Buffer has occurred. It is generated when the AHB Buffer is full, an additional read transfer attempts to write into the AHB Buffer and the QSPI_RSER[ABOIE] bit is set.

  The data from the transfers that generated the individual overflow conditions is ignored.

- Serial Flash Command Error Interrupt Request

  If the IPAEF, IPIEF, IPGEF flags in the QSPI_FR are set, and the related interrupt enable bits in the QSPI_RSER are also set, then an interrupt is requested.

- Transaction Finished Interrupt Request

  The IP Command Transaction Finished IRQ indicates the completion of the current IP Command.It is triggered by the QSPI_FR[TFF] flag and is masked by the QSPI_RSER[TFIE] bit.

20.7.2.11 TX Buffer Operation

The TX Buffer provides the data used for page programming. For proper operation it is required to provide at least four entry in the TX Buffer prior to starting the execution of the page programming command. The application must ensure that the required number of data bytes is written into the TX Buffer fast enough as long as the command is executed without a TX Buffer overflow or underrun.

The QuadSPI module sets the QSPI_FR[TBFF] flag so long as the TX Buffer is not full and can accept more data.

When the QuadSPI module tries to pull data out of an empty TX Buffer the TX Buffer underrun is signaled by the QSPI_FR[TBUF] flag.The TX buffer underrun flag is also asserted when TX buffer contains less than 128 bits of data and QuadSPI module tries to pull out data from it. The current IP Command leading to the underrun condition is continued until the specified number of bytes has been sent to the serial flash device, in the underrun condition when QuadSPI module tries to pull out data of empty TX buffer, the data transferred is all F's i.e. once the underrun flag is set under this condition, it will return F's until the required number of bytes are not sent. This has been done to ensure that the software need not to erase whole sector after underrun, just reprogramming from failure point will serve the purpose. When this Sequence Command is finished, the QSPI_FR[TBFF] flag is asserted indicating that the Tx Buffer is ready to be written again.

The TX Buffer overflow isn't signaled explicitly, but the TX Buffer fill level can be monitored by the QSPI_TBSR[TRBFL] field.

Refer to TX Buffer Status Register (QuadSPI_TBSR) and Flag Register (QuadSPI_FR) for details about the TX Buffer related registers.

20.7.2.12 Address scheme

Earlier serial flash memories supported only 24-bit address space hence restricting the maximum memory size of the serial flash as 16 MB. The new memory specification supports two types of 32-bit addressing mode in addition to legacy 24-bit address mode. It also supports segregation of address programmed, into Row address and Column address of the Flash, as per requirement.

- **Extended Address Mode**

  In this mode, the legacy 24-bit commands are converted to accept 32-bit address commands. The flash memory needs to be configured for 32-bit address mode. Also, while programming the LUT sequence in QuadSPI for 32-bit mode, the ADDR and ADDR_DDR command should be programmed with 8’d32 as the operand value with QSPI_SFACR[CAS] programmed to 0. If a flash needs some bits of the address as its column address, then it must always considered that the total bits required by the Flash should not exceed 32, as maximum address supported by QuadSPI is 32 bits. Each of the memory vendors have a different way of enabling this mode (Refer to the memory specification from memory vendors). For example, the command B7h sent to Macronix flash will enable it for 32-bit address mode.

- **Extended Address register**

  In this mode, the upper 8-bit of the 32-bit address is provided by the Extended address register in the memory itself. The memory provides a specific register which is updated according to the address to be accessed. This effectively converts the legacy 24-bit address command into 32-bit address commands. The memories greater in size than 16 MB, consists of banks of 16 MB. The 8-bit written in the extended address register effectively enables a bank. For example in Spansion memory, when the extended address register is updated with a value of 0x01 with the help of the command 17h, it will open Bank1 of the memory. The consequent 24-bit address commands will lead to Bank1. The extended address register needs to be update with the respective value for access to other banks. This effectively converts the legacy 24-bit address command into 32-bit address commands.

- **Separation of address into row and column address**

  This mode has been introduced for flashes which needs addresses segregated into Row and Column. The value in QSPI_SFACR[CAS] defines the width of the column address required by a flash. The actual address to be provided will be derived from the incoming address in case of AHB initiated transactions and the value of SFAR in case of IPS initiated transactions , if QSPI_SFACR[CAS] is set to 0, else the actual address will take CAS into consideration. If QSPI_SFACR[CAS] is 3 then bits 26-3 of the address programmed are sent to flash as it page address in case flash is operating in 24bit mode and bits 2-0 are sent as its column address. If a flash requirement for column address is less than the number of pads in which address has to be sent than the remaining bits are appended with 0 by QuadSPI. The user must program the operand value in CADDR and CADDR_DDR command accordingly. It must be ensured that the total number of address bits request by flash as its page and column address must not be more than 32 bits.

- **Word addressable mode for Flash**

  This mode has been introduced for flashes which has word addressable memory i.e. each address of the flash contains one word (two bytes) of data. The QSPI_SFACR [WA] is set to 1 to enter this mode. QuadSPI internally divides the incoming address in the AHB bus or the address in the QSPI_SFAR to map it to a valid flash location. For example, if the incoming address is 0x2004, the controller re-maps this address to access the flash location 0x1002. If not in this mode, the incoming address 0x2004 will be mapped to flash location 0x2004.

20.8 Initialization/Application Information

This section provides the initialization and application information of the QuadSPI module.

20.8.1 Power Up and Reset

Note that the serial flash devices connected to the QuadSPI module may require special voltage characteristics of their inputs during power up or reset. It is the responsibility of the application to ensure this.

20.8.2 Available Status/Flag Information

This paragraph gives an overview of the different status and flag information available and their interdependencies for different use cases. Related registers are QSPI_SR and QSPI_FR. Refer to the related descriptions how to set up the QuadSPI module appropriately.

20.8.2.1 IP Commands

Refer to IP Configuration Register (QuadSPI_IPCR) for additional details not explicitly covered in this paragraph.

- **IP Commands - Normal Operation**

  Writing the QSPI_IPCR[SEQID] field triggers the execution of a new IP Command. Given that this is a legal command the QSPI_SR[IPACC] and the QSPI_SR[BUSY] bits are asserted simultaneously, immediately after the execution is started.

  When the instruction on the serial flash device has been finished these bits are de-asserted and the QSPI_FR[TFF] flag is set.

- **IP Commands - Error Situations**

  Refer to Table 20-20 below.

20.8.2.2 AHB Commands

Refer to Section 1, Reading Serial Flash Data into the QuadSPI Module, in Flash Read for additional details not explicitly covered in this paragraph.

- **AHB Commands - Normal Operation**

  Memory mapped read access to a serial flash address not contained in the AHB Buffer, triggers the execution of an AHB Command. Given that this is a legal command the QSPI_SR[AHBACC] and the QSPI_SR[BUSY] bits are asserted simultaneously immediately after the execution is started. When the instruction on the serial flash device has been finished these bits are de-asserted.

- **IP Commands - Error Situations**

  Refer to Table 20-20 below.

20.8.2.3 Overview of Error Flags

The following table gives an overview of the different error flags in the QSPI_FR register and additional error-related details.

Table 20-20. Overview of QSPI_FR Error Flags

Error Category	Error Flag in QSPI_FR	Command Execution on Serial Flash Device TFF Behavior (in case of IP commands only)	Description
AHB Error Flag	ABSEF	Flash transaction is aborted	AHB sequence contains • WRITE instruction • WRITE_DDR instruction
AHB Error Flag	ABOF	Flash transaction continues until it finishes	Set when the module tried to push data into the AHB buffer that exceeded the size of the AHB buffer. Only occurs due to wrong programming of the QSPI_BUFxCR[ADATSZ].
AHB Error Flag	AIBSEF	Flash transaction is aborted	Total burst size of AHB transaction is greater than prefetch data size
AHB Error Flag	AITEF	Flash transaction is aborted	No response generated from QSPI to AHB bus in case of illegal transaction and the watchdog timer expires
Miscellaneous Error Flag	DLPFF	Flash transaction continues until it finishes	Set when DATA_LEARN instruction was encountered in a sequence but no sampling point was found for the data learning pattern.

Table continues on the next page...

Table 20-20. Overview of QSPI_FR Error Flags (continued)

Error Category	Error Flag in QSPI_FR	Command Execution on Serial Flash Device TFF Behavior (in case of IP commands only)	Description
Miscellaneous Error Flag	ILLINE	Flash transaction aborted	Illegal instruction Error Flag – Set when an illegal instruction is encountered by the controller in any of the sequences.
Command Arbitration Error	IPIEF	TFF not asserted in conjunction with that command	IP Command Error - caused when IP access is currently in progress (IP_ACC set) and • write attempt to QSPI_IPCR register. • write attempt to QSPI_SFAR register. • write attempt to QSPI_RBCT register.
Command Arbitration Error	IPAEF		• AHB Command already running, another IP Command could not be executed. • AHB Command already running, write attempt to QSPI_IPCR[SEQID] field.
Command Arbitration Error	IPGEF		• Exclusive access to the serial flash granted for AHB Commands, write attempt to QSPI_IPCR[SEQID] field.
Buffer Related Error	RBOF	TFF is asserted on completion	• RX Buffer Overrun
Buffer Related Error	TBUF		• TX Buffer Underrun

Note that only the buffer related errors are related to a transaction on the external serial flash. All the other errors do not trigger an actual transaction.

20.8.2.4 IP Bus and AHB Access Command Collisions

There are two flags related to this topic, the QSPI_FR[IPAEF] and QSPI_FR[IPIEF]. Refer to sub-section "Reading Serial Flash Data into the QuadSPI Module" of Flash Read section, for a description of the flags and Command Arbitration , for details about possible command collisions.

20.8.3 Exclusive Access to Serial Flash for AHB Commands

It is possible that several masters need to access the serial flash device connected to the QuadSPI module separately, one master by triggering IP Commands and reading the RX Buffer (via RBDR*n* register) and the other masters by triggering AHB Commands (via ARDB*n* Registers). These two set of buffer (RBDR and ARDB Buffer) points to the same physical buffer.Refer to Figure 20-6 To avoid command collisions resulting in excessive latencies the QuadSPI module implements a request-handshake mechanism between the master triggering AHB Commands and the QuadSPI module allowing this specific master to request exclusive access to the serial flash device for AHB Commands. If this exclusive access is granted the execution of IP Commands is blocked. This resolves command collisions and excessive times where the AHB interface may be blocked.

If this capability is used in the device there is additional status and flag information available related to this mechanism. The QSPI_SR[AHBGNT] bit reflects the module-internal state that the exclusive access mentioned above is granted, any attempt to trigger an IP Command is rejected and results in the assertion of the QSPI_FR[IPGEF] flag. Refer to the descriptions of the related bit and flag for details.

It is within the responsibility of the application to set up the master using this mechanism appropriately, if used incorrectly no IP Commands at all can be triggered.

Two different cases can be distinguished:

20.8.3.1 RX Buffer Read via QSPI_ARDB Registers

In this case all masters share the AHB bus for RX Buffer as well as for AHB Buffer read. In this case the access to the AHB interface by the master triggering AHB Commands must be deferred until any pending IP Command has been finished **and** the RX Buffer readout has been finished as well. The QSPI ARDB Buffers access the Rx buffer i.e the data from the Rx Buffer is returned and no data from AHB Buffer is touched. This is the conservative use case, corresponding to the reset value 0 of the QSPI_RBCT[RXBRD] bit.

In this case the QSPI_SR[AHBGNT] bit is asserted not earlier than any running IP Command has been finished (QSPI_SR[IP_ACC] is 0), the RX Buffer has been read out completely (QSPI_RBSR[RDBFL] equal to 0) or no DMA read is pending (QSPI_SR[RXDMA] equal to 0 and Rx Buffer readout is via AHB(QSPI_RBCT[RXBRD]) equal to 1.

20.8.3.2 RX Buffer Read via QSPI_RBDR Registers

This is the preferred use case as an access to the AHB buffer (memory mapped flash) does not interfere with any IPS access to read the RBDR buffer. It is not possible that a pending AHB bus access triggered by an AHB Command stalls the AHB bus and blocks the RX Buffer readout since the RX Buffer is read via the IP bus based registers QSPI_RBDR0 to QSPI_RBDR15.

For this case it is recommended to program the QSPI_RBCT[RXBRD] bit to 1. The QSPI_SR[AHBGNT] bit is asserted immediately after any running IP Command has been finished (QSPI_SR[IP_ACC] is 0), the RX Buffer has been read out completely (QSPI_RBSR[RDBFL] equal to 0) or no DMA read is pending (QSPI_SR[RXDMA] equal to 0, allowing the master triggering AHB Commands to trigger AHB Commands as soon as possible without the need to wait for the RX Buffer readout to be finished.

20.8.4 Command Arbitration

In case of overlapping commands, the arbitration scheme is described in the following paragraphs under the assumption that the priority mechanism described in Exclusive Access to Serial Flash for AHB Commands is **not** used:

- During the execution of an IP Command, the running IP Command can't be terminated by issuing another IP Command or AHB Command. The QSPI_FR[IPIEF] flag is asserted when the host tries to write into the QSPI_IPCR register. When the host triggers an AHB Command (refer to sub-section "Reading Serial Flash Data into the QuadSPI Module" of Flash Read section, for details), this command is stalled until the currently running IP Command is finished.
- During the execution of an AHB Command, the running AHB Command can't be terminated by issuing an IP Command. The command is ignored and the QSPI_FR[IPAEF] flag is asserted. Refer to Flag Register (QuadSPI_FR) for the description of these flags.

  When another AHB Command is triggered the address of the memory mapped access is considered. If the requested address is currently read from the serial flash device, the running command is continued. If this is not the case the currently running command is terminated and another AHB Command related to the requested address is executed. Refer to sub-section "Reading Serial Flash Data into the QuadSPI Module" of Flash Read section, for further details.

In case of coinciding commands the IP Command is triggered and the AHB Command is stalled until the IP Command has been finished (QSPI_SR[IP_ACC] has been deasserted).

The IP Commands ignored in case of command collision will not result in the assertion of the QSPI_FR[TFF] flag.

20.8.5 Flash Device Selection

Regardless of the SFM Command (IP or AHB) the access mode is selected by specifying the 32 bit address value for the following SFM Command.

For IP Commands the access mode is selected with the address programmed into the QSPI_SFAR register. Refer to Serial Flash Address Register (QuadSPI_SFAR) for details.

For AHB Commands the access mode is determined by the memory mapped address which is accessed Refer to Flash memory mapped AMBA bus for details.

20.8.6 DMA Usage

For the complete description of the DMA module refer to the related DMA Controller chapter. In this paragraph only the details specific to the DMA usage related to the QuadSPI module are given.

20.8.6.1 DMA Usage in Normal Mode

20.8.6.1.1 Bandwidth considerations

Careful consideration of the throughput rate of the entire chain (serial flash -> AHB bus / IP Bus -> DMA controller) involved in the read/write data process is essential for proper operation. Such analysis must take into account not only the data rate provided by the serial flash but also the data rate of the AHB bus and the performance of the DMA controller in reading/writing data from/to the RX/TX buffer.

Two figures must match for proper operation, that means that the data rate provided by the serial flash device must not exceed the average RX Buffer readout data rate. Otherwise, the longer this state persists, a RX Buffer overflow will result. Similarly, the data consumed by the serial flash device must not exceed the average TX buffer fill rate. If this persists, it would eventually lead to an underrun.

AHB Bus Side (data read):

The following table gives some examples for typical use cases:

Case 1: DMA need to read 4 bytes from SRAM and provide to QuadSPI. It costs total 4 BUS clock cycles. Then DMA handshake added additional 6 BUS clock cycles. Total [6 + 4 * (32/4) = 38] BUS clock cycles.

Table 20-21. Access Duration Examples - Bus Clock Side

QSPI_TBCT[WMRK]	Number of Bytes per DMA Loop	Number of Bus Clock cycles for DMA Minor Loop	Time Duration of DMA Minor Loop for 60Mhz Bus clock Frequency
3	16	6+(16/4)*4 = 22	~366ns
7	32	6+(32/4)*4 = 38	~633ns
11	48	6+(48/4)*4 = 54	~900ns
15	64	6+(64/4)*4 = 70	~1166ns

Case2: DMA need to read 32 bytes from SRAM and provide to QuadSPI DMA handshake takes additional 6 BUS cycles, with 32byte DMA read from SRAM costs (8 + 3) CORE clock cycles. DMA write 32 byte to QSPI takes 2 * (32/4) = 16 BUS cycles with one additional CPU access to QuadSPI, costing 2 BUS clock cycles. Total 6 + (8+3)/2 + 2 * (32/4) +2 = 30 BUS clock cycles.

Table 20-22. Access Duration Examples - Bus Clock Side

QSPI_TBCT[WMRK]	Number of Bytes per DMA Loop >	Number of Bus Clock cycles for DMA Minor Loop	Time Duration of DMA Minor Loop for 80Mhz Bus clock Frequency
3	16	6 + (4+3) /2 + (16/4)*2 + 2 = 20	~333ns
7	32	6 + (8+3) /2 + (32/4)*2 + 2 = 30	~500ns
11	48	6 + (4+3)/2*3 + (48/4)*2 + 2 = 44	~733ns
15	64	6 + (8+3) + (64/4)*2 + 2 = 51	~810ns

NOTE

The table figure represents ideal scenario, actual performance will depend on how the system is integrated.

Serial Flash Device Side (data read):

The number of serial flash cycles can be determined in the following way:

- Number of serial flash clock cycles required to read 4 bytes, corresponding to one RX Buffer entry (setup of command and address not considered): , 2 cycles for Octal DDR mode (Hyperflash) in individual flash mode , 8 cycles for Quad Mode (SDR) instructions in Individual Flash Mode etc.
- Overhead due to clock domain crossing : 1 cycle.

The following table lists the number of clock cycles required to read the data from the serial flash corresponding to the different settings of the QSPI_RBCT[WMRK] field:

Table 20-23. Access Duration Examples - Serial Flash side

QSPI_RBCT[WMRK] setting	Num Bytes per DMA Loop [1]	Num SCKFx for 80MHz SCKFx		Time duration of Flash data readout for 80MHz SCKFx (~12.5ns period)	
		IFM [2] Quad	IFM Quad DDR	IFM Quad	IFM Quad DDR
0	4	8	4	~100ns	~50ns
1	8	16	8	~200ns	~100ns
3	16	32	16	~400ns	~200ns
7	32	64	32	~800ns	~400ns
11	48	96	48	~1200ns	~600ns

1. DMA Loop means one Minor loop completion which is equivalent to one Major Loop iteration.
2. Individual flash mode.

NOTE

The table figure represents ideal scenario, actual performance will depend on how the system is integrated.

A complementary example would be when the watermark is set to be too high. In such a case, the time taken by the DMA to read out the RX buffer entries should be smaller than the time taken by the controller to push in the remaining entries in the buffer.

IPS Bus Side (data write):

The total number of bus cycles for each DMA Minor Loop completion is added from the following components:

- Overhead for each minor loop, given by DMA controller: Assume 10 cycles
- Overhead due to clock domain crossing: Assume 2 cycles
- Number of bus clock cycles required for 16 bytes (128 bit write size): Assume 4 cycles (read/write sequence of DMA controller)

Note that the size of the minor loop is determined by the size of the QSPI_TBCT[WMRK] field, therefore the overhead given above distributes among (QSPI_TBCT[WMRK]+1) write accesses of 32 bit each.

The following table gives some examples for typical use cases:

Table 20-24. Access Duration Examples - Bus Clock Side

QSPI_TBCT[WMRK]	Number of Bytes per DMA Loop [1]	Number of Bus Clock cycles for DMA Minor Loop	Time Duration of DMA Minor Loop for 80Mhz Bus clock Frequency
3	16	12+4 = 16	~200ns
7	32	12+8 = 20	~250ns

Table continues on the next page...

Table 20-24. Access Duration Examples - Bus Clock Side (continued)

QSPI_TBCT[WMRK]	Number of Bytes per DMA Loop [1]	Number of Bus Clock cycles for DMA Minor Loop	Time Duration of DMA Minor Loop for 80Mhz Bus clock Frequency
11	48	12+12 = 24	~300ns
15	64	12+16 = 28	~350ns

1. DMA Loop means one Minor Loop Completion which is equivalent to one.

NOTE

The table figure represents ideal scenario, actual performance will depend on how the system is integrated.

Serial Flash Device Side (data write):

The number of serial flash cycles can be determined in the following way:

- Number of serial flash clock cycles required to write 16 bytes, corresponding to four TX Buffer entry(setup of command and address not considered): 8 cycles for Octal DDR mode (Hyperflash) instructions in Individual Flash Mode, 32 cycles for Quad SDR writes in individual flash mode.
- Overhead due to clock domain crossing : 1 cycle.

The following table lists the number of clock cycles required to read the data from the serial flash corresponding to the different settings of the QSPI_TBCT[WMRK] field:

Table 20-25. Access Duration Examples - Serial Flash side

QSPI_TBCT[WMRK] setting	Num Bytes per DMA Loop [1]	Num SCKFx		Time duration for consuming data at Flash interface 100MHz SCKFx (10ns period)[2]		Time for fifo to get empty[3]	
		IFM [4] Quad	IFM Octal DDR	IFM Quad	IFM Octal DDR	IFM Quad	IFM Octal DDR
3	16	32	8	320ns	80ns	2240ns	560ns
7	32	64	16	640ns	160ns	1920ns	480ns
15	64	128	32	1280ns	320ns	1280ns	320ns

1. DMA Loop means one Minor loop completion which is equivalent to one Major Loop iteration.
2. Not all flash devices support writes at 100Mhz. Please refer to the flash datasheet for the actual page program frequency supported.
3. The assumption for these timings is that the TX Fifo was full when the transaction was initiated
4. Individual flash mode.

NOTE

The tables mentioned above are only examples which must be correlated with the DMA in the system.

From the examples given in the two tables above for TX fifo, it can be seen that depending on the relationship between the Bus clock and Serial flash clock frequencies, there are settings possible where the serial flash consumes data faster than the IPS bus can write data in TX buffer. In these cases, a TX buffer underrun situation will occur. To avoid TX Buffer underrun, the data transaction size should be large enough.

20.9 Byte Ordering - Endianness

QuadSPI provides support for swapping the flash read/write data based on the configuration of the QSPI_MCR[END_CFG]. By default the data is always returned in 64 bit LE format on the AHB bus and 32 bit LE format on the IPS interface when read via the RX buffer and written in 32 bit LE format when written via the TX buffer.

The table(QSPI_MCR[END_CFG]) below shows the complete bit ordering. BE signifies Big Endian which means the high order bits of the associated data vectors are associated with low order address positions. LE signifies Little Endian which means the lower order bits of the associated data vectors are associated with low order address positions.Refer to figure Figure 20-6

Table 20-26. QSPI_MCR[END_CFG]

00	64 bit BE
01	32 bit LE
10	32 bit BE
11	64 bit LE

The tables below (Byte ordering configuration in AHB) and (Byte ordering configuration in IPS) show how this configuration is implemented in QSPI AHB and IPS interfaces respectively. B in the table signifies Byte and the index 1-8 refers to the byte position i.e. 1 refer to bits[7:0], 8 refer to bits[63:56] and so on.

Table 20-27. Byte ordering configuration in AHB

64 bit BE	B1	B2	B3	B4	B5	B6	B7	B8
64 bit LE	B8	B7	B6	B5	B4	B3	B2	B1
32 bit BE	B5	B6	B7	B8	B1	B2	B3	B4
32 bit LE	B4	B3	B2	B1	B8	B7	B6	B5

Table 20-28. Byte ordering configuration in IPS

32BE	B1	B2	B3	B4
32LE	B4	B3	B2	B1

The examples below show the byte ordering in 64 bit BE configuration for AHB Buffer and 32 bit BE for TX/RX Buffer:

20.9.1 Programming Flash Data

CPU write instructions to the QSPI_TBDR register like

- Write QSPI_TBDR -> 0x01_02_03_04
- Write QSPI_TBDR -> 0x05_06_07_08

result in the following content of the TX Buffer:

Table 20-29. Example of QuadSPI TX Buffer

TX Buffer Entry	Content
0	32'h01_02_03_04
1	32'h05_06_07_08

Programming the TX Buffer into the external serial flash device results in the following byte order to be sent to the serial flash:

- 01…02…03…04…05…06…07…08

20.9.2 Reading Flash Data into the RX Buffer

Reading the content from the same address provides the following sequence of bytes, identical to the write case:

- 01…02…03…04…05…06…07…08

This results in the RX Buffer filled with:

Table 20-30. Resulting RX Buffer Content

RX Buffer Entry	Content
0	32'h01_02_03_04
1	32'h05_06_07_08

20.9.2.1 Readout of the RX Buffer via QSPI_RBDRn

The RX Buffer content appears at CPU read access via the Peripheral bus interface in the following order:

- Read QSPI_RBDR0 <- 0x01_02_03_04
- Read QSPI_RBDR1 <- 0x05_06_07_08

20.9.2.2 Readout of the RX Buffer via ARDBn

The RX Buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- (1a): 32 Bit Access: Read QSPI_ARDB0 <- 0x01_02_03_04
- (2a): 32 Bit Access: Read QSPI_ARDB1 <- 0x05_06_07_08
- (1b/2b): 64 Bit Access: Read QSPI_ARDB0 <- 0x01_02_03_04_05_06_07_08

20.9.3 Reading Flash Data into the AHB Buffer

Reading the content from the same address as it was written to provides the following sequence of bytes, identical to the write case:

- 01…02…03…04…05…06…07…08

This results in the AHB Buffer filled with:

Table 20-31. Resulting AHB Buffer Content

AHB Buffer Entry	Content
0	64'h01_02_03_04_05_06_07_08

20.9.3.1 Readout of the AHB Buffer via Memory Mapped Read

The AHB Buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- (1a): 32 Bit Read Access: <- 0x01_02_03_04

- (2a): 32 Bit Read Access: <- 0x05_06_07_08
- (1/2): 64 Bit Read Access: <- 0x01_02_03_04_05_06_07_08

20.10 Driving Flash Control Signals in Single and Dual Mode

In single and dual mode the serial flash devices which can connect to the QuadSPI module expect additional control signals on the inputs which are connected to IOFA[3], IOFA[2] in quad mode. For easy interfacing the outputs IOFA[3:2] for Flash A are driven to the logic state given by the configuration bits QSPI_MCR[ISD3FA], QSPI_MCR[ISD2FA].

These outputs are driven all the time to the logic level programmed in the QSPI_MCR register except the time when quad commands of the serial flash are executed. Refer to the specification of the related serial flash device for details about the inactive level.

20.11 Serial Flash Devices

Several different vendors make flash devices with a QuadSPI interface. At present there is no set standard for the QuadSPI instruction set. Most common commands currently have the same instruction code for all vendors, however some commands are unique to specific vendors. Some example sequences are provided below.

20.11.1 Example Sequences

This section provides the example sequences of the QuadSPI module.

Table 20-32. Exit 4 x I/O Read Enhance Performance Mode (XIP) (Macronix) and Read Status

INSTR	PAD	OPERAND	COMMENT
CMD	0x0	0xEB	4xIO Read Command
ADDR	0x2	0x18	24 Bit address to be send on 4 pads
MODE	0x2	0x00	2 mode cycles (exit XIP)
DUMMY	0x0	0x04	4 dummy cycles
READ	0x2	0x08	Read 64 bits
CMD	0x0	0x05	Read Status register
READ	0x0	0x01	Status register data
STOP	0x0	0x00	STOP, Instruction over

20.11.1.1 Read Command (Spansion Hyperflash)

This section provides the read command sequences (Spansion Hyperflash) of the QuadSPI module.

Table 20-33. Read Command (Spansion Hyperflash)

INSTR	PAD	OPERAND	COMMENT
CMD_DDR	0x3	0xA0	Read command with continuous burst type
ADDR_DDR	0x3	0x18	24 bit row address
CADDR_DDR	0x3	0x10	16 bit column address with lower 3 bits valid rest 0
DUMMY	0x3	0x0F	15 dummy cycles
READ_DDR	0x3	0x4	32 bit data read on 8 pads
STOP	0x3	0x00	STOP, Instruction over

Hyperflash is a word addressable flash i.e. each address accesses a word wide (2 bytes) data value, the software should ensure that when Hyperflash is connected to the controller, the QSPI_SFACR [WA] bit must be set. If this bit is set the controller remaps a byte addressable access to a word addressable access.

20.11.1.2 Read Status Register(Spansion Hyperflash)

This section provides the read status register of the QuadSPI module.

Table 20-34. Read Status register (Spansion Hyperflash)

INSTR	INSTR SEQUENCE	PAD	OPERAND	COMMENT
CMD_DDR	Read Pre Comman	0x3	0x00	Write command with wrapped burst type.
ADDR_DDR		0x3	0x18	24 bit row address(0000AAh)
CADDR_DDR		0x3	0x10	16 bit column address with lower 3 bits valid rest 0(0005h),treated as command
CMD_DDR		0x3	0x00	Write command with wrapped burst type
CMD_DDR		0x3	0x70	Write data to be sent to flash as pre-command

Table continues on the next page...

Table 20-34. Read Status register (Spansion Hyperflash) (continued)

CMD_DDR	Command phase (fourth/final chip select phase)	0x3	0xA0	Read command with continuous burst type
ADDR_DDR		0x3	0x18	24 bit row address
CADDR_DDR		0x3	0x10	16 bit column address with lower 3 bits valid rest 0
DUMMY		0x3	0x0F	15 dummy cycles
READ_DDR		0x3	0x4	32 bit data read on 8 pads
STOP		0x3	0x00	STOP, Instruction over

20.11.1.3 Word Program (Spansion Hyperflash)

This section provides the Word Program (Spansion Hyperflash) of the QuadSPI module.

Table 20-35. Word Program (Spansion Hyperflash)

INSTR	INSTR SEQUENCE	PAD	OPERAND	COMMENT
CMD_DDR	Unlock Sequence 1 (first chip select phase)	0x3	0x00	Write command with wrapped burst type
CMD_DDR		0x3	0x00	8 bit address 00h treated as command
CMD_DDR		0x3	0x00	8 bit address 00h treated as command
CMD_DDR		0x3	0xAA	8 bit address AAh treated as command
CADDR_DDR		0x3	0x10	16 bit column address with lower 3 bits valid rest 0(0005h),treated as command
CMD_DDR		0x3	0x00	Write command with wrapped burst type.
CMD_DDR		0x3	0xAA	Write data to be sent to flash as pre-command.
CMD_DDR	Unlock Sequence 2 (second chip select phase)	0x3	0x00	Write command with wrapped burst type
CMD_DDR		0x3	0x00	8 bit address 00h treated as command
CMD_DDR		0x3	0x00	8 bit address 00h treated as command
CMD_DDR		0x3	0x55	8 bit address 55h treated as command

Table continues on the next page...

Table 20-35. Word Program (Spansion Hyperflash) (continued)

CADDR_DDR		0x3	0x10	16 bit column address with lower 3 bits valid rest 0(0002h),treated as command
CMD_DDR		0x3	0x00	Write command with wrapped burst type
CMD_DDR		0x3	0x55	Write data to be sent to flash as pre-command
CMD_DDR	Program setup phase (third chip select phase)	0x3	0x00	Write command with wrapped burst type
CMD_DDR		0x3	0x00	8 bit address 00h treated as command
CMD_DDR		0x3	0x00	8 bit address 00h treated as command
CMD_DDR		0x3	0xAA	8 bit address AAh treated as command
CADDR_DDR		0x3	0x10	16 bit column address with lower 3 bits valid rest 0(0005h),treated as command
CMD_DDR		0x3	0x00	Write command with wrapped burst type
CMD_DDR		0x3	0xA0	Write data to be sent to flash as pre-command
CMD_DDR	Command phase (fourth/final chip select phase)	0x3	0x00	Write command with wrapped burst type
ADDR_DDR		0x3	0x18	24 bit row address
CADDR_DDR		0x3	0x10	16 bit column address with lower 3 bits valid rest 0
WRITE_DDR		0x3	0x2	2 bytes data written on 8 pads (D1D2)
STOP		0x3	0x00	STOP, Instruction over

20.11.1.4 Fast Read Sequence (Macronix/Numonyx/Spansion/ Winbond)

The following table shows the fast read sequence for Macronix/Numonyx/Spansion/ Winbond flashes.

Table 20-36. Fast Read sequence

Instruction	Pad	Operand	Comment
CMD	0x0	0x0B	Fast Read command = 0x0B
ADDR	0x0	0x18	24 Addr bits to be sent on one pad

Table continues on the next page...

Table 20-36. Fast Read sequence (continued)

Instruction	Pad	Operand	Comment
DUMMY	0x0	0x08	8 Dummy cycles
READ	0x0	0x04	Read 32 Bits on one pad
JMP_ON_CS	0x0	0x00	Jump to instruction 0 (CMD)

20.11.1.5 Fast Dual I/O DT Read Sequence (Macronix)

The following table shows the Fast Dual I/O DT read sequence for Macronix flashes.

Table 20-37. Fast Dual I/O DT Read sequence

Instruction	Pad	Operand	Comment
CMD	0x0	0xBD	Fast Dual I/O DT read command = 0xBD
ADDR_DDR	0x1	0x18	24 Addr bits to be sent on 2 pads in DDR mode
MODE4_DDR	0x1	0x00	P2=P0 or P3=P1 is necessary. Refer to Macronix datasheet for details. One clock cycle for mode.
DUMMY	0x0	0x06	6 Dummy cycles
READ_DDR	0x1	0x04	Read 32 Bits on 2 pads in DDR mode
JMP_ON_CS	0x0	0x00	Jump to instruction 0 (CMD)

20.11.1.6 Fast Read Quad Output (Winbond)

The following table shows the Fast read quad output sequence for Winbond memories

Table 20-38. Fast Read Quad output sequence

Instruction	Pad	Operand	Comment
CMD	0x0	0x6B	Fast read quad output command = 0x6B
ADDR	0x0	0x18	24 Addr bits to be sent on 1 pad
DUMMY	0x0	0x08	8 Dummy cycles
READ	0x2	0x04	Read 32 Bits on 4 pads
JMP_ON_CS	0x0	0x00	Jump to instruction 0 (CMD)

20.11.1.7 4 x I/O Read Enhance Performance Mode (XIP) (Macronix)

The following table shows the 4 x I/O Read Enhance Performance Mode for Macronix flashes. The enhanced performance mode is also known as XIP mode.

Table 20-39. Fast Read Quad output sequence

Instruction	Pad	Operand	Comment
CMD	0x0	0xEB	4xI/O Read command = 0xEB
ADDR	0x2	0x18	24 Addr bits to be sent on 4 pads
MODE	0x2	0xA5	2 mode cycles
DUMMY	0x0	0x04	4 Dummy cycles
READ	0x2	0x04	Read 32 Bits on 4 pads
JMP_ON_CS	0x0	0x01	Jump to instruction 1 (ADDR)

When in XIP mode the software should ensure that all the flashes connected to the controller are in XIP mode. As a part of initializing the controller, all the flashes may be enabled with XIP by carrying out dummy reads.

20.11.1.8 Dual Command Page Program (Numonyx)

The following table shows the Dual command page program sequence for Numonyx flashes.

Table 20-40. Dual Command Page Program sequence

Instruction	Pad	Operand	Comment
CMD	0x1	0x02	Dual command page program = 0x02 on 2 pads
ADDR	0x1	0x18	24 Addr bits to be sent on 2 pads
WRITE	0x1	0x20	Write 32 Bytes on 2 pads
STOP	0x0	0x00	STOP, Instruction over

20.11.1.9 Sector Erase (Macronix/Spansion/Numonyx)

The following table shows the Sector erase sequence for Macronix/Spansion/Numonyx flashes

Table 20-41. Sector Erase sequence

Instruction	Pad	Operand	Comment
CMD	0x0	0xD8	Sector erase command = 0xD8
ADDR	0x0	0x18	24 Addr bits to be sent on 1 pad
STOP	0x0	0x00	STOP, Instruction over

20.11.1.10 Read Status Register (Macronix/Spansion/Numonyx/Winbond)

The following table shows the Read status register sequence for Macronix/Spansion/Numonyx/Winbond flashes.

Table 20-42. Read Status Register Sequence

Instruction	Pad	Operand	Comment
CMD	0x0	0x05	Read status register command = 0x05
READ	0x0	0x01	Read status register data
STOP	0x0	0x00	STOP, Instruction over

20.11.1.11 Data Learn Instruction Sequence

The following table shows the data learn sequence for 4 I/O flash.

Table 20-43. Data learn Instruction sequence (4 I/O)

Instruction	Pad	Operand	Comment
CMD	0x0	0x6B	Fast Read Quad command = 0x6B
ADDR_DDR	0x2	0x18	24 address bits in DDR mode to be sent on four pads
DUMMY	0x2	0x08	8 Dummy cycle
DATA_LEARN	0x2	0x01	1 Byte data learn
READ_DDR	0x2	0x06	Read 6 bytes in 4 pads
JMP_ON_CS	0x0	0x00	Jump to Inst 0 (CMD)

The following table shows the data learn sequence for 8 I/O (Spansion Hyperflash) flash.

Table 20-44. Data learn Instruction sequence (8 I/O)

Instruction	Pad	Operand	Comment
CMD_DDR	0x3	0xAO	Read command for Hyperflash
ADDR_DDR	0x3	0x18	24 bit row address[1]
CADDR_DDR	0x3	0x10	16 bit column address
DUMMY	0x3	0x0F	15 cycle dummy
DATA_LEARN	0x3	0x01	1 byte data learn
READ_DDR	0x3	0x08	Read 8 bytes in 8 pads
STOP	0x3	0x00	STOP, instruction over

1. Address needs to be aligned i.e. No latency should be there between the RWDS edges.

20.11.2 Dual Die Flashes

Certain serial flash vendors provide dual-die packages which are essentially two devices (dies) stacked within the same package to increase the memory capacity of a single package. These two devices within a package share the same data and clock pins, but have individual Chip Selects. The figure below shows the dual-die package and the naming conventions used in this document. For simplicity, the data pins are shown to be unidirectional.

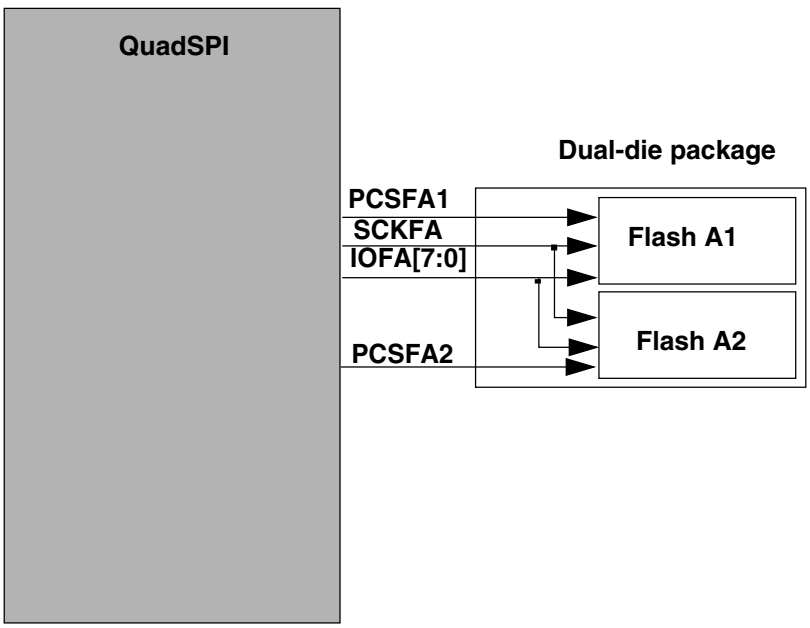


Figure 20-7. Dual-die support

Since the two devices within one package share the same i/o pads, they cannot function in parallel mode.

20.11.3 Boot initialization sequence

The following are the recommended sequence of steps for booting from QuadSPI:

- System out of reset and flash available (300us)
- Clocks still at very low frequency. Clock tree configured, I/O pins configured. First request sent to QuadSPI for address 0x0 of flash.
- The reset command sequence in QuadSPI has 0x03 (basic read command) which is applicable to all flashes at < 50MHz serial flash clock

- The first few bytes of data is read from the flash which contains the following information:
  - The total sizes of all the flashes connected on board
  - Whether DDR mode supported
  - Frequency of DDR operation
  - Continuous mode entry sequence
  - 24bit or 32bit addressing (assuming 24bit for first accesses)
- All the serial flashes are configured
  - Quad Mode enabled
  - Dummy reads to enter into XIP
- QuadSPI is configured
  - LUT configured for highest performance reads
  - DDR mode enabled (if applicable)
  - Buffers configured
- Serial flash clock frequency increased.
- Boot reads happen in DDR enabled, quad output mode @66MHz.

20.11.4 Serial Flash Clock Frequency Limitations

Certain commands of some serial flash devices are limited in the frequency applied to the serial flash device on command execution. In order to support these commands without having to recalibrate the module clocks, the the serial flash device clock can be divided by 2 (half speed) by setting the QSPI_SMPR[HSENA] bit. The SCLK will return to full speed once the the QSPI_SMPR[HSENA] bit is cleared.

20.12 Sampling of Serial Flash Input Data

20.12.1 Basic Description

QuadSPI is used to read data from the serial flash device. Depending on the actual implementation, there is a delay between the internal clocking in the QuadSPI module and the external serial flash device. Refer to the following figure for an overview of this scheme.

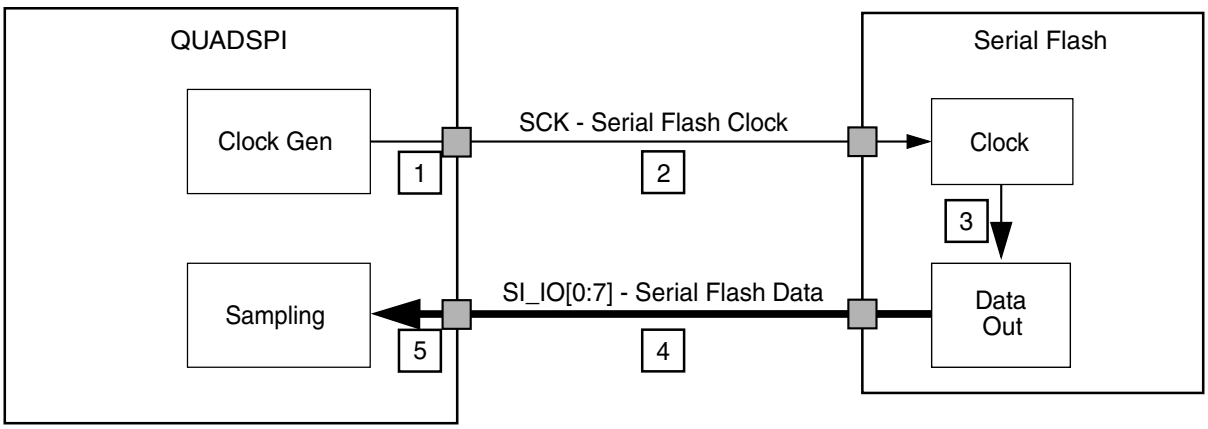


Figure 20-8. Serial Flash Sampling Clock Overview

The rising edge of the internal reference clock is taken as timing reference for the data output of the serial flash. After a time of $t_{Del,total}$ the data arrives at the internal sampling stage of the QuadSPI module. According to the Serial Flash Sampling Clock Overview figure, the following parts of the delay chain contribute to $t_{Del,total}$:

1. Output delay of the serial flash clock output of the device containing the QuadSPI module
2. Wire delay of application/PCB from the device containing the QuadSPI module to the external serial flash device
3. Clock to data out delay of the external serial flash device, including input and output delays
4. Wire delay of application/PCB from the external serial flash device to the device containing the QuadSPI module
5. Device delay corresponding to the input data

NOTE

The amount of total delay $t_{Del,total}$ is specific to the characteristics of the actual implementation. Also, the serial flash device clock (SCK) is inverted with respect to the QuadSPI internal reference clock.

20.12.2 Supported read modes

20.12.2.1 SDR mode

Most flash memories operate in single data rate (SDR) mode. In SDR mode, the data is transferred only on one edge of the clock signal. The SDR serial flash memories sample the incoming data on the rising edge of serial flash clock and drive the output data on the falling edge of the serial flash clock.

20.12.2.1.1 Internal sampling

QuadSPI uses different edges of the internal reference clock for sampling the input data in SDR mode.

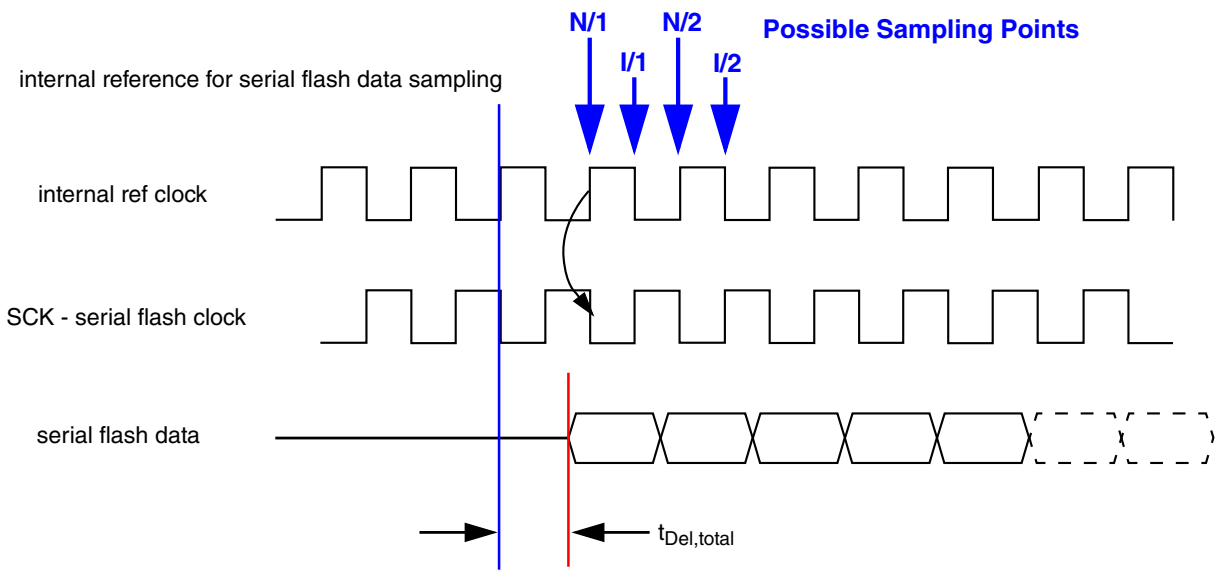


Figure 20-9. Internal sampling in SDR mode

The possible points in time for sampling incoming data are denoted as N/1, I/1, N/2 and I/2 above. The sampling point relevant for the internal sampling is configured in the QSPI_SMPR register. Refer to Sampling Register (QuadSPI_SMPR) for details. The following table gives an overview of the available configurations for the commands running at regular (full) speed:

Table 20-45. Sampling Configuration

Sampling Point	Description	Delay [FSDLY] [HSDLY]	Phase [FSPHS] [HSPHS]	QSPI_SMPR for Full Speed Setting[1]
N/1	sampling with non-inverted clock, 1 sample delay	0	0	0x0000000x
I/1	sampling with inverted clock, 1 sample delay	0	1	0x0000002x
N/2	sampling with non-inverted clock, 2 samples delay	1	0	0x0000004x
I/2	sampling with inverted clock, 2 samples delay	1	1	0x0000006x

1. 'x' is not considered here

Depending on the actual delay and the serial flash clock frequency, the appropriate sampling point can be chosen. The following remarks should be considered when selecting the appropriate setting:

- Theoretically there should be two settings possible to capture the correct data, since the serial flash output is valid for 1 clock cycle, disregarding rise and fall times and timing uncertainties.
- Depending on the timing uncertainties, it may turn out in actual applications that only one possible sample position remains. This is subject to careful consideration depending on the actual implementation.
- The delay $t_{Del,total}$ is an absolute size to shift the point in time when the serial flash data get valid at the QuadSPI input.
- For decreasing frequency of the serial flash clock the distance between the edges increases. So for large differences in the frequency the required setting may change.
- For commands running at half of the regular serial flash clock (QSPI_SMPR[HSENA] bit set) the sampling point must be figured separately to allow for the compensation of the absolute shift in time with respect to the sample-relative setting in the QSPI_SPMPR register.

20.12.2.1.2 DQS sampling method

Data sampling in SDR mode can be supported using the DQS sampling method. Refer to Data Strobe (DQS) sampling method for more details.

20.12.2.2 DDR Mode

The increasing requirement of improved throughput has introduced the double data rate (DDR) mode. In DDR mode, the data is transferred on both the rising and falling edges of the serial flash clock. The DDR serial flashes sample as well as drive the data on both rising and falling edges of serial flash clock.

20.12.2.2.1 Internal sampling (4x sampling method)

When the serial flash memories function in DDR mode, the data is valid for only half a clock cycle. This, along with the fact that the time for which the data is actually valid is smaller than half a clock cycle, requires that we provide closely spaced sampling points. The QuadSPI module provides a mechanism to sample the incoming data at multiple sampling points provided by a 4x serial flash clock in DDR mode. The figure below shows the different sampling points as configured by QuadSPI_SMPR[DDRSMP]. The FSDLY/FSPHS and HSDLY/HSPHS bits are ignored for DDR instructions.

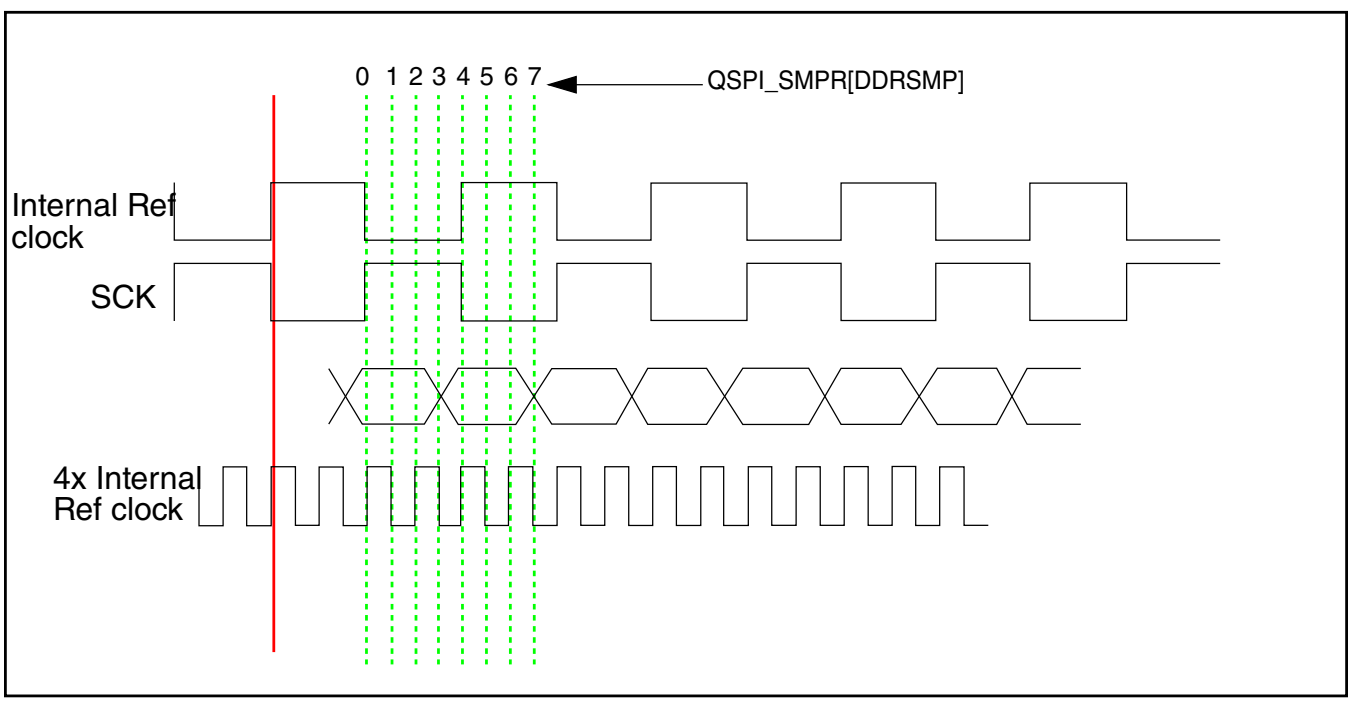


Figure 20-10. 4x sampling edges in DDR mode

Software should ensure that the correct sampling value is configured in the QuadSPI_SMPR[DDRSMP] register.

NOTE

Higher frequency can be achieved using data learning. For details, refer to Data Learning.

20.12.2.2.2 DQS sampling method

Data sampling in DDR mode can be supported using DQS method. Refer to Data Strobe (DQS) sampling method for more details.

A higher frequency can be achieved using Data learning. For details refer to Data Learning.

20.12.3 Data Strobe (DQS) sampling method

20.12.3.1 Basic Description

In DQS mode, the data strobe signal (DQS/RWDS) is used to sample the read data. Here, both DQS and the data sent by the flash move in the same direction, so it is relatively easier to achieve at higher frequencies.

When using DQS for SDR reads, QuadSPI internally samples the incoming data on rising edge of the strobe signal.

The figure below shows sampling read data in SDR mode using DQS.

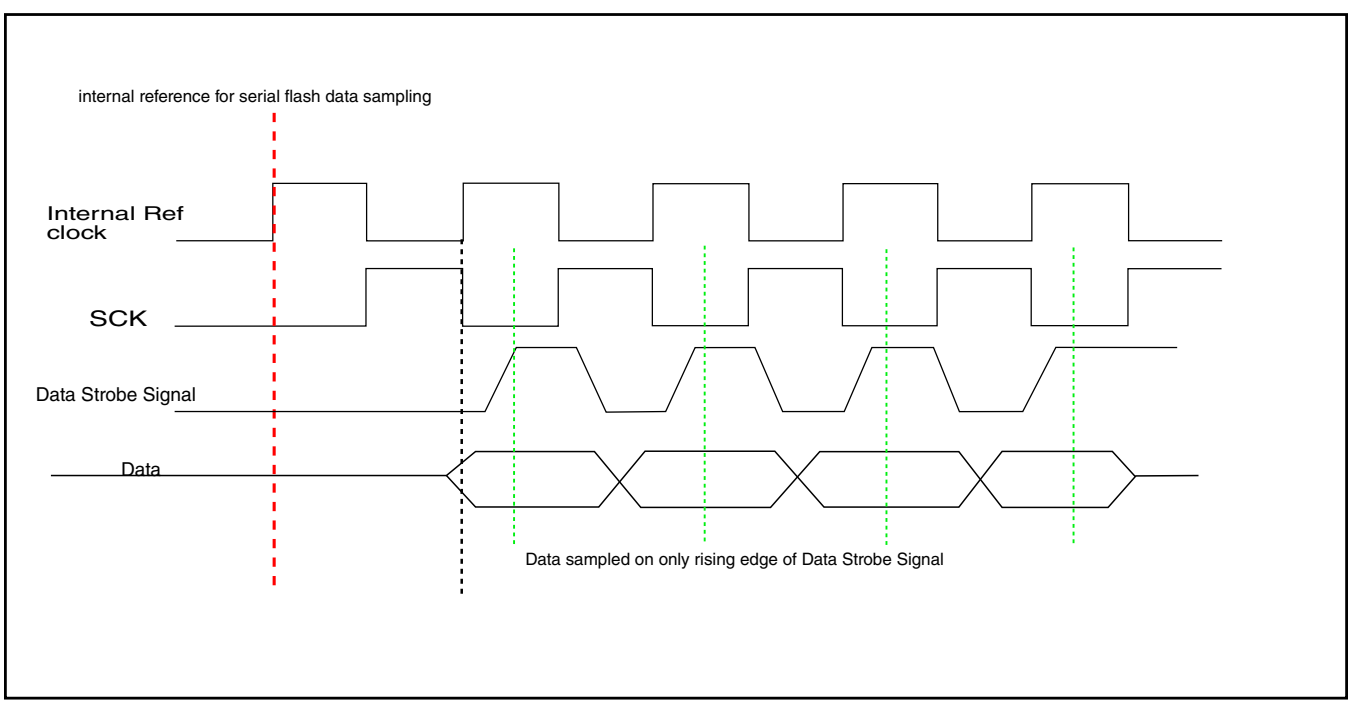


Figure 20-11. Data Strobe functionality in SDR mode

When using DQS for DDR reads, QuadSPI internally samples the incoming data on both the edges of the strobe signal. Refer to the figure below for more detail.

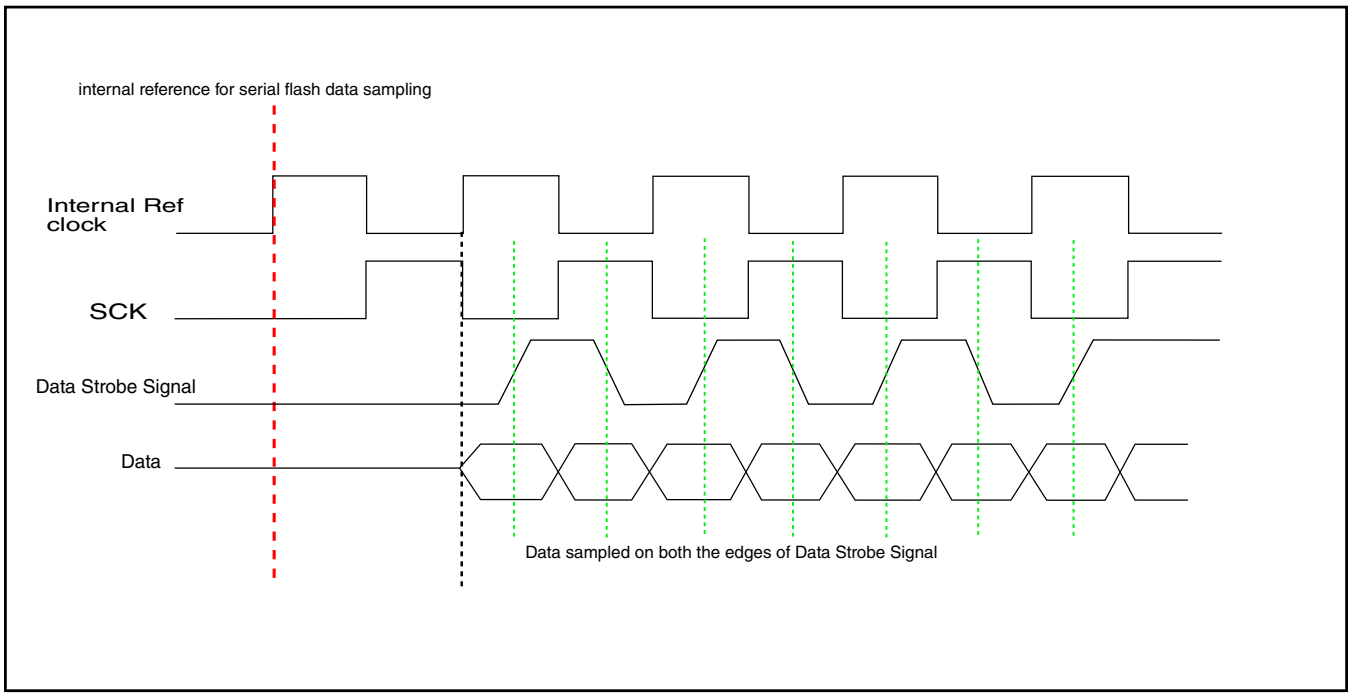


Figure 20-12. Data Strobe functionality in DDR mode

20.12.3.2 Internally generated DQS

In this mode, the internal reference clock is fed as a strobe to QuadSPI for read data sampling. The data strobe must be generated in a way such that the data is correctly sampled by the QuadSPI module. This internally generated data strobe signal can be used by QuadSPI to capture the data in:

- SDR mode
- DDR mode

Refer to QuadSPI chip-specific information for additional details about internally generated DQS.

20.12.3.3 External DQS

In serial flash memories supporting DQS, the data strobe signal is an output from the flash device that indicates when data is being transferred from the flash to the host controller. The data is then captured by the controller on:

- Only one edge (either rising or falling edge) of DQS signal in SDR mode
- Both rising/falling edge of the DQS signal in DDR mode

Some serial flash memories (for example, HyperFlash) provide the data strobe output (RWDS) with latency cycles included in between the ongoing read transaction. The data is sampled on both the edges of this signal taking into consideration that when no signal is provided by flash between ongoing transaction, data is not sampled then and at the end of transaction the required number of data to be fetched remains the same. Refer to the figure below for more detail.

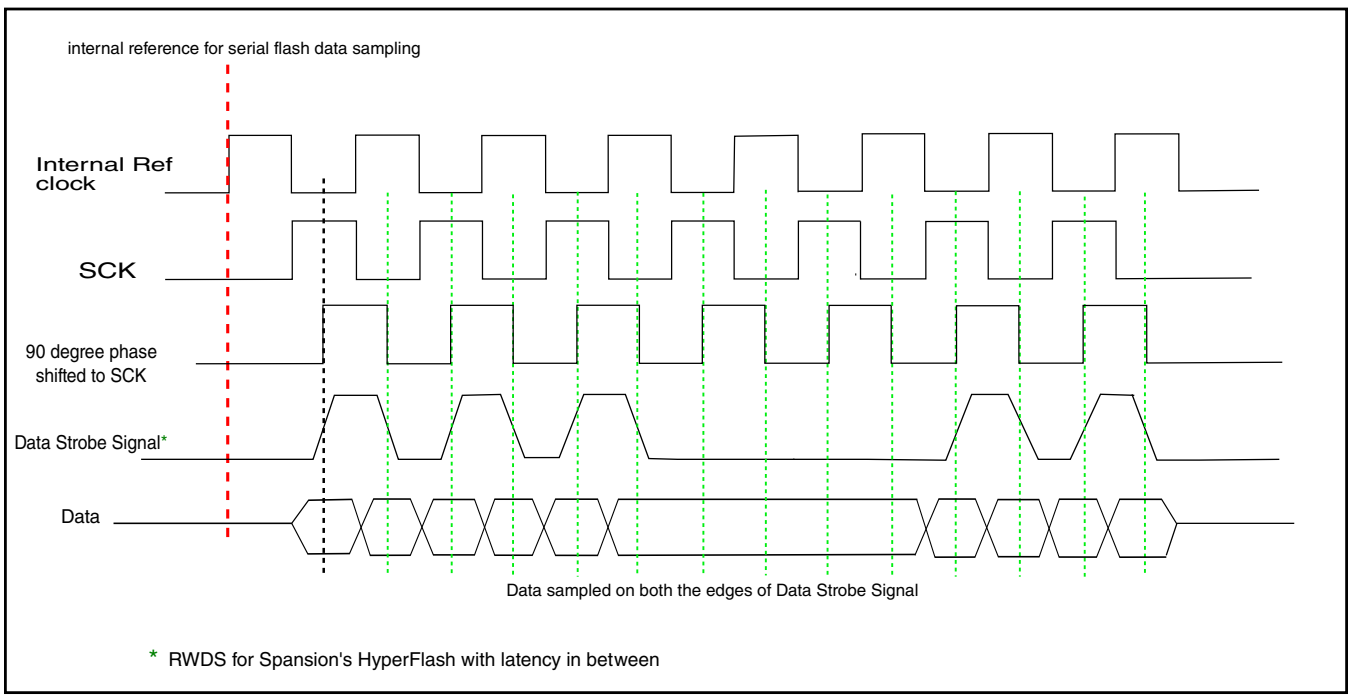


Figure 20-13. Data Strobe functionality with latency included

20.12.3.3.1 External Center Aligned Read Strobe

The QuadSPI supports serial flash memories using the Center Aligned Read Strobe (CRS) protocol. By virtue of this protocol, in addition to the SCK, the host controller provides a 90^{o} phase shifted SCK that allows the flash memory to generate valid data timed off SCK edges and a Data Read Strobe (DQS) signal output that is timed off the 90^{o} phase shifted SCK edges that is centered within the valid read data window. This allows the host controller to easily capture read data from the flash by using the data read strobe transitions and thereby allows for higher performance operation. Since the flash memory is the originator of the center aligned data read strobe as well as the read data, the temporal timing phase alignment between these signals is easier to guarantee across PVT (Process, Voltage and Temperature) corners.

The figure below shows data strobe functionality using CRS protocol.

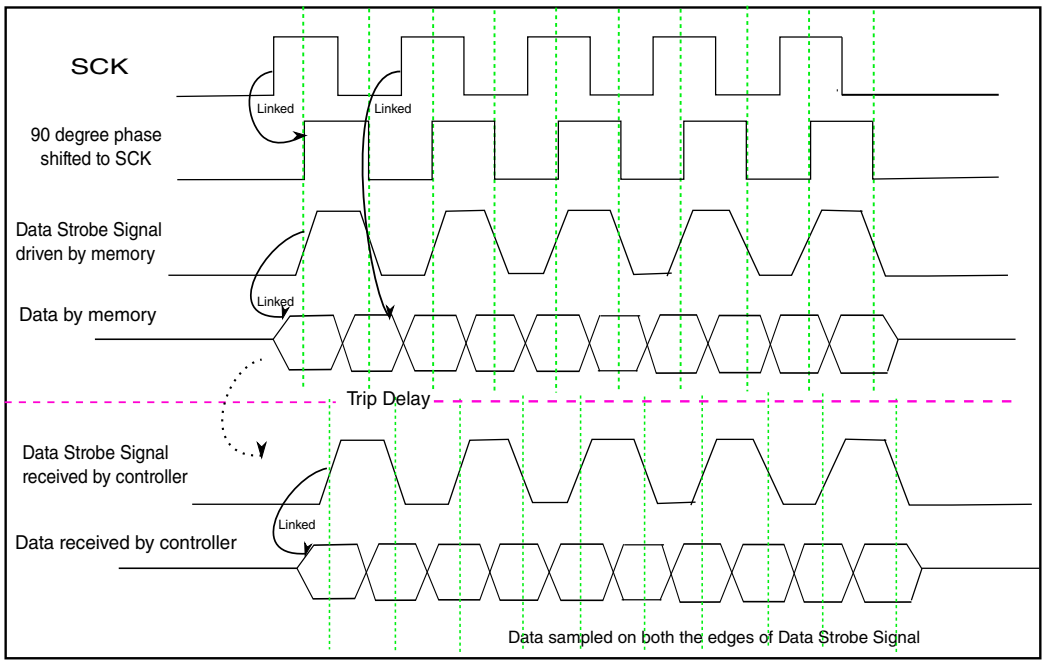


Figure 20-14. Data Strobe functionality using CRS protocol

20.12.4 Data Learning

20.12.4.1 Basic description

“Data learning” is used to manage varying data valid windows from flash memory as well as any variations in the chip based on PVT conditions in DDR mode. QuadSPI provides this feature via the DATA_LEARN instruction for all flash memories, irrespective of whether the flash supports it. The DATA_LEARN instruction accepts an operand that defines the number of bits of the known pattern for which the data learning has to be done. The known pattern is provided in the QSPI_DLPR register.

QuadSPI supports the data learning methods below:

- 4x sampling method
- Data Strobe (DQS) sampling method

The following sections explain data learning in greater detail.

20.12.4.2 4x sampling method

Automatic data learning is supported in 4x sampling mode. To start data learning:

1. Certain flash memories provide data learning as a feature for DDR data reads.

   For flash memories that support data learning, configure a known pattern in the Data Learn register inside flash. The pattern should be chosen to have multiple low and high transitions in the data bus. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction before the READ_DDR instruction. The flash will itself return the data learning pattern in between dummy cycles in every read command. Each I/O will output the same DLP value for every clock edge.

   For flash memories that *do not* support data learning, configure a known location in flash memory with a known pattern. The pattern should not have 0x00 or 0xFF in it. The pattern should be chosen to have multiple low and high transitions in the data bus. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction before the READ_DDR instruction. The address in the QuadSPI_SFAR register should point to the known location with the pattern. Configure the known pattern in the QuadSPI_DLPR [DLPV] field. Refer to Table 20-46 for details on how the data has to be ordered in memory for correct operation.

2. Select a sampling point using QuadSPI_SMPR[DDRSMP].
3. Initiate the read via a peripheral transaction.
4. QuadSPI reads the data from the flash. It then encounters the DATA_LEARN instruction and samples the incoming data on all 8 possible sampling points for all the data bus bits.
5. The sample point at which data matches the known pattern is reported in QuadSPI_SR[DLPSMP].
6. If data doesn't match at any sample point, QuadSPI_FR[DLPFF] is set.
7. If the correct sampling point is located, QuadSPI uses the new sampling point to sample the read data following the DATA_LEARN instruction.

This feature may be used to auto-calibrate the sampling point for DDR reads. This auto-calibration may be triggered at fixed intervals, or depending on a change in PVT conditions.

The following figure shows data learning with 4x DDR sampling:

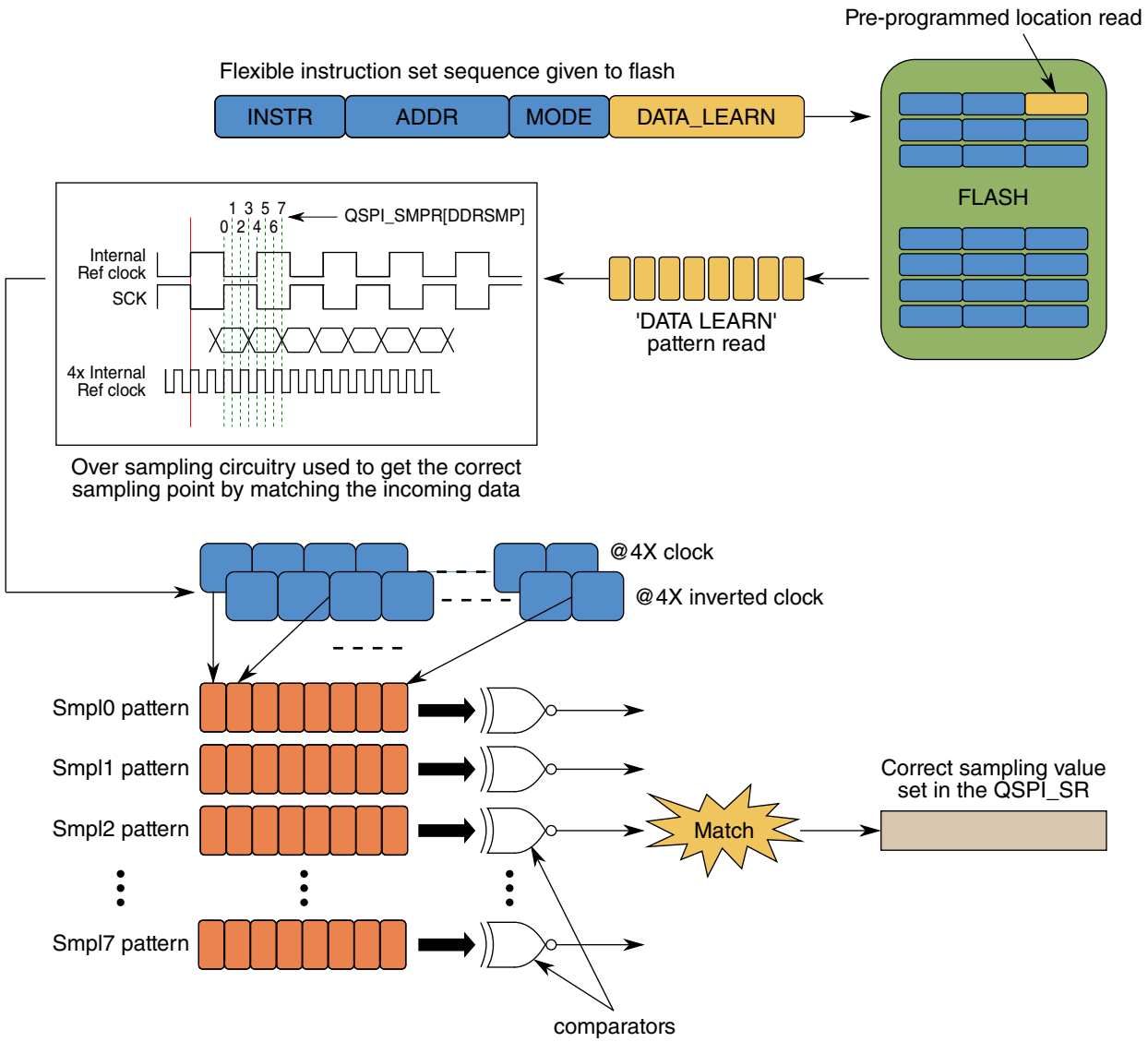


Figure 20-15. Data learning with 4x sampling method

20.12.4.3 Data Strobe (DQS) sampling method

Semi-automatic data learning is supported in DQS sampling method. To start data learning:

1. Certain flash memories provide data learning as a feature for DDR data reads.

   For flash memories that support data learning, configure a known pattern in the Data Learn register inside flash. The pattern should be chosen to have multiple low and high transitions in the data bus. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction before the READ_DDR instruction. The flash will itself return the data learning pattern in between dummy cycles in every read command. Each I/O will output the same DLP value for every clock edge.

For flash memories that *do not* support data learning, configure a known location in flash memory with a known pattern. The pattern should not have 0x00 or 0xFF in it. The pattern should be chosen to have multiple low and high transitions in the data bus. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction before the READ_DDR instruction. The address in the QSPI_SFAR register should point to the known location with the pattern. Configure the known pattern in the QSPI_DLPR[DLPV] field. Refer to Table 20-46 for details on how the data has to be ordered in memory for correct operation.

2. Select a sampling point. See chip-specific QuadSPI information for selection of the sampling point.
3. Initiate the read via a peripheral transaction.
4. QuadSPI reads the data from the flash. It then encounters the DATA_LEARN instruction and samples the incoming data on both edges (rising and falling) of the DQS.
5. If the data from the flash doesn't match the data learning pattern, the QuadSPI_FR[DLPFF] flag is set.
6. No sampling point is reported automatically by the QuadSPI module.
7. These steps (2-7) are repeated with varying VT conditions for a particular process until the QuadSPI_FR[DLPFF] is set. The sampling points where no QuadSPI_FR[DLPFF] is set signify valid setting.
8. In the case of multiple valid settings, software should choose the middle point.
9. The sampling point should be fixed with the above chosen point for the next READ transactions.

NOTE

It must be ensured that QuadSPI is not accessed during calibration.

NOTE

Ensure that there is no latency in between DQS edges while doing data learning using DQS sampling method.

This feature may be used to auto-calibrate the sampling point for DDR reads. This auto-calibration may be triggered at fixed intervals, or depending on a change in PVT conditions.

The following figure shows data learning in DQS mode:

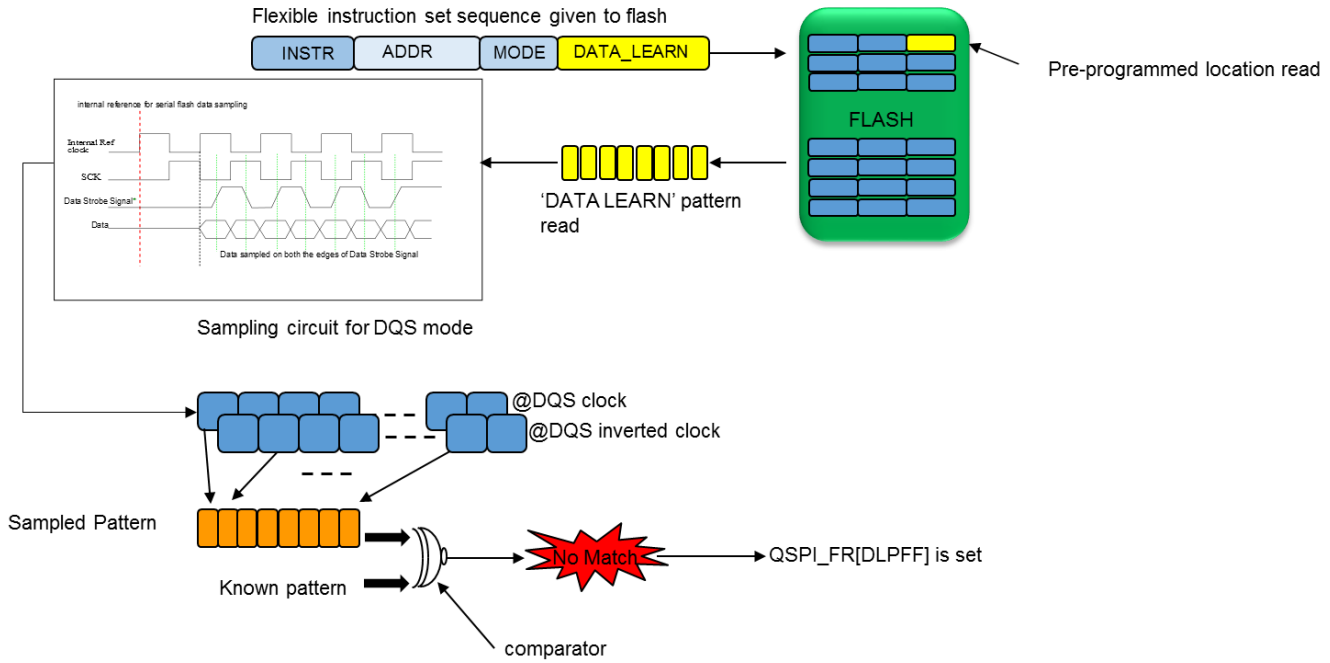


Figure 20-16. Data learning with DQS sampling method

20.12.4.4 Example: programming the data learning pattern in flash memory

If QSPI_DLPR[DLPV] is set to 0x43, QuadSPI tries to match this sample on every incoming data line for single, dual, quad, or octal configurations. The table below shows the data programmed in flash with endianness taken into consideration.

Table 20-46. Programming the Data Learning Pattern in flash memory

Endianness	Single IO	Dual IO	Quad IO	Octal IO
BE	0x43	0x300F	0x0F0000FF	0x00FF00000000FFFF
LE	0x43	0X0F30	0XFF00000F	0XFFFF0000000000FF

20.13 Data Input Hold Requirement of Flash

In DDR mode, the data is valid only for half clock cycle. It is difficult to meet the data input hold time requirement of flash. QuadSPI internally delays the data sent to flash so that it will be easy to meet the hold requirement. The QSPI_FLSHCR[TDH] is used for this purpose. If QSPI_FLSHCR[TDH] is configured to 0, the data sent to flash is aligned

with the internal reference clock of QuadSPI, if it is 1, then, the data is aligned to 2x internal reference half clock, if it is 2, then, the data is aligned to 4x internal reference half clock.

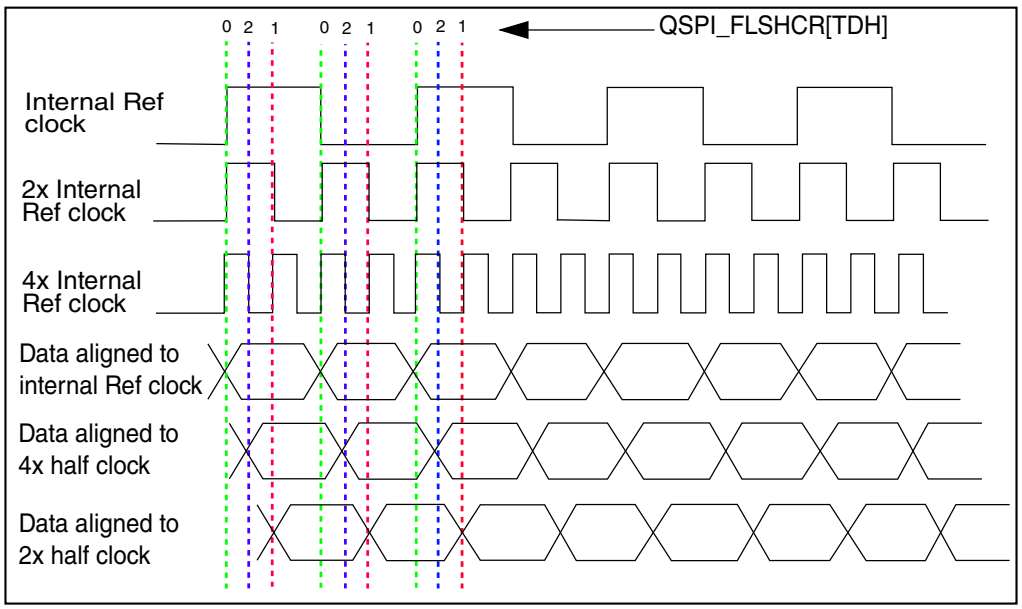


Figure 20-17. Data Hold

Chapter 21
FlexBus

21.1 Chip-specific FlexBus information

Table 21-1. Reference links to related information

Topic	Related module	Reference
Full description	FlexBus	FlexBus
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

21.1.1 External Bus Interface (FlexBus)

The External Bus Interface (FlexBus) module provides external memory expansion and provides connection to external peripherals with a parallel, memory-mapped interface. The FlexBus supports asynchronous and synchronous interface to external ROM, NOR flash, SRAM, PSRAM, programmble logic devices and other memory-mapped slave devices. The FlexBus supports 8-bit, 16-bit and 32-bit data bus width options and the multiplexing of address and data configuration. The following table shows the configuration of the FlexBus.

Table 21-2. Flexbus configuration

Parameter	Description
Name	External Bus Interface (FlexBus)
Instances	1
Configurable features	NA
Interface speed	Up to 100 MHz
External I/O pins	See the attached IOMUXC spreadsheet for pin details.

21.1.2 Transfer Acknowledge signal

The Transfer Acknowledge Signal ($\overline{\text{FB\_TA}}$) is hard-wired in the design of I.MX 7ULP, so this signal is not available. See the FB_CSPMCR register for the multiplexing of other signals.

21.1.3 Reset value for CSCRn register

The reset values of CSCR0-CSCR5 for i.MX 7ULP are specified in the table below

Table 21-3. CSCRn reset values

Register name	Reset value
CSCR0	0x003F_FC00
CSCR1-CSCR5	0x0000_0000

21.2 Introduction

The FlexBus multifunction external bus interface controller:

- Provides memory expansion and provides connection to external peripherals with a parallel bus
- Provides connections to external peripherals with a parallel bus, and is often used to expand memory
- Can be directly connected to asynchronous or synchronous slave-only devices, using little or no additional circuitry:
  - External ROMs
  - Flash memories
  - Programmable logic devices
  - Other simple target (slave) devices

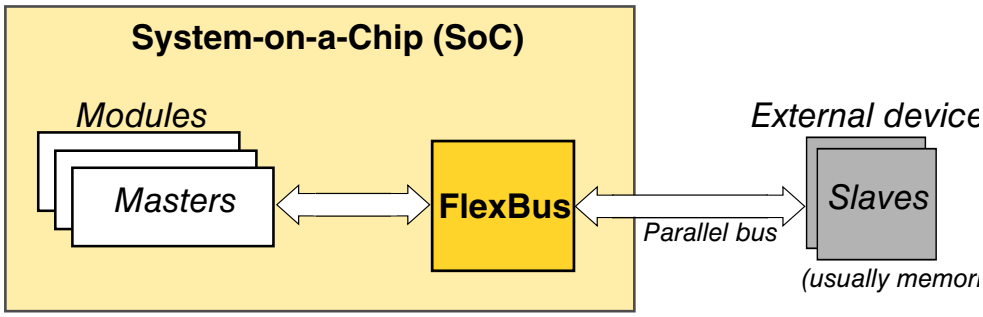


Figure 21-1. Flexbus module in a System-on-a-Chip (SoC)

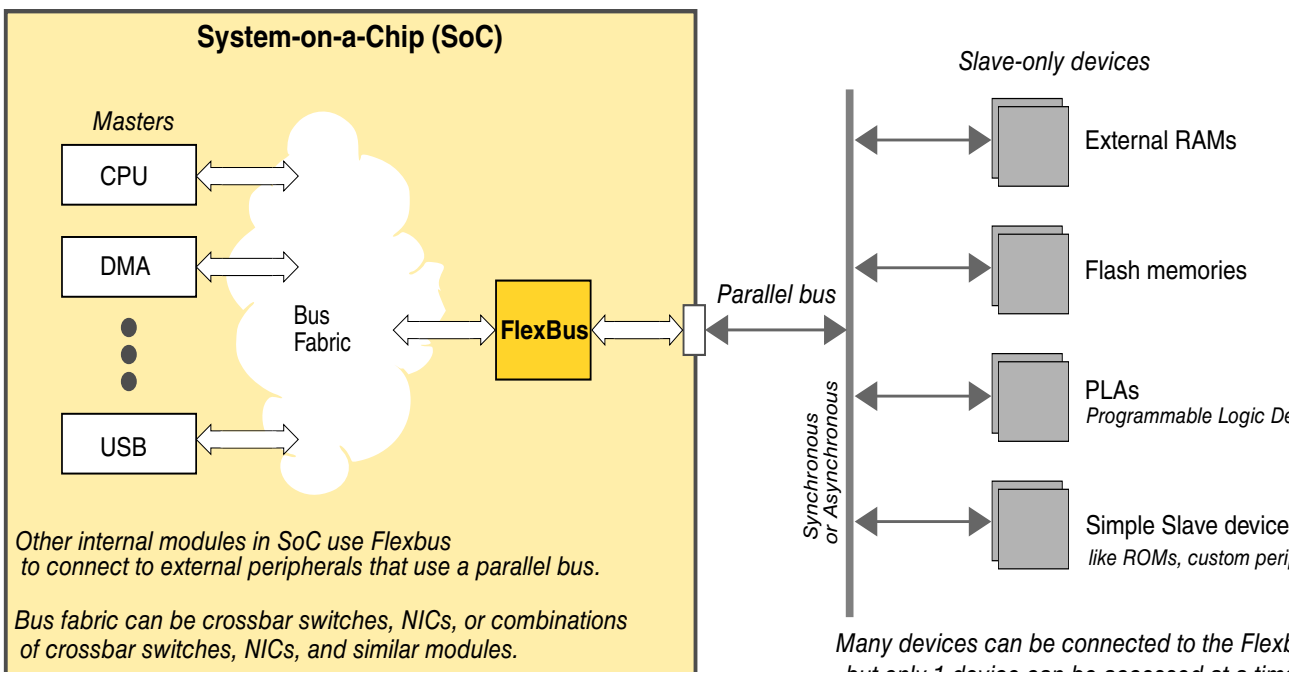


Figure 21-2. Flexbus module in SoC details

FlexBus features:

- 6 independent, user-programmable chip-select signals (active low) (FB_CS 5 – FB_CS0)
- 8-bit, 16-bit, and 32-bit transfers
- Programmable burst and burst-inhibited transfers selectable for each chip-select and transfer direction
- Programmable address-setup time with reference to the assertion of a chip-select
- Programmable address-hold time with reference to the deassertion of a chip-select and transfer direction
- Extended transfer start/address latch enable option to assist with glueless connections to synchronous and asynchronous memory devices

21.3 Signal descriptions

This table describes the external signals involved in data-transfer operations.

NOTE

Not all of the following signals may be available on a particular device. See the Chip Configuration details for information on which signals are available.

NOTE

Throughout this document, active low signals are indicated by one of 3 methods:

- "_b" appended to the end of a signal name
- a black line (called an "overbar") over the signal name
- a phrase "active low" next to the signal name

Table 21-4. FlexBus signal descriptions

Signal	I/O	Function
FB_AD31 - FB_AD0 (multiplexed address/ data bus)	I/O	When FlexBus is used in a multiplexed configuration, this is the address and data bus, FB_AD. The number of byte lanes carrying the data is determined by the port size associated with the matching chip-select. The full 32-bit address is driven on the first clock of a bus cycle (address phase). After the first clock, the data is driven on the bus (data phase). During the data phase, the address is driven on the pins not used for data. For example, in 16-bit mode, the lower address is driven on FB_AD15–FB_AD0, and in 8-bit mode, the lower address is driven on FB_AD23–FB_AD0.
FB_CS5 – FB_CS0	O	General Purpose Chip-Selects (active low)—Indicate which external memory or peripheral is selected. A particular chip-select is asserted when the transfer address is within the external memory's or peripheral's address space, as defined in CSAR[BA] and CSMR[BAM].
FB_BE_31_24 FB_BE_23_16 FB_BE_15_8 FB_BE_7_0	O	Byte Enables (active low)—Indicate that data is to be latched or driven onto a specific byte lane of the data bus. CSCR[BEM] determines if these signals are asserted on reads and writes or on writes only. For external SRAM or flash devices, the active low FB_BE outputs should be connected to individual byte strobe signals.
FB_OE_b	O	Output Enable (active low)—Sent to the external memory or peripheral to enable a read transfer. This signal is asserted during read accesses only when a chip-select matches the current address decode.
FB_R/W_b	O	Read/Write—Indicates whether the current bus operation is a read operation (FB_R/W_b high) or a write operation (FB_R/W_b low).
FB_TS_b	O	Transfer Start—Indicates that the chip has begun a bus transaction and that the address and attributes are valid. An inverted FB_TS_b is available as an address latch enable (FB_ALE), which indicates when the address is being driven on the FB_AD bus.

Table continues on the next page...

Table 21-4. FlexBus signal descriptions (continued)

Signal	I/O	Function
		FB_TS_b /FB_ALE is asserted for one bus clock cycle. The chip can extend this signal until the first positive clock edge after active low FB_CS asserts. See CSCR[EXTS] and Extended Transfer Start/Address Latch Enable.
FB_ALE	O	Address Latch Enable—Indicates when the address is being driven on the FB_A bus (inverse of FB_TS_b).
FB_TSIZ1–FB_TSIZ0	O	Transfer Size—Indicates (along with FB_TBST_b) the data transfer size of the current bus operation. The interface supports 8-, 16-, and 32-bit operand transfers and allows accesses to 8-, 16-, and 32-bit data ports. • 00b = 4 bytes • 01b = 1 byte • 10b = 2 bytes • 11b = 16 bytes (line) For misaligned transfers, FB_TSIZ1–FB_TSIZ0 indicate the size of each transfer. For example, if a 32-bit access through a 32-bit port device occurs at a misaligned offset of 1h, 8 bits are transferred first (FB_TSIZ1–FB_TSIZ0 = 01b), 16 bits are transferred next at offset 2h (FB_TSIZ1–FB_TSIZ0 = 10b), and the final 8 bits are transferred at offset 4h (FB_TSIZ1–FB_TSIZ0 = 01b). For aligned transfers larger than the port size, FB_TSIZ1–FB_TSIZ0 behave as follows: • If bursting is used, FB_TSIZ1–FB_TSIZ0 are driven to the transfer size. • If bursting is inhibited, FB_TSIZ1–FB_TSIZ0 first show the entire transfer size and then show the port size. For burst-inhibited transfers, FB_TSIZ1–FB_TSIZ0 change with each FB_TS_b assertion to reflect the next transfer size. For transfers to port sizes smaller than the transfer size, FB_TSIZ1–FB_TSIZ0 indicate the size of the entire transfer on the first access and the size of the current port transfer on subsequent transfers. For example, for a 32-bit write to an 8-bit port, FB_TSIZ1–FB_TSIZ0 are 00b for the first transaction and 01b for the next three transactions. If bursting is used for a 32-bit write to an 8-bit port, FB_TSIZ1–FB_TSIZ0 are driven to 00b for the entire transfer.
FB_TBST_b	O	Transfer Burst—Indicates that a burst transfer is in progress as driven by the chip. A burst transfer can be 2 to 16 beats depending on FB_TSIZ1–FB_TSIZ0 and the port size. **Note:** When a burst transfer is in progress (FB_TBST_b = 0b), the transfer size is 16 bytes (FB_TSIZ1–FB_TSIZ0 = 11b), and the address is misaligned within the 16-byte boundary, the external memory or peripheral must be able to wrap around the address.
FB_TA_b	I	Transfer Acknowledge—Indicates that the external data transfer is complete. When FB_TA_b is asserted during a read transfer, FlexBus latches the data and then terminates the transfer. When FB_TA_b is asserted during a write transfer, the transfer is terminated. If auto-acknowledge is disabled (CSCR[AA] = 0), the external memory or peripheral drives FB_TA_b to terminate the transfer. If auto-acknowledge is enabled (CSCR[AA] = 1), FB_TA_b is generated internally after a specified number of wait states, or the external memory or peripheral may assert external FB_TA_b before the wait-state countdown to terminate the transfer early. The chip deasserts active low FB_CS one

Table continues on the next page...

Table 21-4. FlexBus signal descriptions (continued)

Signal	I/O	Function
		cycle after the last FB_TA_b is asserted. During read transfers, the external memory or peripheral must continue to drive data until FB_TA_b is recognized. For write transfers, the chip continues driving data one clock cycle after active low FB_CS is deasserted. The number of wait states is determined by CSCR or the external FB_TA_b input. If the external FB_TA_b is used, the external memory or peripheral has complete control of the number of wait states. **Note:** External memory or peripherals should assert FB_TA_b only while the active low FB_CS signal to the external memory or peripheral is asserted. The CSPMCR register controls muxing of FB_TA_b with other signals. When the CSPMCR register does not allow FB_TA_b control, auto-acknowledge must be used (CSCR[AA] =1'b1); otherwise the bus may hang.
FB_CLK	O	FlexBus Clock Output

21.4 Memory Map/Register Definition

The following tables describe the registers and bit meanings for configuring chip-select operation.

The actual number of chip selects available depends upon the device and its pin configuration. If the device does not support certain chip select signals or the pin is not configured for a chip-select function, then that corresponding set of chip-select registers has no effect on an external pin.

Note

You must set CSMR0[V] before the chip select registers take effect.

A bus error occurs when writing to reserved register locations (for example, when writing to the region between the CS registers and the pin mux). Addresses *base address* + (0x48 - 0x5C) are reserved.

21.4.1 FB register descriptions

21.4.1.1 FB Memory map

FB base address: 4010_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Chip Select Address Register (CSAR0)	32	RW	0000_0000h
4h	Chip Select Mask Register (CSMR0)	32	RW	0000_0000h
8h	Chip Select Control Register (CSCR0)	32	RW	See description.
Ch	Chip Select Address Register (CSAR1)	32	RW	0000_0000h
10h	Chip Select Mask Register (CSMR1)	32	RW	0000_0000h
14h	Chip Select Control Register (CSCR1)	32	RW	See description.
18h	Chip Select Address Register (CSAR2)	32	RW	0000_0000h
1Ch	Chip Select Mask Register (CSMR2)	32	RW	0000_0000h
20h	Chip Select Control Register (CSCR2)	32	RW	See description.
24h	Chip Select Address Register (CSAR3)	32	RW	0000_0000h
28h	Chip Select Mask Register (CSMR3)	32	RW	0000_0000h
2Ch	Chip Select Control Register (CSCR3)	32	RW	See description.
30h	Chip Select Address Register (CSAR4)	32	RW	0000_0000h
34h	Chip Select Mask Register (CSMR4)	32	RW	0000_0000h
38h	Chip Select Control Register (CSCR4)	32	RW	See description.
3Ch	Chip Select Address Register (CSAR5)	32	RW	0000_0000h
40h	Chip Select Mask Register (CSMR5)	32	RW	0000_0000h
44h	Chip Select Control Register (CSCR5)	32	RW	See description.
60h	Chip Select Port Multiplexing Control Register (CSPMCR)	32	RW	0000_0000h

21.4.1.2 Chip Select Address Register (CSAR0 - CSAR5)

21.4.1.2.1 Offset

Register	Offset
CSAR0	0h
CSAR1	Ch
CSAR2	18h
CSAR3	24h

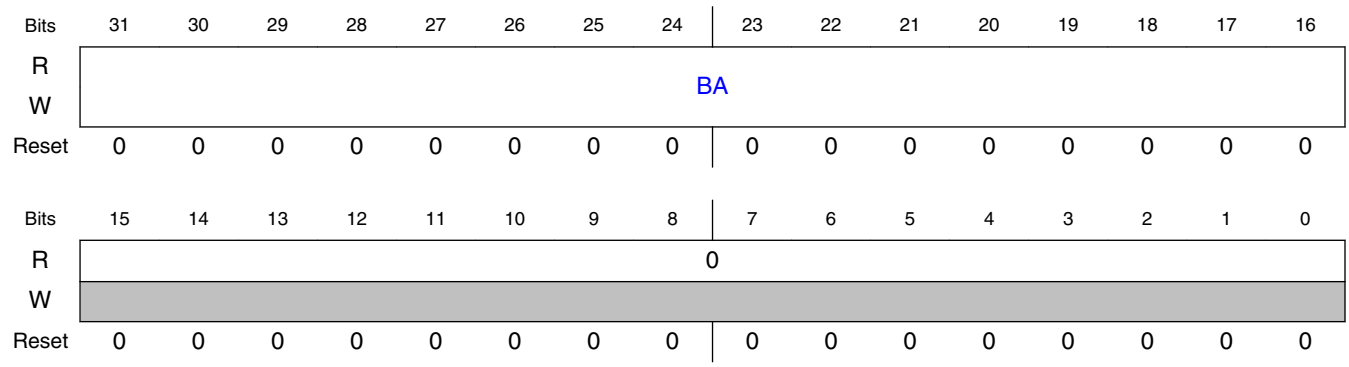
Table continues on the next page...

Register	Offset
CSAR4	30h
CSAR5	3Ch

21.4.1.2.2 Function

Specifies the associated chip-select's base address.

21.4.1.2.3 Diagram



21.4.1.2.4 Fields

Field	Function
31-16 BA	Base Address Defines the base address for memory dedicated to the associated chip-select. BA is compared to bits 31–16 on the internal address bus to determine if the associated chip-select's memory is being accessed. **NOTE:** Because the FlexBus module is one of the slaves connected to the crossbar switch, it is only accessible within a certain memory range. See the chip memory map for the applicable FlexBus "expansion" address range for which the chip-selects can be active. Set the CSARn and CSMRn registers appropriately before accessing this region.
15-0 —	Reserved

21.4.1.3 Chip Select Mask Register (CSMR0 - CSMR5)

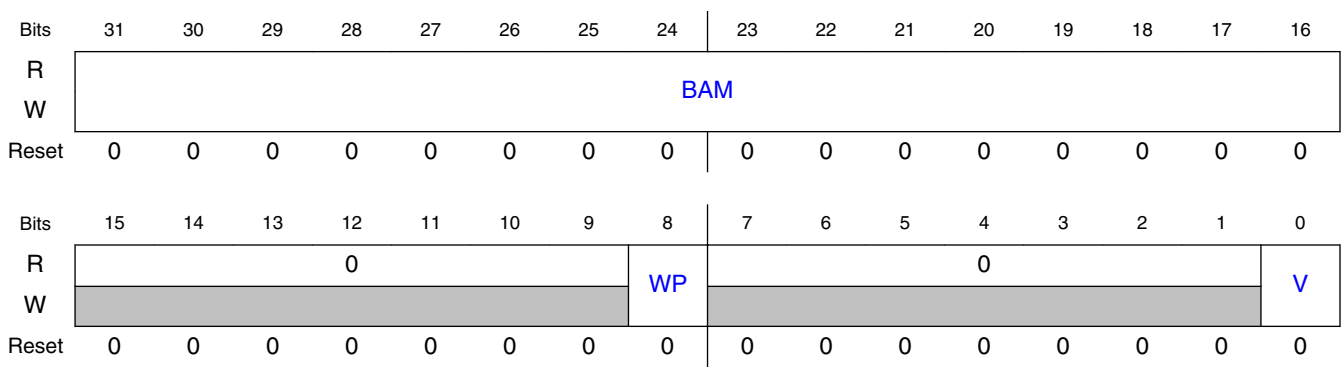
21.4.1.3.1 Offset

Register	Offset
CSMR0	4h
CSMR1	10h
CSMR2	1Ch
CSMR3	28h
CSMR4	34h
CSMR5	40h

21.4.1.3.2 Function

Specifies the address mask and allowable access types for the associated chip-select.

21.4.1.3.3 Diagram



21.4.1.3.4 Fields

Field	Function
31-16 BAM	Base Address Mask Defines the associated chip-select's block size by masking address bits. 0000000000000000b - The corresponding address bit in CSAR is used in the chip-select decode. 0000000000000001b - The corresponding address bit in CSAR is a don’t care in the chip-select decode.
15-9 —	Reserved
8 WP	Write Protect Controls write accesses to the address range in the corresponding CSAR. 0b - Write accesses are allowed. 1b - Write accesses are not allowed. Attempting to write to the range of addresses for which the WP bit is set results in a bus error termination of the internal cycle and no external cycle.

Table continues on the next page...

Field	Function
7-1 —	Reserved
0 V	Valid Specifies whether the corresponding CSAR, CSMR, and CSCR contents are valid. Programmed chip-selects do not assert until the V bit is 1b (except for FB_CS0_B, which acts as the global chip-select). **NOTE:** At reset, FB_CS0_B will fire for any access to the FlexBus memory region. CSMR0[V] must be set as part of the chip select initialization sequence to allow other chip selects to function as programmed. 0b - Chip-select is invalid. 1b - Chip-select is valid.

21.4.1.4 Chip Select Control Register (CSCR0 - CSCR5)

21.4.1.4.1 Offset

Register	Offset
CSCR0	8h
CSCR1	14h
CSCR2	20h
CSCR3	2Ch
CSCR4	38h
CSCR5	44h

21.4.1.4.2 Function

Controls the auto-acknowledge, address setup and hold times, port size, burst capability, and number of wait states for the associated chip select.

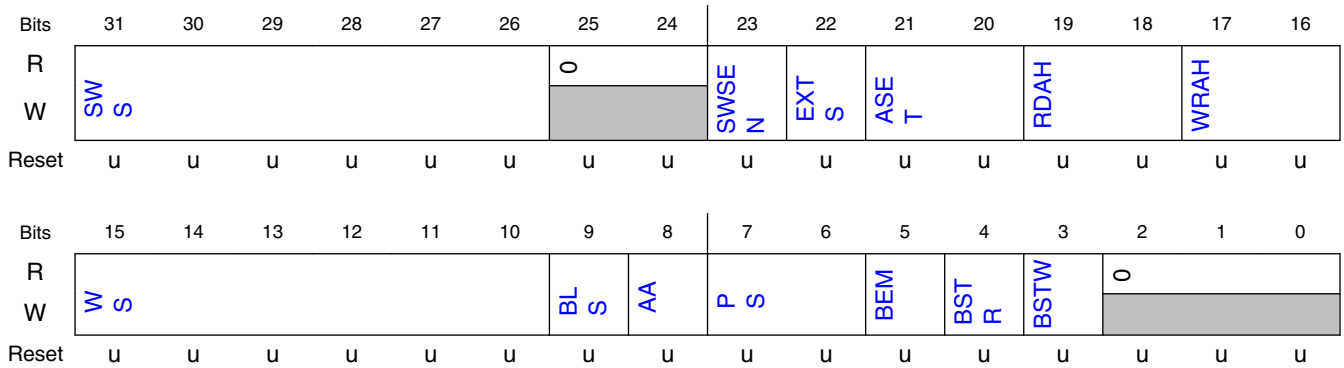
NOTE

To support the global chip-select (FB_CS0_B), the CSCR0 reset values differ from the other CSCRs (CSCR1 - CSCR5 all reset to 0). The reset value of CSCR0 is as follows:

- Bits 31 – 24 are 0b
- Bit 23 – 3 are chip-dependent
- Bits 2 – 0 are 0b

See the chip configuration details for your particular chip for information on the exact CSCR0 reset value.

21.4.1.4.3 Diagram



21.4.1.4.4 Fields

Field	Function
31-26 SWS	Secondary Wait States Used only when the SWSEN bit is 1b. Specifies the number of wait states inserted before an internal transfer acknowledge is generated for a burst transfer (except for the first termination, which is controlled by WS).
25-24 —	Reserved
23 SWSEN	Secondary Wait State Enable 0b - Disabled. A number of wait states (specified by WS) are inserted before an internal transfer acknowledge is generated for all transfers. 1b - Enabled. A number of wait states (specified by SWS) are inserted before an internal transfer acknowledge is generated for burst transfer secondary terminations.
22 EXTS	EXTS Extended Transfer Start/Extended Address Latch Enable Controls how long FB_TS_B/FB_ALE is asserted. 0b - Disabled. FB_TS_B/FB_ALE asserts for one bus clock cycle. 1b - Enabled. FB_TS_B/FB_ALE remains asserted until the first positive clock edge after FB_CSn_B asserts.
21-20 ASET	Address Setup Controls when the chip-select is asserted with respect to assertion of a valid address and attributes. 00b - Assert FB_CSn_B on the first rising clock edge after the address is asserted (default for all but FB_CS0_B). 01b - Assert FB_CSn_B on the second rising clock edge after the address is asserted. 10b - Assert FB_CSn_B on the third rising clock edge after the address is asserted. 11b - Assert FB_CSn_B on the fourth rising clock edge after the address is asserted (default for FB_CS0_B).
19-18 RDAH	Read Address Hold or Deselect Controls the address and attribute hold time after the termination during a read cycle that hits in the associated chip-select's address space.

Table continues on the next page...

Field	Function
	NOTE: • The hold time applies only at the end of a transfer. Therefore, during a burst transfer or a transfer to a port size smaller than the transfer size, the hold time is only added after the last bus cycle. • The number of cycles the address and attributes are held after FB_CSn_B deassertion depends on the value of the AA bit. 00b - When AA is 1b, 1 cycle. When AA is 0b, 0 cycles. 01b - When AA is 1b, 2 cycles. When AA is 0b, 1 cycle. 10b - When AA is 1b, 3 cycles. When AA is 0b, 2 cycles. 11b - When AA is 1b, 4 cycles. When AA is 0b, 3 cycles.
17-16 WRAH	Write Address Hold or Deselect Controls the address, data, and attribute hold time after the termination of a write cycle that hits in the associated chip-select's address space. **NOTE:** The hold time applies only at the end of a transfer. Therefore, during a burst transfer or a transfer to a port size smaller than the transfer size, the hold time is only added after the last bus cycle. 00b - 1 cycle (default for all but FB_CS0_B) 01b - 2 cycles 10b - 3 cycles 11b - 4 cycles (default for FB_CS0_B)
15-10 WS	Wait States Specifies the number of wait states inserted after FlexBus asserts the associated chip-select and before an internal transfer acknowledge is generated (WS = 00h inserts 0 wait states, ..., WS = 3Fh inserts 63 wait states).
9 BLS	Byte-Lane Shift Specifies if data on FB_AD appears left-aligned or right-aligned during the data phase of a FlexBus access. 0b - Not shifted. Data is left-aligned on FB_AD. 1b - Shifted. Data is right-aligned on FB_AD.
8 AA	Auto-Acknowledge Enable Asserts the internal transfer acknowledge for accesses specified by the chip-select address. **NOTE:** If AA is 1b for a corresponding FB_CSn_B and the external system asserts an external FB_TA_B before the wait-state countdown asserts the internal FB_TA_B, the cycle is terminated. Burst cycles increment the address bus between each internal termination. **NOTE:** This field must be 1b if CSPMCR disables FB_TA_B. 0b - Disabled. No internal transfer acknowledge is asserted and the cycle is terminated externally. 1b - Enabled. Internal transfer acknowledge is asserted as specified by WS.
7-6 PS	Port Size Specifies the data port width of the associated chip-select, and determines where data is driven during write cycles and where data is sampled during read cycles. 00b - 32-bit port size. Valid data is sampled and driven on FB_D[31:0]. 01b - 8-bit port size. Valid data is sampled and driven on FB_D[31:24] when BLS is 0b, or FB_D[7:0] when BLS is 1b. 1xb - 16-bit port size. Valid data is sampled and driven on FB_D[31:16] when BLS is 0b, or FB_D[15:0] when BLS is 1b.
5 BEM	Byte-Enable Mode Specifies whether the corresponding FB_BE_B is asserted for read accesses. Certain memories have byte enables that must be asserted during reads and writes. Write 1b to the BEM bit in the relevant CSCR to provide the appropriate mode of byte enable support for these SRAMs. 0b - FB_BE_B is asserted for data write only. 1b - FB_BE_B is asserted for data read and write accesses.

Table continues on the next page...

Field	Function
4 BSTR	Burst-Read Enable Specifies whether burst reads are enabled for memory associated with each chip select. 0b - Disabled. Data exceeding the specified port size is broken into individual, port-sized, non-burst reads. For example, a 32-bit read from an 8-bit port is broken into four 8-bit reads. 1b - Enabled. Enables data burst reads larger than the specified port size, including 32-bit reads from 8- and 16-bit ports, 16-bit reads from 8-bit ports, and line reads from 8-, 16-, and 32-bit ports.
3 BSTW	Burst-Write Enable Specifies whether burst writes are enabled for memory associated with each chip select. 0b - Disabled. Data exceeding the specified port size is broken into individual, port-sized, non-burst writes. For example, a 32-bit write to an 8-bit port takes four byte writes. 1b - Enabled. Enables burst write of data larger than the specified port size, including 32-bit writes to 8- and 16-bit ports, 16-bit writes to 8-bit ports, and line writes to 8-, 16-, and 32-bit ports.
2-0 —	Reserved

21.4.1.5 Chip Select Port Multiplexing Control Register (CSPMCR)

21.4.1.5.1 Offset

Register	Offset
CSPMCR	60h

21.4.1.5.2 Function

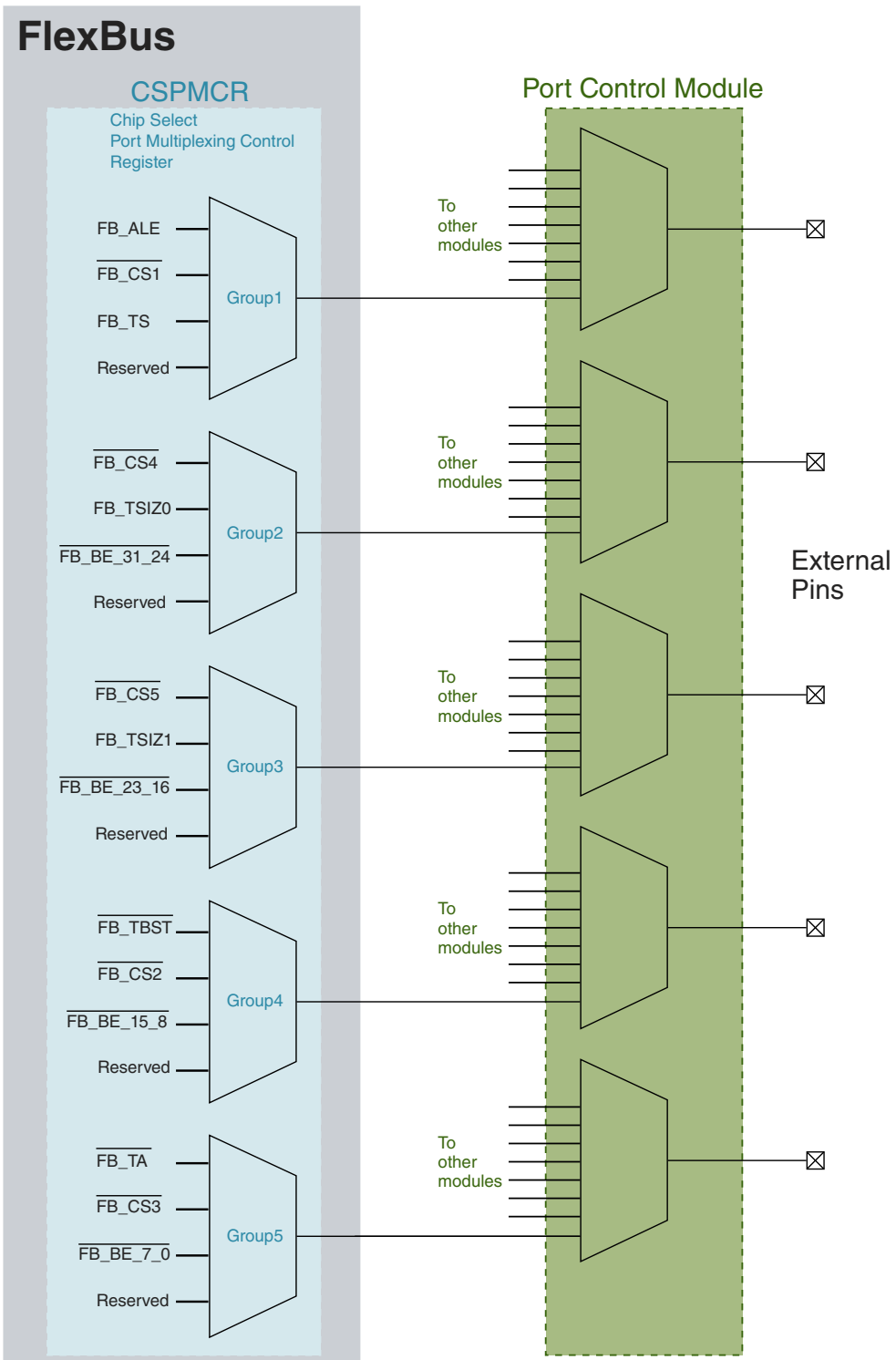
Controls the multiplexing of the FlexBus signals. Use the CSPMCR register (Chip Select Port Multiplexing Control) and port control registers to configure which control signals are available on the external pins.

The multiplexing of the FlexBus address and data signals is controlled by the port control module. However, the multiplexing of some of the FlexBus control signals are controlled by **both** the FlexBus and port control modules. The FlexBus's CSPMCR register configures which FlexBus signals are available from the FlexBus module, and the port control module registers control whether the FlexBus or other module signals are available on the external pin.

NOTE

A bus error occurs when you do any of the following:

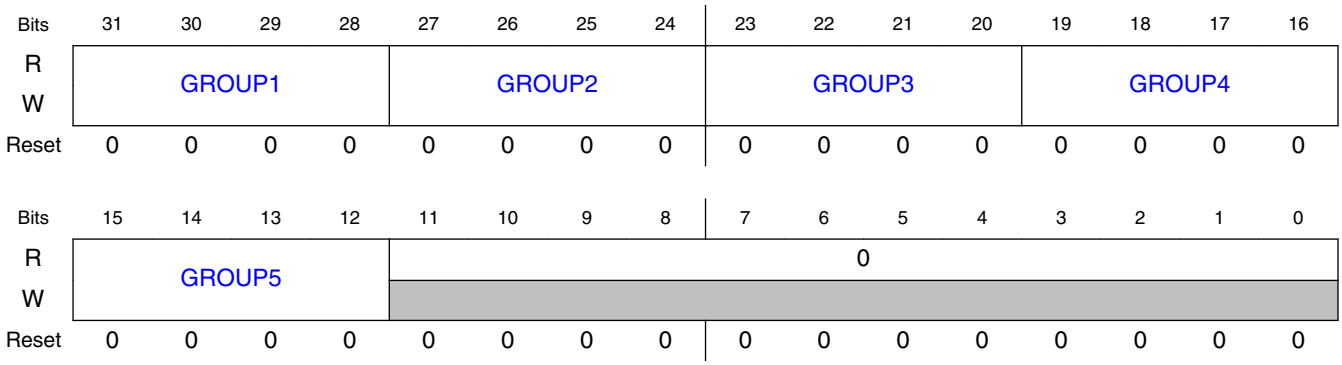
- Write to a reserved address
- Write to a reserved field in this register, or
- Access this register using a size other than 32 bits.



CSPMCR and Port Control Module control the multiplexing of signals to the external pins

Figure 21-3. FlexBus and Port Control module

21.4.1.5.3 Diagram



21.4.1.5.4 Fields

Field	Function
31-28 GROUP1	FlexBus Signal Group 1 Multiplex control Controls the multiplexing of the FB_ALE, FB_CS1_B, and FB_TS_B signals. 0000b - FB_ALE 0001b - FB_CS1_B 0010b - FB_TS_B
27-24 GROUP2	FlexBus Signal Group 2 Multiplex control Controls the multiplexing of the FB_CS4_B, FB_TSIZ0, and FB_BE_31_24_B signals. 0000b - FB_CS4_B 0001b - FB_TSIZ0 0010b - FB_BE_31_24_B
23-20 GROUP3	FlexBus Signal Group 3 Multiplex control Controls the multiplexing of the FB_CS5_B, FB_TSIZ1, and FB_BE_23_16_B signals. 0000b - FB_CS5_B 0001b - FB_TSIZ1 0010b - FB_BE_23_16_B
19-16 GROUP4	FlexBus Signal Group 4 Multiplex control Controls the multiplexing of the FB_TBST_B, FB_CS2_B, and FB_BE_15_8_B signals. 0000b - FB_TBST_B 0001b - FB_CS2_B 0010b - FB_BE_15_8_B
15-12 GROUP5	FlexBus Signal Group 5 Multiplex control Controls the multiplexing of the FB_TA_B, FB_CS3_B, and FB_BE_7_0_B signals. **NOTE:** When GROUP5 is not 0000b, you must write 1b to the CSCR[AA] bit. Otherwise, the bus hangs during a transfer. 0000b - FB_TA_B 0001b - FB_CS3_B. You must also write 1b to CSCR[AA]. 0010b - FB_BE_7_0_B. You must also write 1b to CSCR[AA].
11-0 —	Reserved

21.5 Functional description

21.5.1 Address and data bus configurations

NOTE

Address and data line configurations require signals to actually be available on the pins of your specific device's package. To make sure that the address and data bus signals are available on a specific device's pins, see your device's signal multiplexing table for that device package.

FlexBus supports:

- **Multiplexed address and data lines:**
  - Multiplexed 32-bit address and 32-bit data
  - Multiplexed 32-bit address and 16-bit data (non-multiplexed 16-bit address and 16-bit data)
  - Multiplexed 32-bit address and 8-bit data (non-multiplexed 24-bit address and 8-bit data)

21.5.2 Address comparison

When a bus cycle is routed to FlexBus, FlexBus compares the transfer address to the base address (see CSAR[BA]) and base address mask (see CSMR[BAM]). This table describes how FlexBus decides to assert a chip-select and complete the bus cycle based on the address comparison.

When the transfer address	Then FlexBus
Matches one address register configuration	Asserts the appropriate chip-select, generating a FlexBus bus cycle as defined in the appropriate CSCR. If CSMR[WP] is set and a write access is performed, FlexBus terminates the internal bus cycle with a bus error, does not assert a chip-select, and does not perform an external bus cycle.
Does not match an address register configuration	Terminates the transfer with a bus error response, does not assert a chip-select, and does not perform a FlexBus cycle.
Matches more than one address register configuration	Terminates the transfer with a bus error response, does not assert a chip-select, and does not perform a FlexBus cycle.

21.5.3 Address driven on address bus

Regardless of the external memory's or peripheral's address size,

- For multiplexed address and data busses, FlexBus always drives a 32-bit address on the FB_AD bus.

21.5.4 Connecting address/data lines

The external device must connect its address and data lines as follows:

- Address lines
  - FB_AD from FB_AD0 upward
- Data lines
  - If CSCR[BLS] = 0, FB_AD from FB_AD31 downward
  - If CSCR[BLS] = 1, FB_AD from FB_AD0 upward

21.5.5 Bit ordering

No bit ordering is required when connecting address and data lines to the FB_AD bus. For example, a full 16-bit address/16-bit data device connects its addr15–addr0 to FB_AD16–FB_AD1 and data15–data0 to FB_AD31–FB_AD16. See Data-byte alignment and physical connections for a graphical connection.

21.5.6 Data transfer signals

Data transfers between FlexBus and the external memory or peripheral involve these signals:

- Address/data bus (FB_AD31–FB_AD0 (multiplexed address/data bus))
- Control signals (FB_TS_b/FB_ALE, FB_TA_b, FB_CS_b*n*, FB_OE_b, FB_R/W_b, FB_BE*n*)
- Attribute signals (FB_TBST_b, FB_TSIZ1–FB_TSIZ0)

21.5.7 Signal transitions

These signals change on the rising edge of the FlexBus clock (FB_CLK):

- Address
- Write data

- FB_TS_b/FB_ALE
- active low FB_CSn
- All attribute signals

FlexBus latches the read data on the rising edge of the clock.

21.5.8 Data-byte alignment and physical connections

The device aligns data transfers in FlexBus byte lanes with the number of lanes depending on the data port width.

The following figure shows the byte lanes that external memory or peripheral connects to and the sequential transfers of a 32-bit transfer for the supported port sizes when byte lane shift is disabled. For example, an 8-bit memory connects to the single lane FB_AD31–FB_AD24 (FB_BE_31_24, active low). A 32-bit transfer through this 8-bit port takes 4 transfers, starting with the LSB to the MSB. A 32-bit transfer through a 32-bit port requires one transfer on each 4-byte lane.

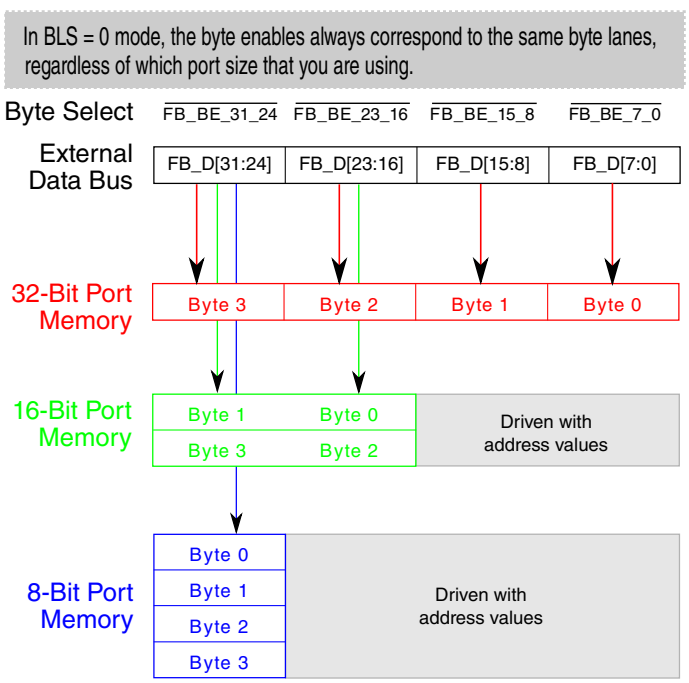


Figure 21-4. Connections for external memory port sizes (CSCRn[BLS] = 0)

The following figure shows the byte lanes that external memory or peripheral connects to and the sequential transfers of a 32-bit transfer for the supported port sizes when byte lane shift is enabled.

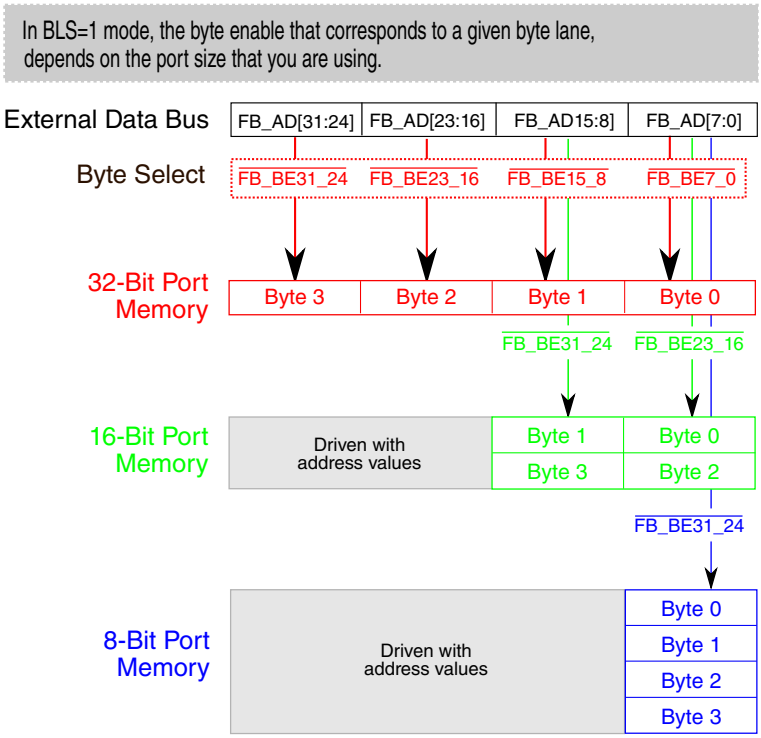


Figure 21-5. Connections for external memory port sizes (CSCRn[BLS] = 1)

21.5.9 Address/data bus multiplexing

FlexBus supports a single 32-bit wide multiplexed address and data bus (FB_AD31–FB_AD0). FlexBus always drives the full 32-bit address on the first clock of a bus cycle. During the data phase, the FB_AD31– FB_AD0 lines used for data are determined by the programmed port size and BLS setting for the corresponding chip-select. FlexBus continues to drive the address on any FB_AD31– FB_AD0 lines not used for data.

21.5.9.1 FlexBus multiplexed operating modes for CSCRn[BLS]=0

This table shows the supported combinations of address and data bus widths when CSCRn[BLS] is 0b.

Port size and phase		FB_AD			
		31–24	23–16	15–8	7–0
32-bit	Address phase	Address			
	Data phase	Data			
16-bit	Address phase	Address			
	Data phase	Data		Address	

Table continues on the next page...

Port size and phase		FB_AD			
		31–24	23–16	15–8	7–0
8-bit	Address phase	Address			
	Data phase	Data	Address		

21.5.9.2 FlexBus multiplexed operating modes for CSCRn[BLS]=1

This table shows the supported combinations of address and data bus widths when CSCRn[BLS] is 1b.

Port size and phase		FB_AD			
		31–24	23–16	15–8	7–0
32-bit	Address phase	Address			
	Data phase	Data			
16-bit	Address phase	Address			
	Data phase	Address		Data	
8-bit	Address phase	Address			
	Data phase	Address			Data

21.5.10 Data transfer states

Basic data transfers occur in 4 clocks or states. (See Basic Read Bus Cycle and Basic Write Bus Cycle for examples of basic data transfers.) The FlexBus state machine controls the data-transfer operation. This figure shows the state-transition diagram for basic read and write cycles.

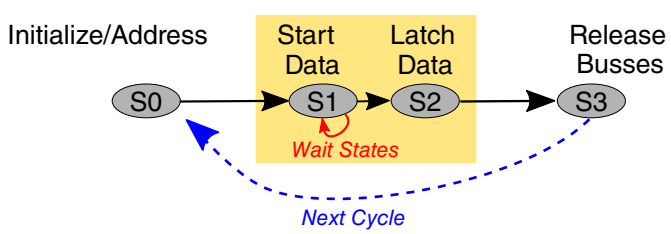


Figure 21-6. Data transfer state transitions

State		Cycle	Description
Initialize/ address	S0	All	The read or write cycle is initiated. On the rising clock edge, FlexBus: • Places a valid address on FB_ADn

Table continues on the next page...

State		Cycle	Description
			• Asserts FB_TS_b/FB_ALE • Drives FB_R/W_b high for a read and low for a write
Start data phase	S1	All	FlexBus: • Negates FB_TS_b/FB_ALE on the rising edge of FB_CLK • Asserts FB_CSn (active low) • Drives the data on FB_AD31– FB_AD*X* for writes • Tristates FB_AD31– FB_AD*X* for reads • Continues to drive the address on FB_AD pins that are unused for data • If the external memory or peripheral asserts FB_TA_b, then the process moves to S2. • If FB_TA_b is not asserted internally or externally, then S1 repeats.
		Read	The external memory or peripheral drives the data before the next rising edge of FB_CLK (the rising edge that begins S2) with FB_TA_b asserted.
Latch data	S2	All	• For internal termination, FlexBus negates FB_CSn (active low) and the transfer is complete. • For external termination, the external memory or peripheral negates FB_TA_b, then FlexBus negates FB_CSn (active low) after the rising edge of FB_CLK at the end of S2.
		Read	FlexBus latches the data on the rising clock edge entering S2. The external memory or peripheral can stop driving the data after this edge, or continue to drive the data until the end of S3 or drive the data through any additional address hold cycles.
Release busses	S3	All	FlexBus invalidates the address, data, and FB_R/W_b on the rising edge of FB_CLK at the beginning of S3, terminating the transfer.

21.5.11 FlexBus Timing Examples

Note

The timing diagrams in this section use signal names that may not be included on your particular device, so ignore these signals.

Note

Throughout this section:

- FB_D[*X*] indicates a 32-, 16-, or 8-bit wide data bus
- FB_A[*Y*] indicates an address bus that can be 32, 24, or 16 bits wide.

21.5.11.1 Basic Read Bus Cycle

During a read cycle, the MCU receives data from memory or a peripheral device.

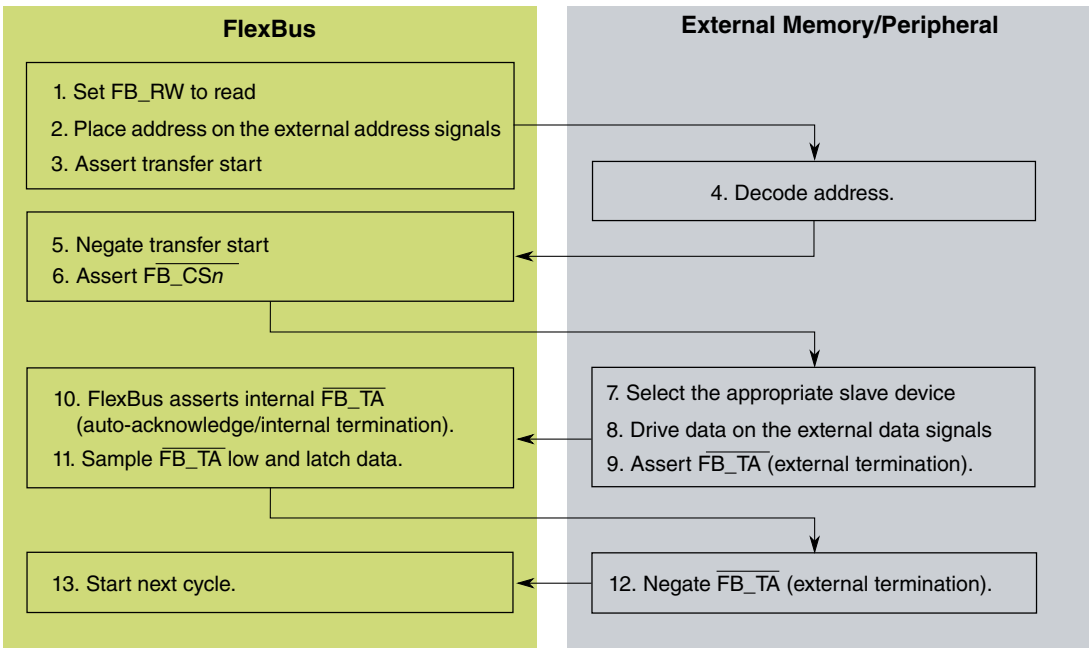


Figure 21-7. Read Cycle Flowchart

Note

FB_TA_b (active low) does not have to be driven by the external device for internally-terminated bus cycles.

Note

The processor drives the data lines during the first clock cycle of the transfer with the full 32-bit address. This may be ignored by standard connected devices using non-multiplexed address and data buses. However, some applications may find this feature beneficial.

The address and data busses are muxed between the FlexBus and another module. At the end of the read bus cycles the address signals are indeterminate.

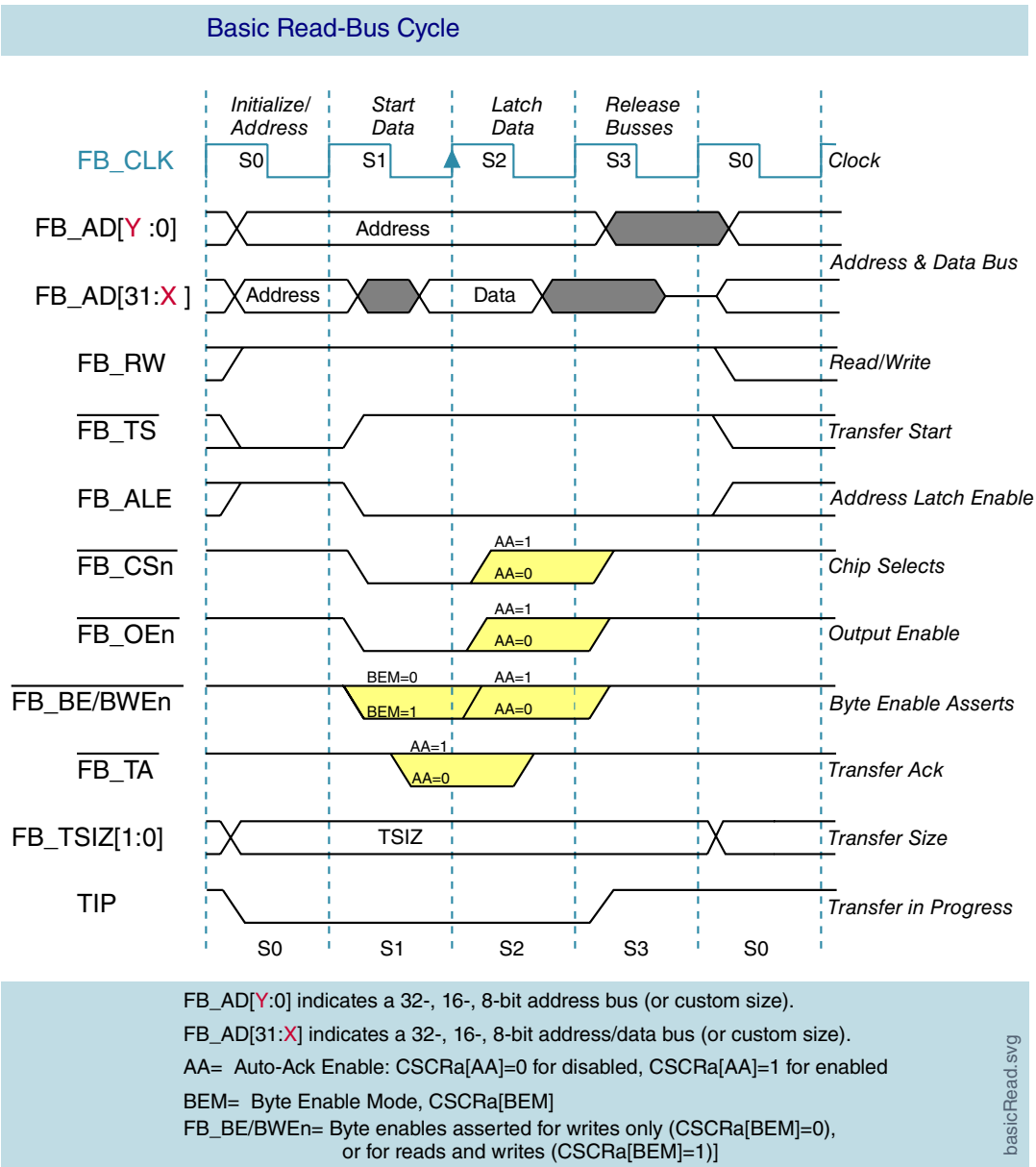


Figure 21-8. Basic Read Bus Cycle

21.5.11.2 Basic Write Bus Cycle

During a write cycle, the device sends data to memory or to a peripheral device.

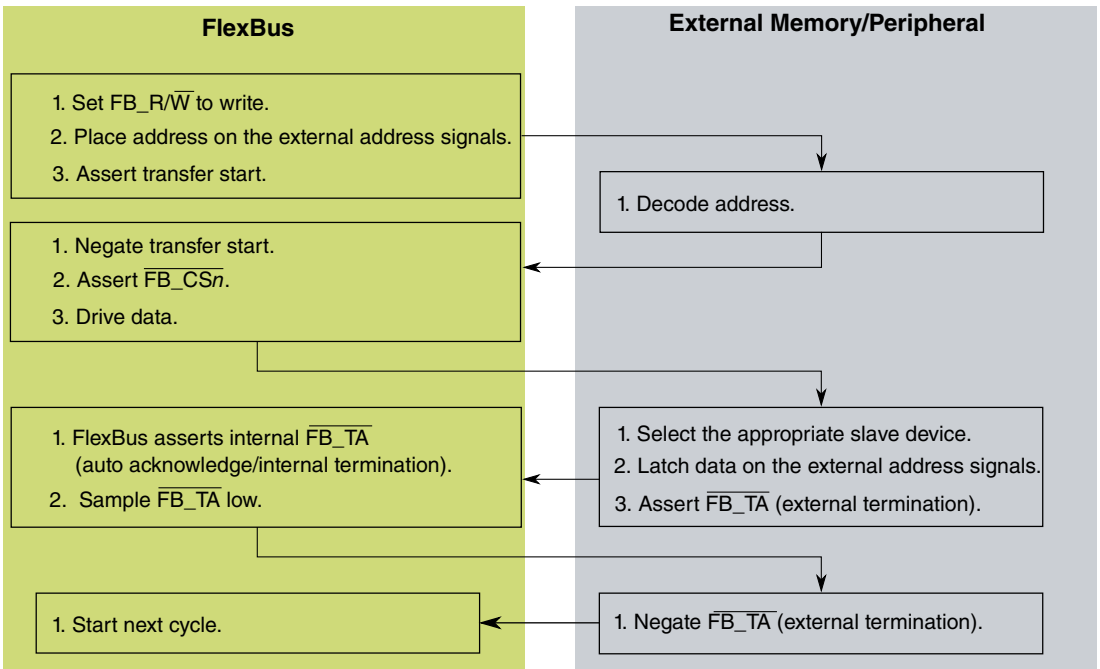


Figure 21-9. Write Cycle Flowchart

Note

The address and data busses are muxed between the FlexBus and another module. At the end of the write bus cycles, the address signals are indeterminate.

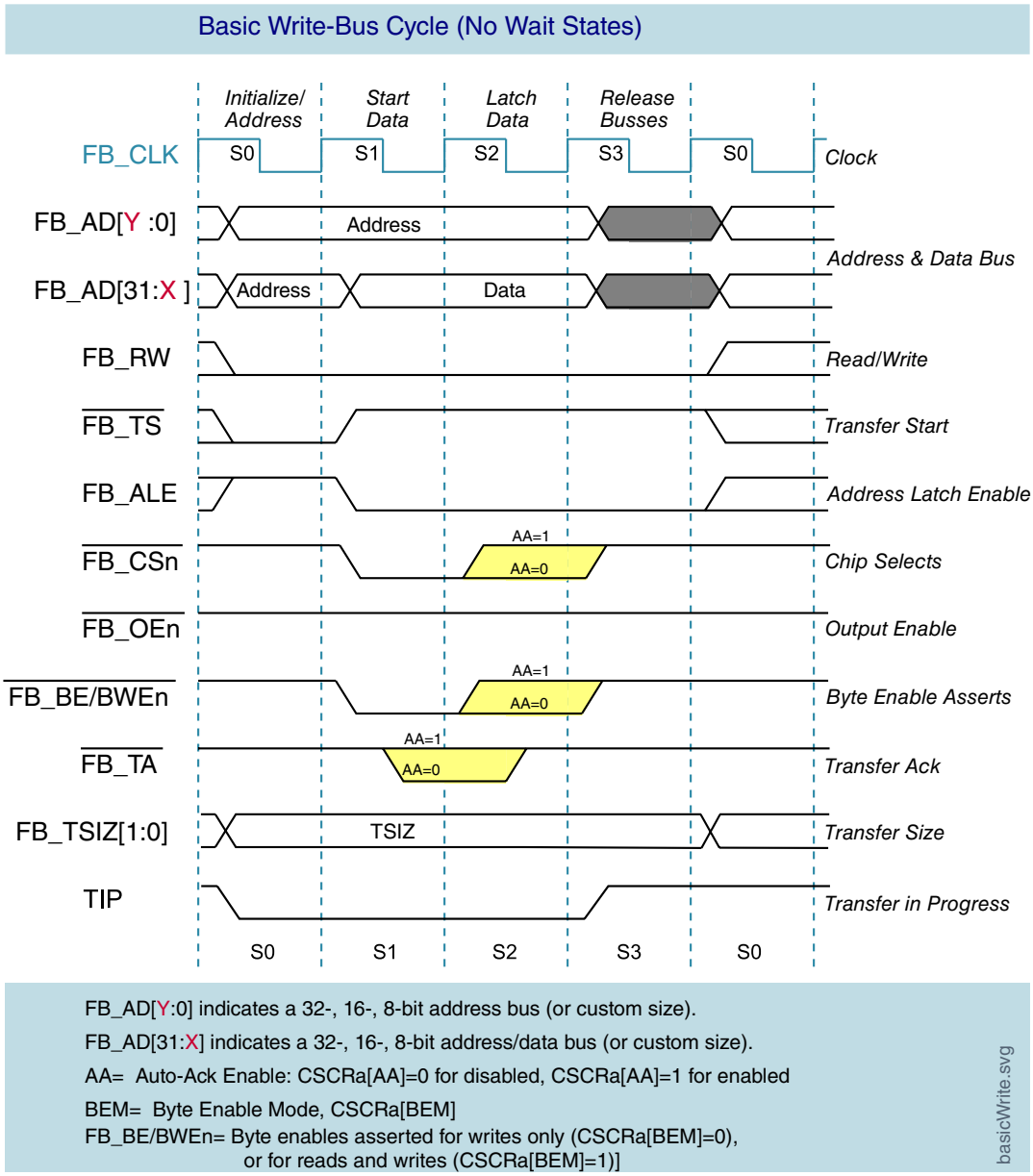


Figure 21-10. Basic Write Bus Cycle

21.5.11.3 Bus Cycle Sizing

This section shows timing diagrams for various port size scenarios.

NOTE

All examples in this "Bus Cycle Sizing" section assume BLS=0.

21.5.11.3.1 Bus Cycle Sizing—Byte Transfer, 8-bit Device, No Wait States

The next figure shows the basic byte read transfer to an 8-bit device with no wait states:

- For multiplexed address and data busses, the address is driven on the full FB_AD[31:0] bus in the 1st clock; the device tristates FB_AD[31:24] on the 2nd clock and continues to drive address on FB_AD[23:0] throughout the bus cycle.
- The external device returns the read data on FB_AD[31:24] and may tristate the data line or continue driving the data 1 clock after FB_TA_b (active low) is sampled asserted.

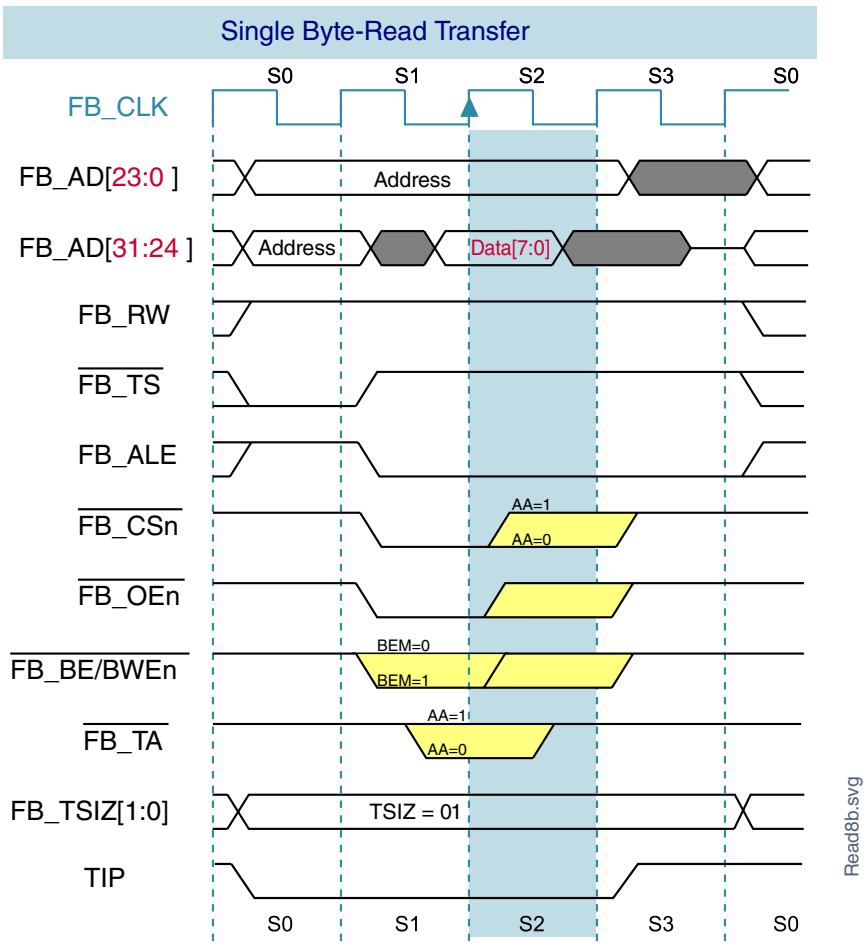


Figure 21-11. Single Byte-Read Transfer

The next figure shows the similar configuration for a write transfer. The data is driven from the 2nd clock on FB_AD[31:24].

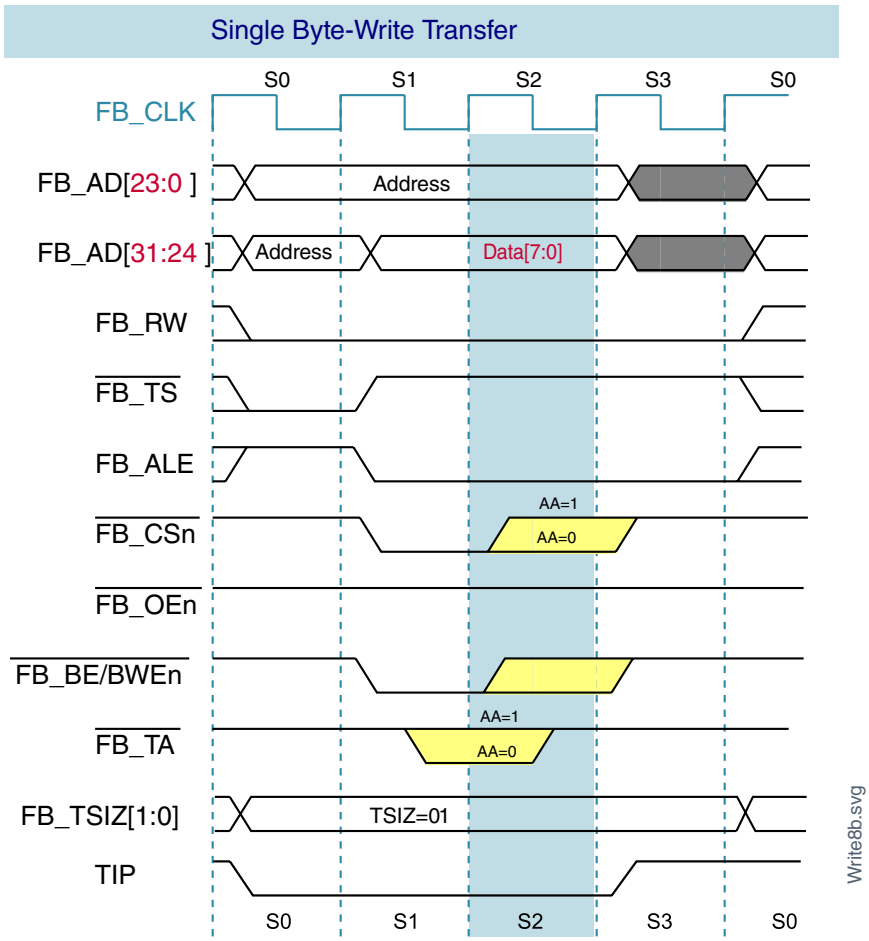


Figure 21-12. Single Byte-Write Transfer

21.5.11.3.2 Bus Cycle Sizing—Word Transfer, 16-bit Device, No Wait States

The next figure shows a basic word read transfer to a 16-bit device with no wait states.

- For multiplexed address and data busses, the address is driven on the full FB_AD[31:0] bus in the first clock; the device tristates FB_AD[31:16] on the second clock and continues to drive address on FB_AD[15:0] throughout the bus cycle.
- The external device returns the read data on FB_AD[31:16] and may tristate the data line or continue driving the data one clock after FB_TA_b (active low) is sampled asserted.

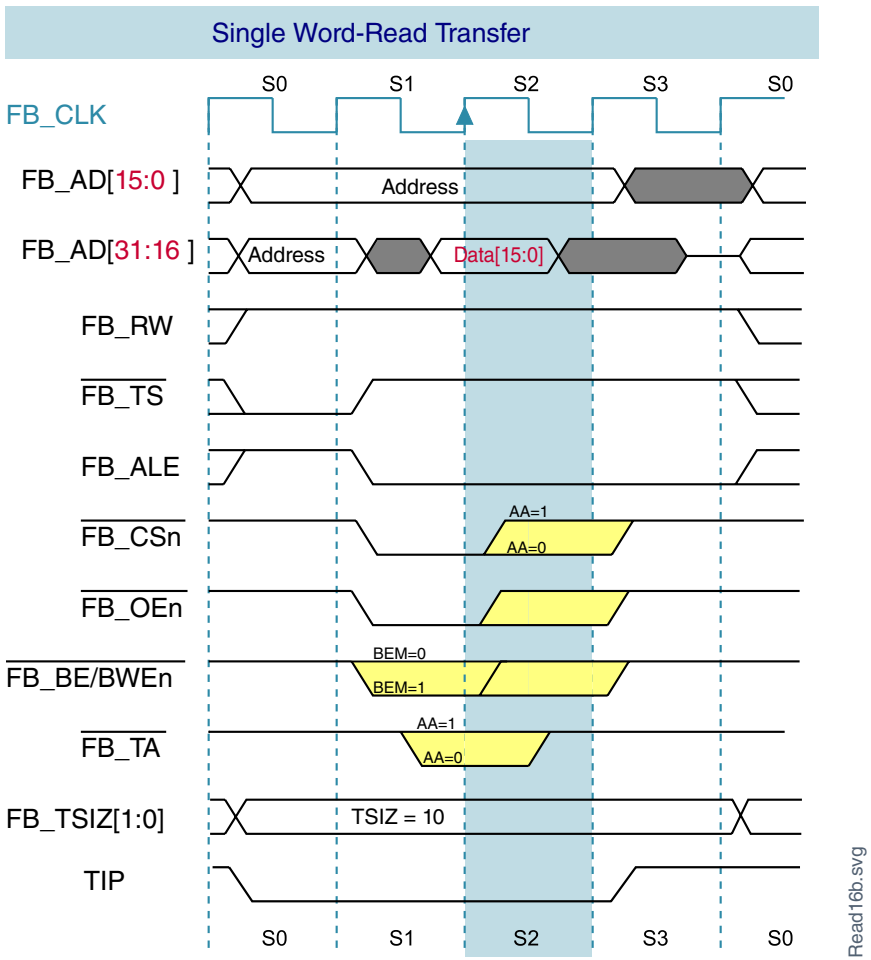


Figure 21-13. Single Word-Read Transfer

The next figure shows a basic word write transfer. The data is driven from the second clock on FB_AD[31:16].

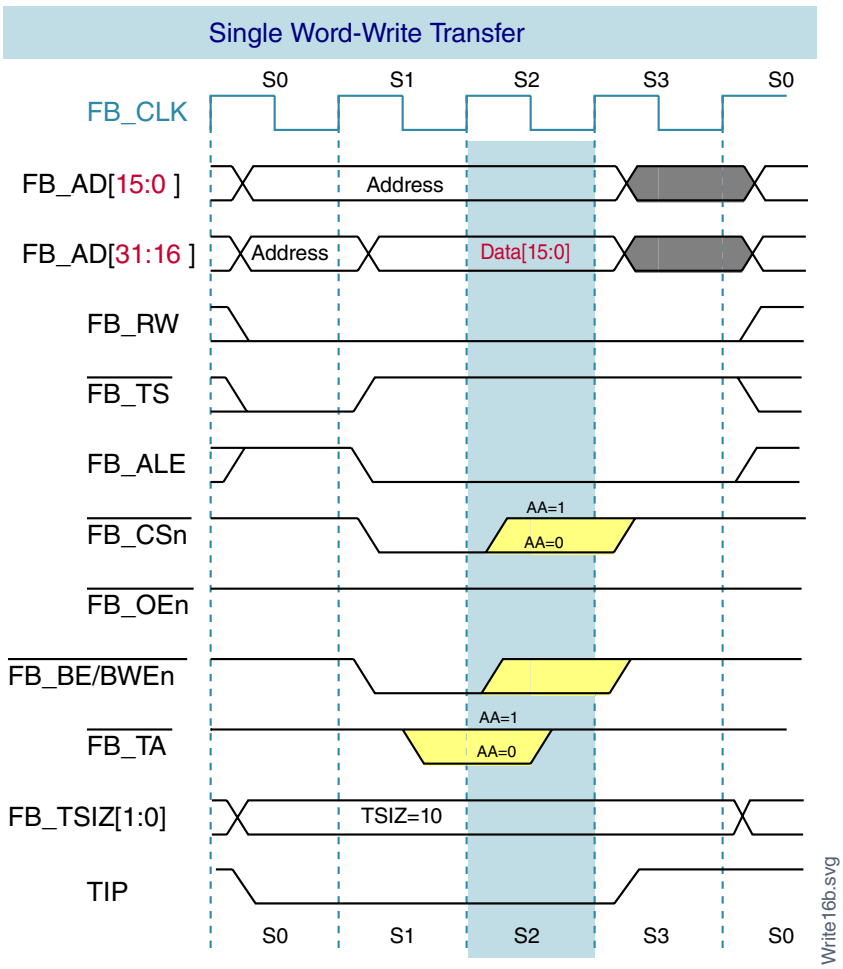


Figure 21-14. Single Word-Write Transfer

21.5.11.3.3 Bus Cycle Sizing—Longword Transfer, 32-bit Device, No Wait States

The next figure shows a longword read from a 32-bit device.

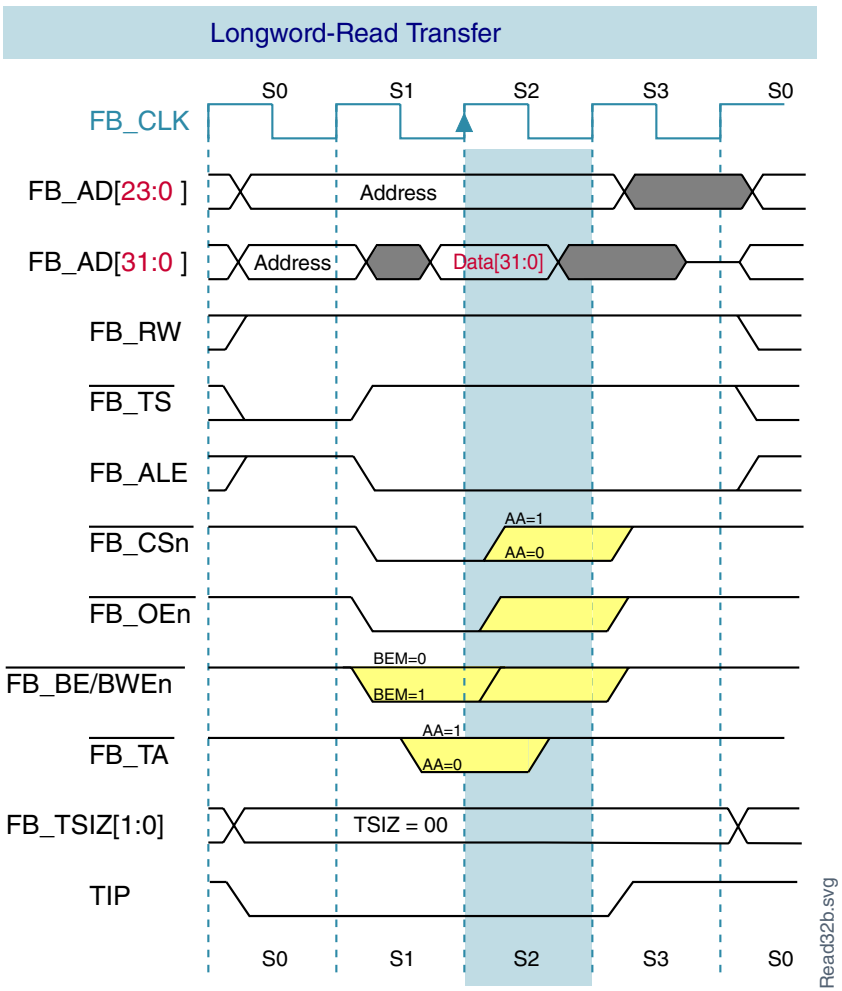


Figure 21-15. Longword-Read Transfer

The next figure shows a longword write to a 32-bit device.

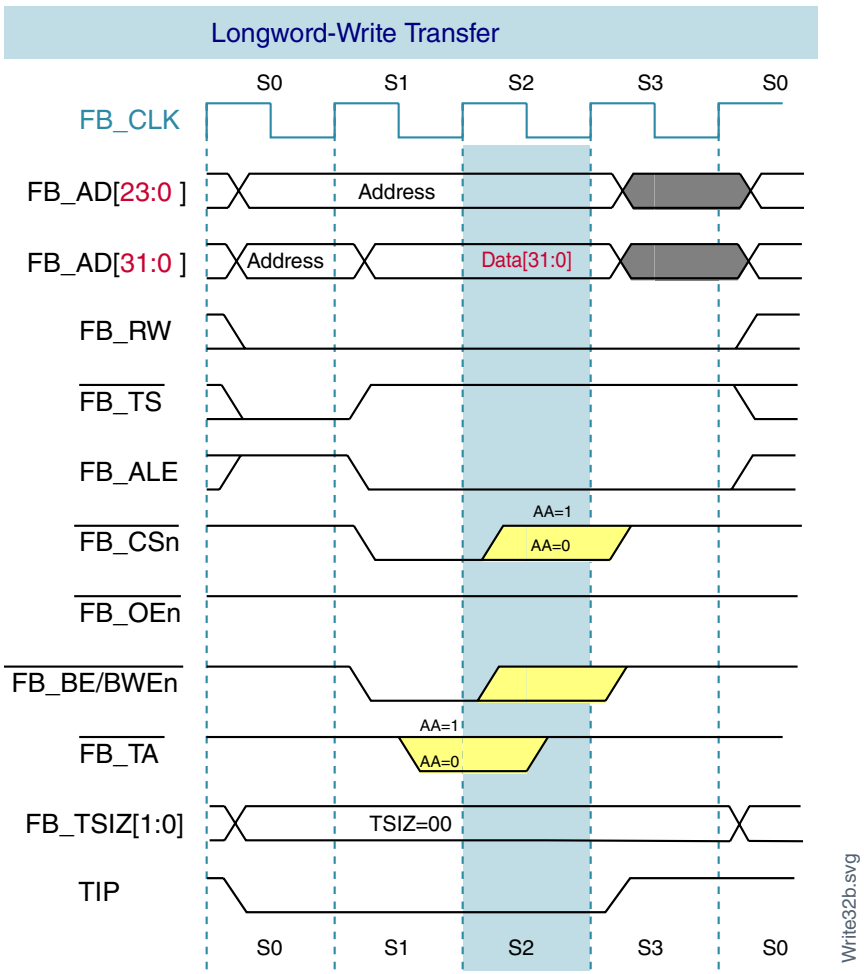


Figure 21-16. Longword-Write Transfer

21.5.11.4 Timing Variations

The FlexBus module has several features that can change the timing characteristics of a basic read- or write-bus cycle to provide additional address setup, address hold, and time for a device to provide or latch data.

21.5.11.4.1 Wait States

Wait states can be inserted before each beat of a transfer by programming the CSCR*n* registers. Wait states can give the peripheral or memory more time to return read data or sample write data.

The following figures show the basic read and write bus cycles (also shown in Basic Read Bus Cycle and Bus Cycle Sizing—Word Transfer, 16-bit Device, No Wait States) with the default of no wait states respectively.

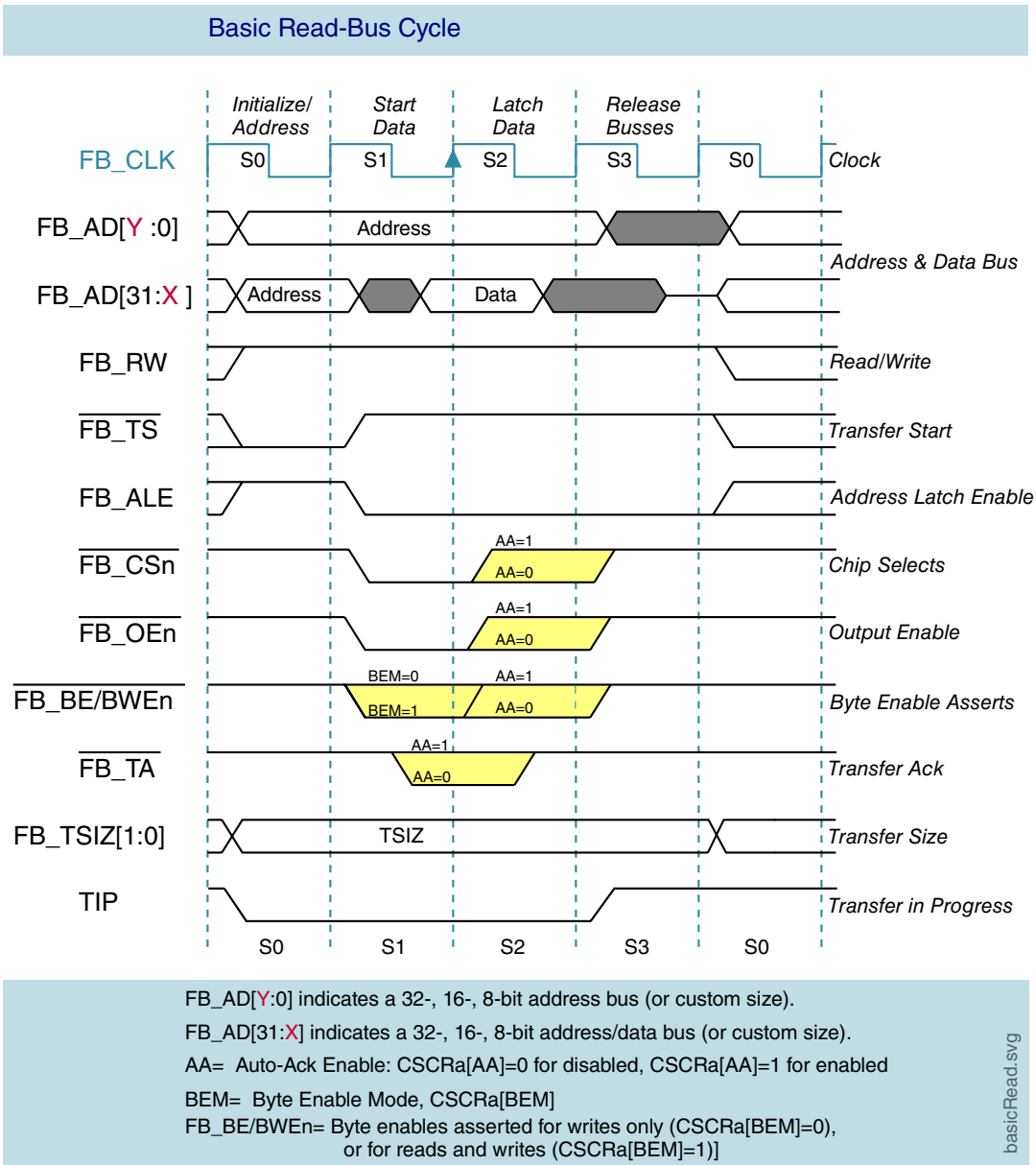


Figure 21-17. Basic Read-Bus Cycle (No Wait States)

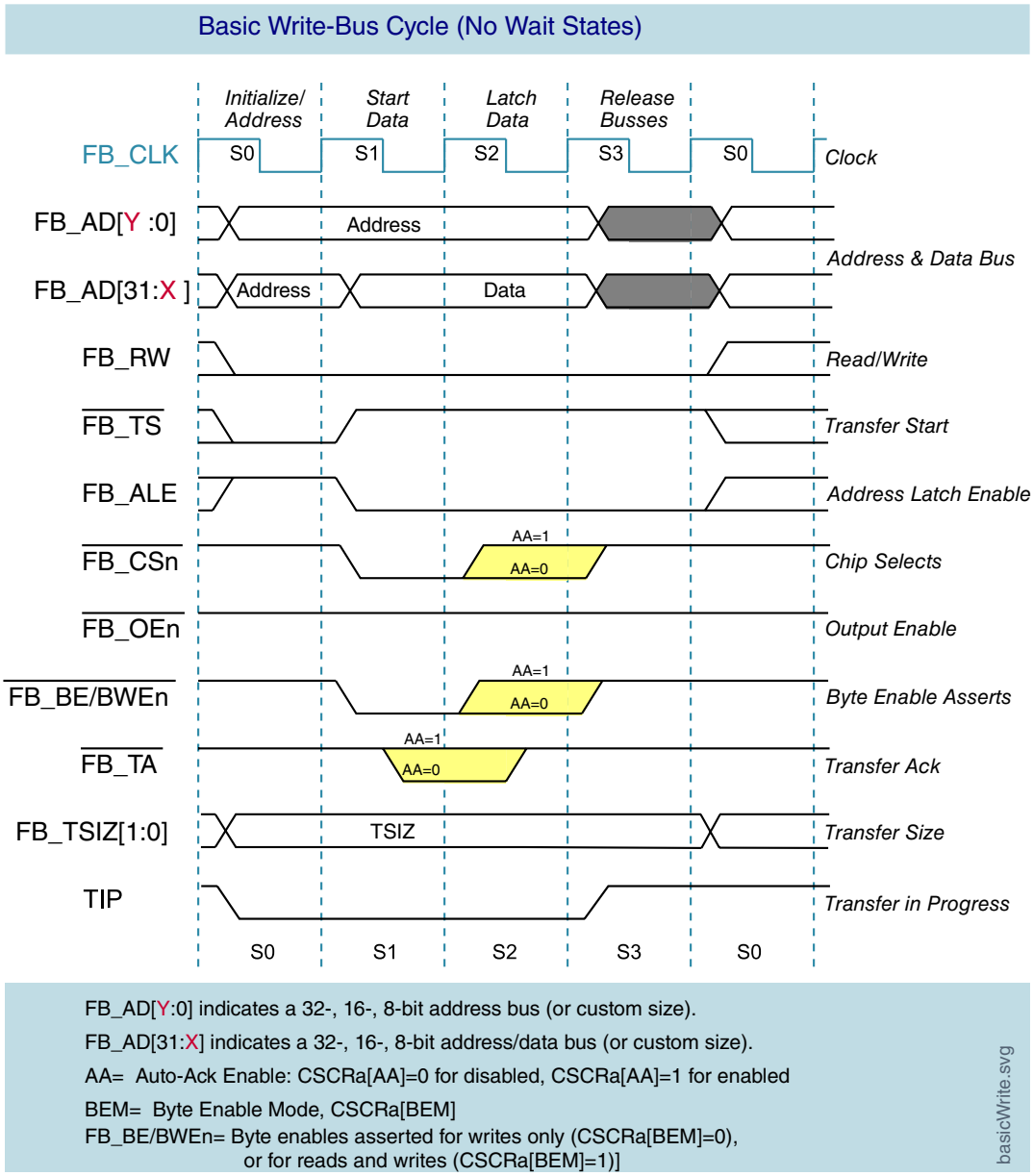
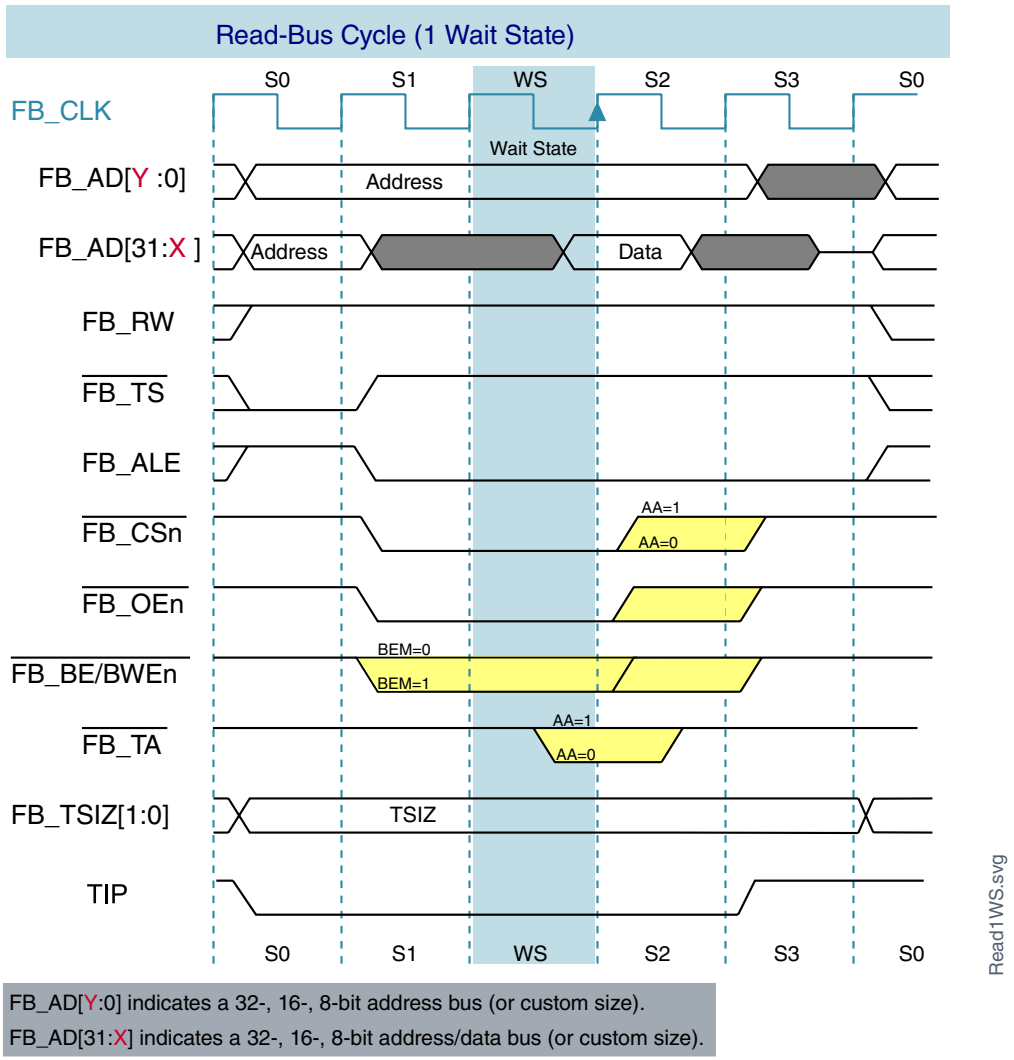


Figure 21-18. Basic Write-Bus Cycle (No Wait States)

If wait states are used, the S1 state repeats continuously until the chip-select auto-acknowledge unit asserts internal transfer acknowledge or the external FB_TA_b is recognized as asserted. The following figures show a read and write cycle with one wait state respectively.



FB_AD[Y:0] indicates a 32-, 16-, 8-bit address bus (or custom size).
FB_AD[31:X] indicates a 32-, 16-, 8-bit address/data bus (or custom size).

Figure 21-19. Read-Bus Cycle (One Wait State)

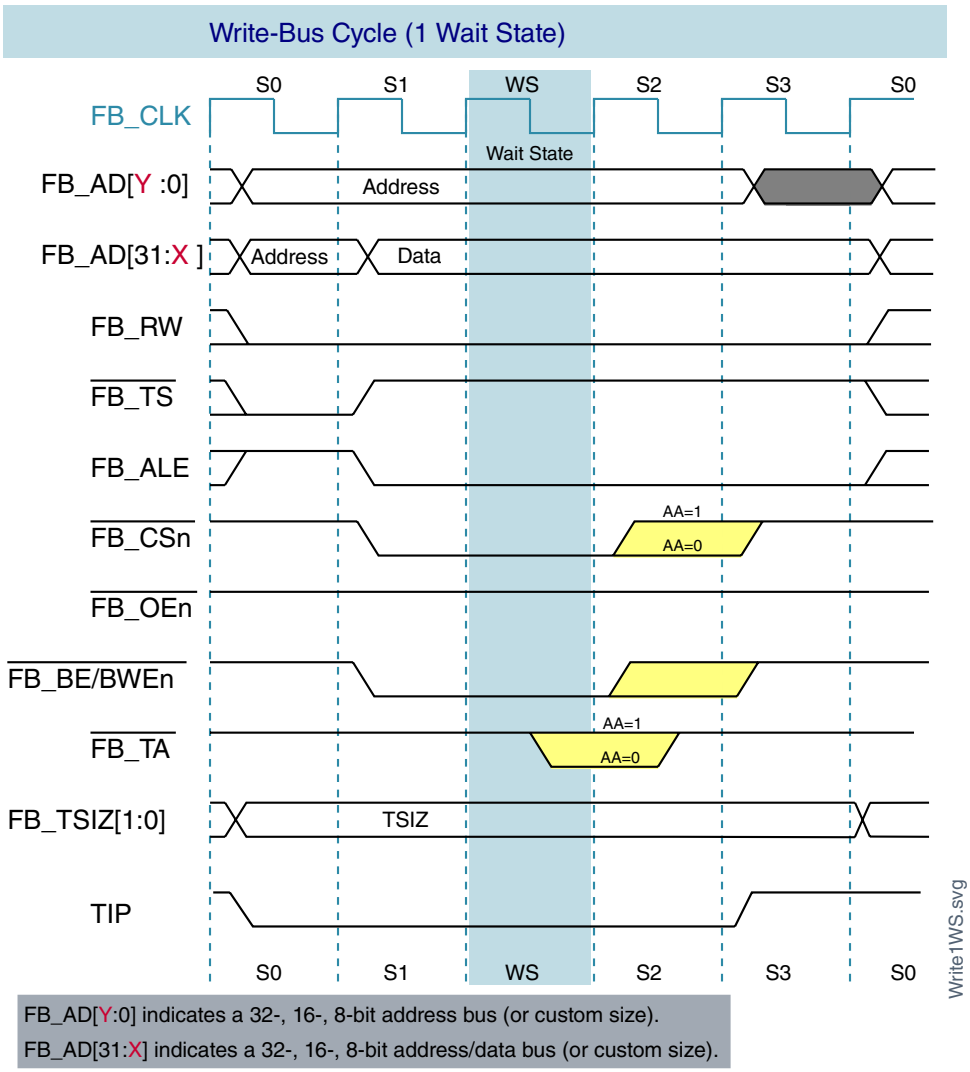


Figure 21-20. Write-Bus Cycle (One Wait State)

21.5.11.4.2 Address Setup and Hold

The timing of the assertion and negation of the chip selects, byte selects, and output enable can be programmed on a chip-select basis. Each chip-select can be programmed to assert one to four clocks after transfer start/address-latch enable (FB_TS_b/FB_ALE) is asserted. The following figures show read- and write-bus cycles with two clocks of address setup respectively.

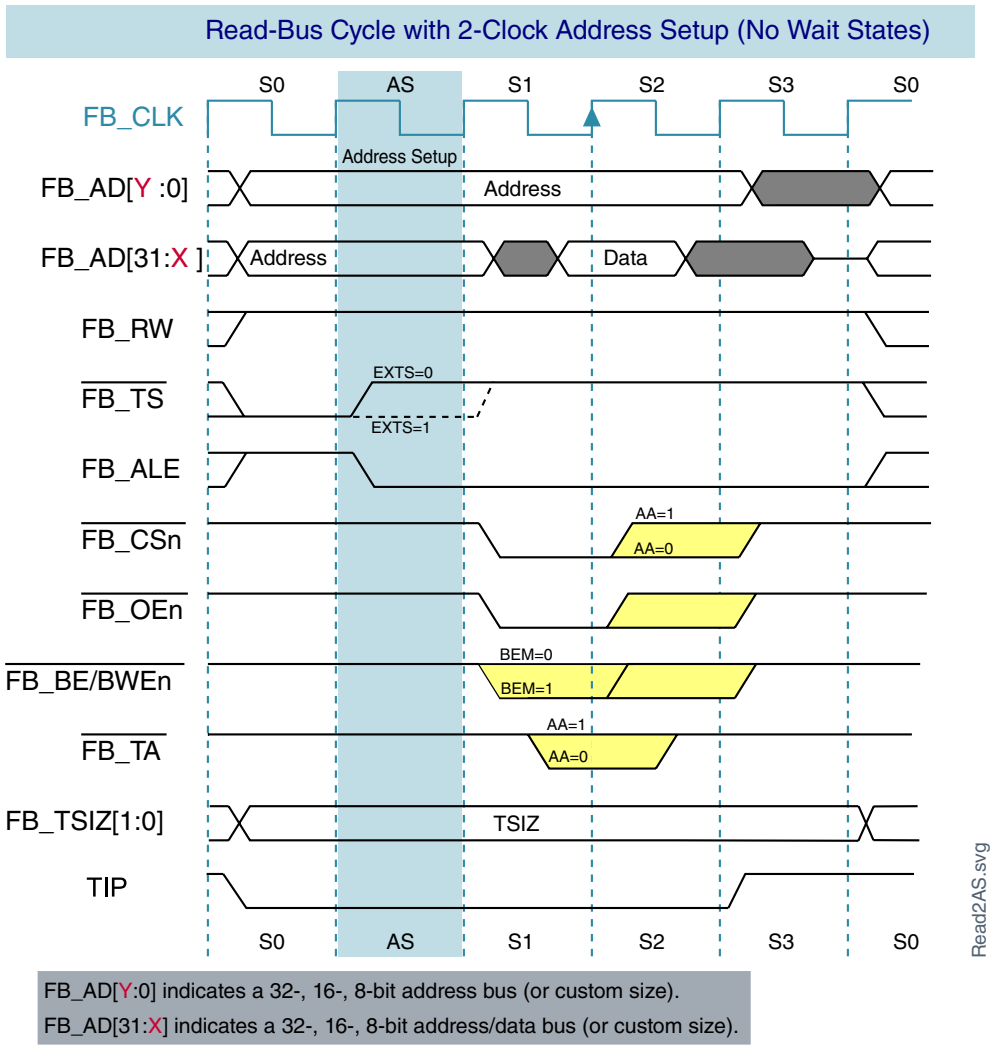


Figure 21-21. Read-Bus Cycle with Two-Clock Address Setup (No Wait States)

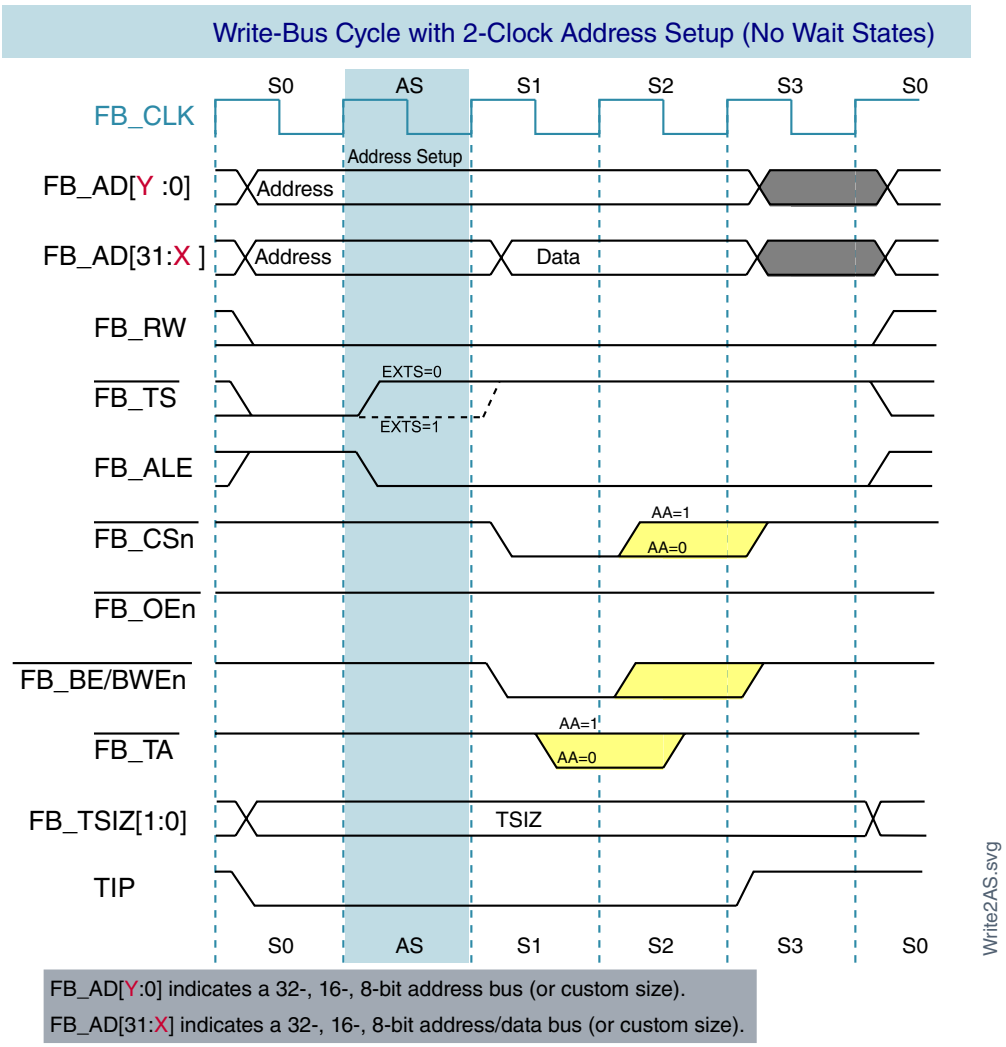


Figure 21-22. Write-Bus Cycle with Two Clock Address Setup (No Wait States)

In addition to address setup, a programmable address hold option for each chip select exists. Address and attributes can be held one to four clocks after chip-select, byte-selects, and output-enable negate. The following figures show read and write bus cycles with two clocks of address hold respectively.

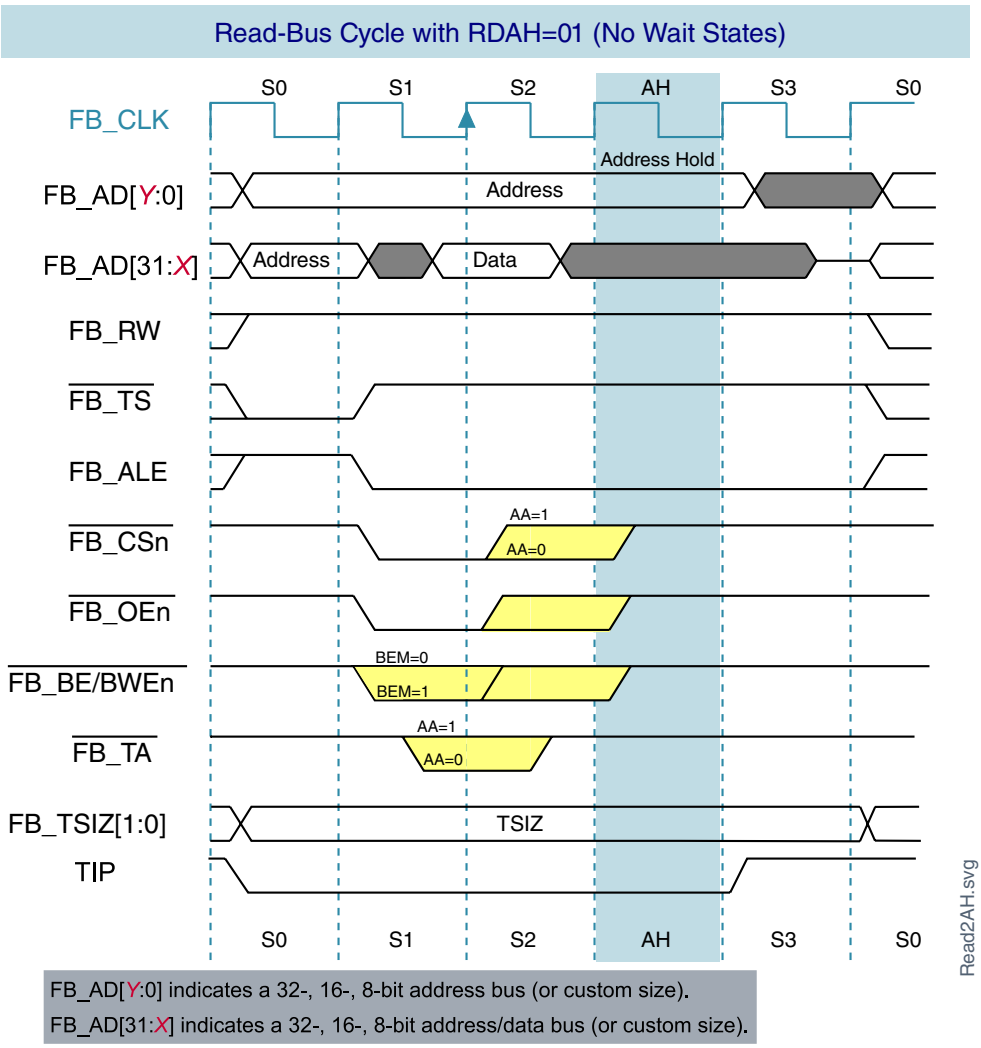


Figure 21-23. Read Cycle with Two-Clock Address Hold (No Wait States)

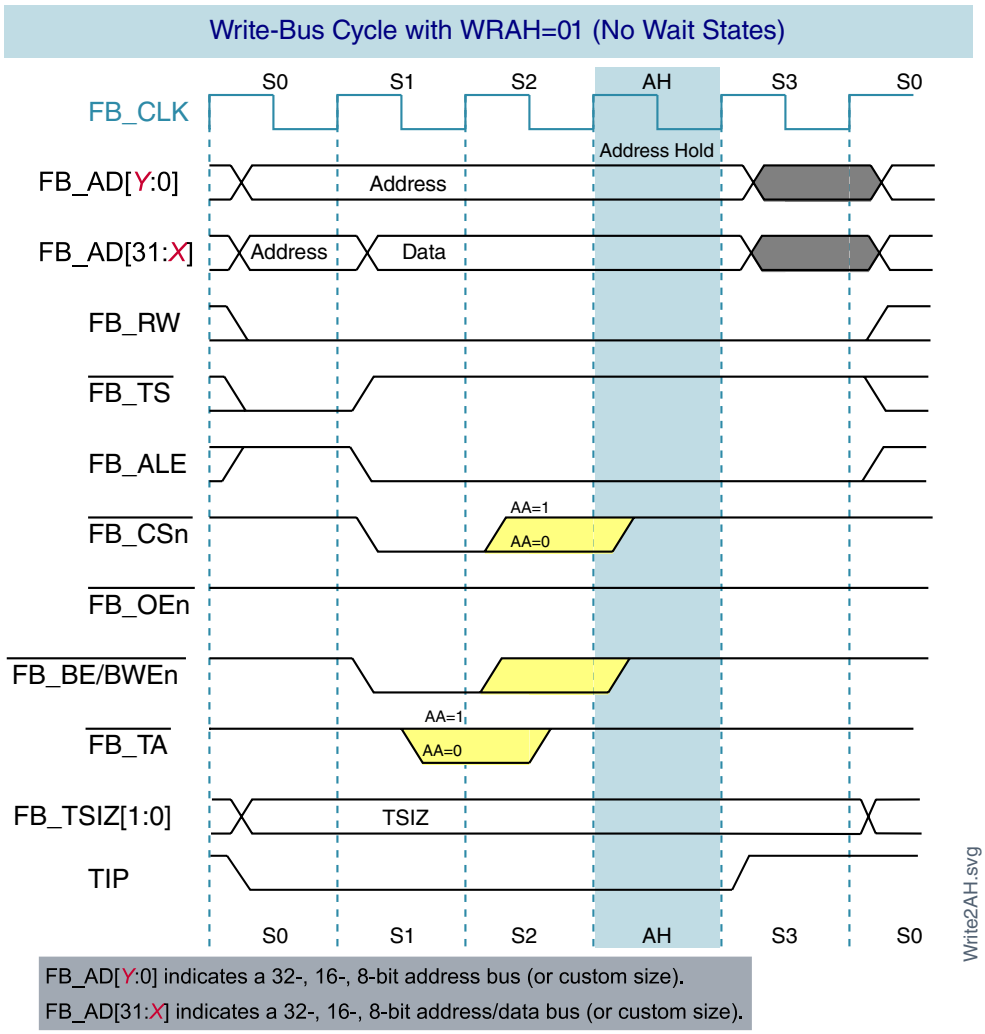


Figure 21-24. Write Cycle with Two-Clock Address Hold (No Wait States)

The following figure shows a bus cycle using address setup, wait states, and address hold.

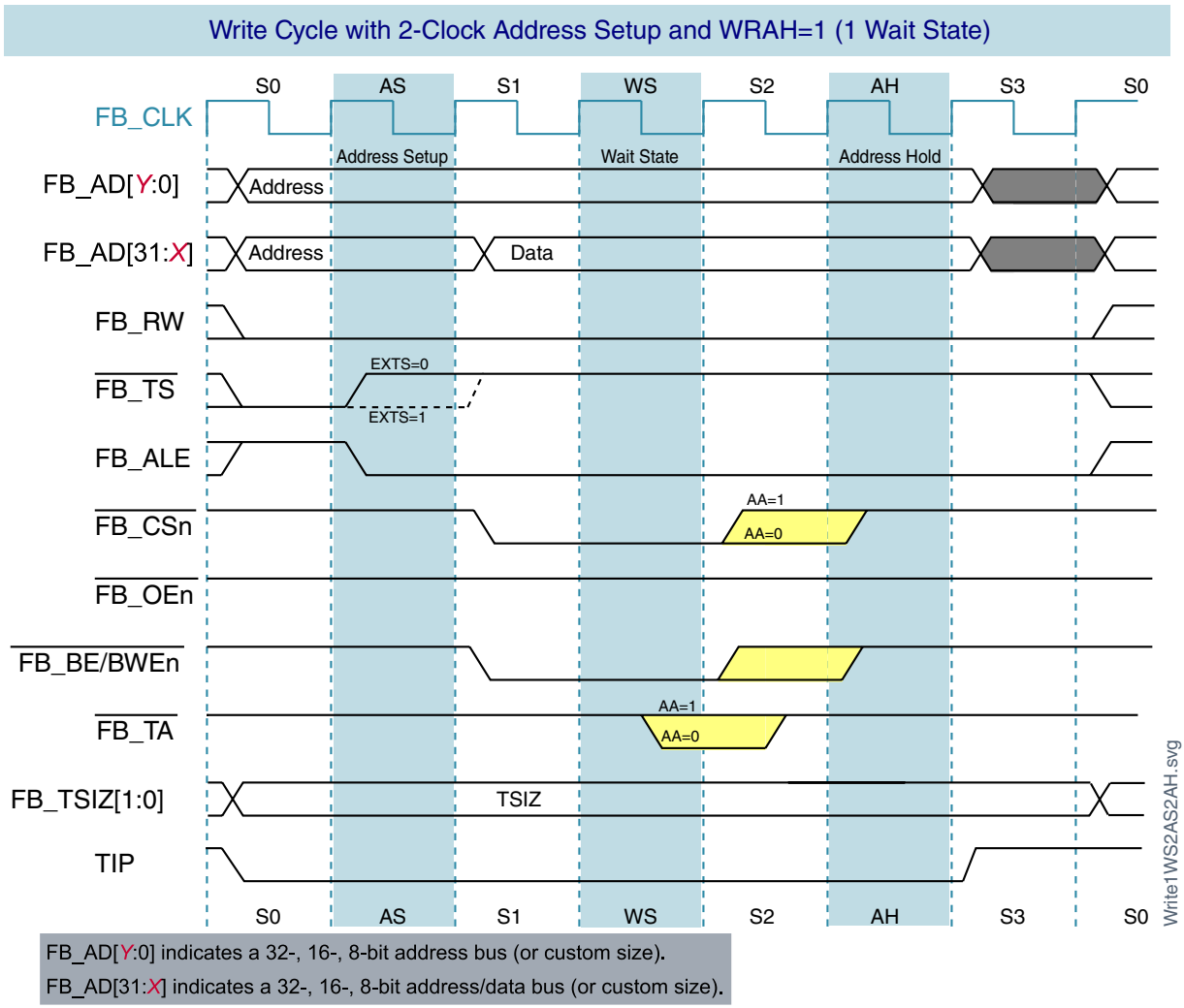


Figure 21-25. Write Cycle with Two-Clock Address Setup and Two-Clock Hold (One Wait State)

21.5.12 Burst cycles

The chip can be programmed to initiate burst cycles if its transfer size exceeds the port size of the selected destination. The initiation of a burst cycle is encoded on the transfer size pins (FB_TSIZ[1:0]). For burst transfers to smaller port sizes, FB_TSIZ[1:0] indicates the size of the entire transfer. For example, with bursting enabled, a 16-bit transfer to an 8-bit port takes two beats (two byte-sized transfers), for which FB_TSIZ[1:0] equals 10b throughout. A 32-bit transfer to an 8-bit port takes four beats (four byte-sized transfers), for which FB_TSIZ[1:0] equals 00b throughout.

21.5.12.1 Enabling and inhibiting burst

The CSCRn registers enable bursting for reads, writes, or both.

Memory spaces can be declared burst-inhibited for reads and writes by writing 0b to the appropriate CSCRn[BSTR] and CSCRn[BSTW] fields.

21.5.12.2 Transfer size and port size translation

With bursting disabled, any transfer larger than the port size breaks into multiple individual transfers (e.g. <Addr><Data><Addr+1><Data><Addr+2><Data>). With bursting enabled, any transfer larger than the port size results in a burst cycle of multiple beats (e.g. <Addr><Data><Data><Data>). The following table shows the result of such transfer translations.

Port size PS[1:0]	Transfer size FB_TSIZ[1:0]	Burst-inhibited: Number of transfers Burst enabled: Number of beats
01b (8 bit)	10b (16 bits)	2
	00b (32 bits)	4
	11b (16 bytes)	16
1Xb (16 bit)	00b (32 bits)	2
	11b (16 bytes)	8
00b (32 bit)	11b (line)	4

The FlexBus can support X-1-1-1 burst cycles to maximize system performance, where X is the primary number of wait states (max 63). Delaying termination of the cycle can add wait states. If internal termination is used, different wait state counters can be used for the first access and the following beats.

21.5.12.3 32-bit-Read burst from 8-Bit port 2-1-1-1 (no wait states)

The following figure shows a 32-bit read to an 8-bit external chip programmed for burst enable. The transfer results in a 4-beat burst and the data is driven on FB_AD[31:24]. The transfer size is driven at 32-bit (00b) throughout the bus cycle.

Note

- In **non-multiplexed address/data mode:** the address on FB_A increments only during internally-terminated burst

cycles. The first address is driven throughout the entire burst for externally-terminated cycles.

- In **multiplexed address/data mode:** the address is driven on FB_AD only during the first cycle for all terminated cycles. In other words, the lower address lines only increment during internally-terminated burst cycles (assuming that the lower address lines are being driven with the address during the data phase).

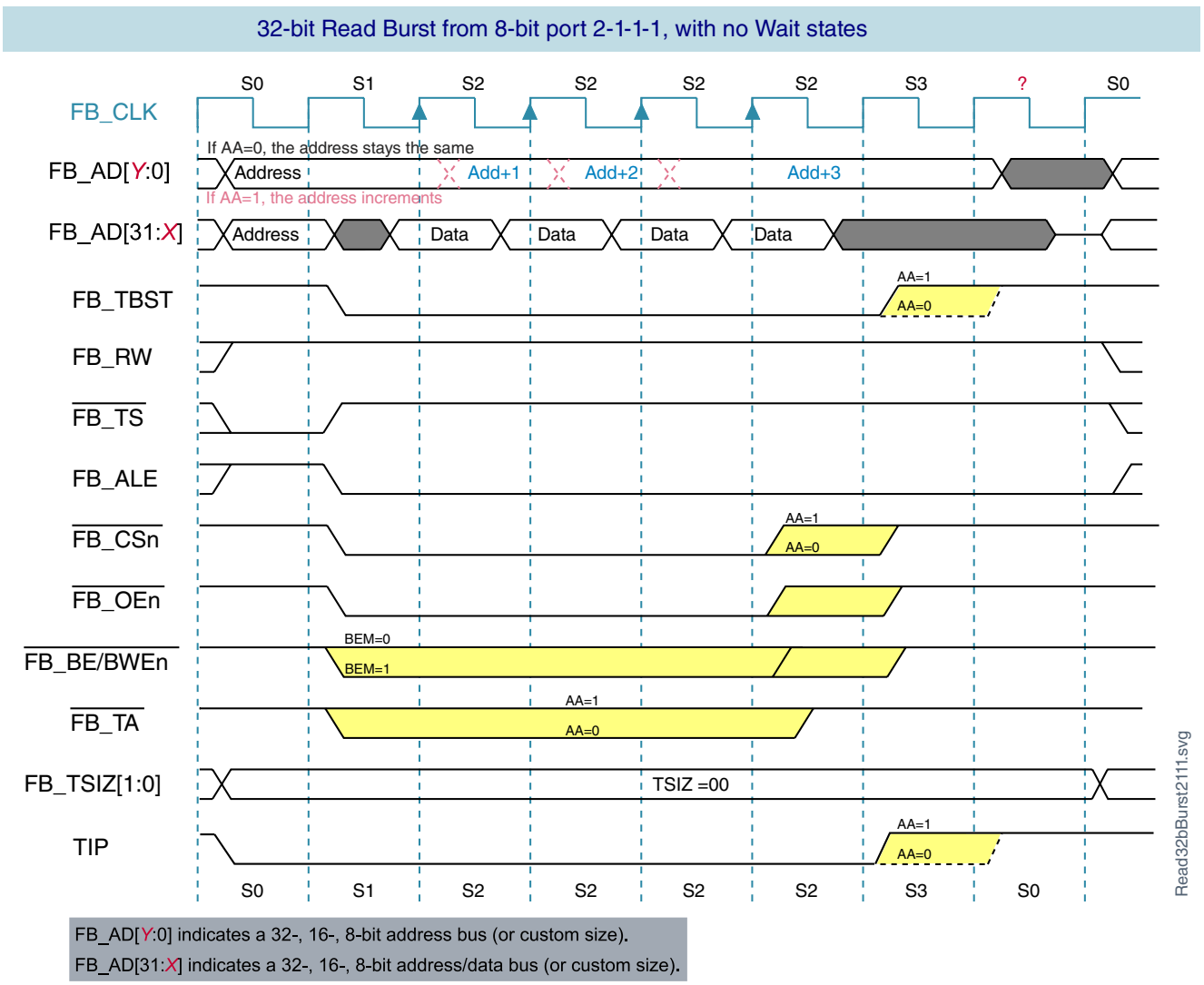


Figure 21-26. 32-bit-Read burst from 8-Bit port 2-1-1-1 (no wait states)

21.5.12.4 32-bit-Write burst to 8-Bit port 3-1-1-1 (no wait states)

The following figure shows a 32-bit write to an 8-bit external chip with burst enabled. The transfer results in a 4-beat burst and the data is driven on FB_AD[31:24]. The transfer size is driven at 32-bit (00b) throughout the bus cycle.

Note

The first beat of any write burst cycle has at least one wait state. If the bus cycle is programmed for zero wait states (CSCRn[WS] = 0b), one wait state is added. Otherwise, the programmed number of wait states are used.

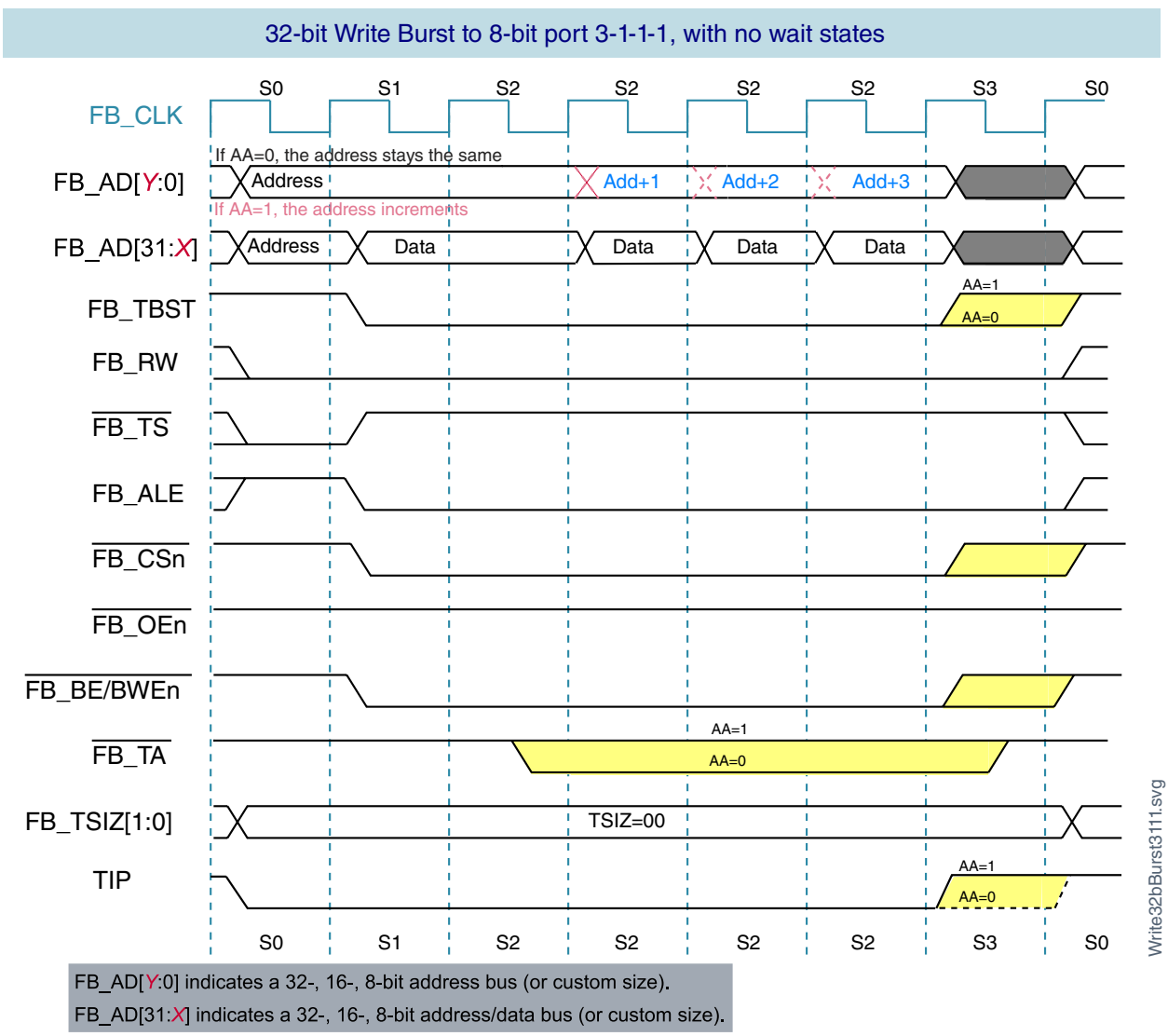


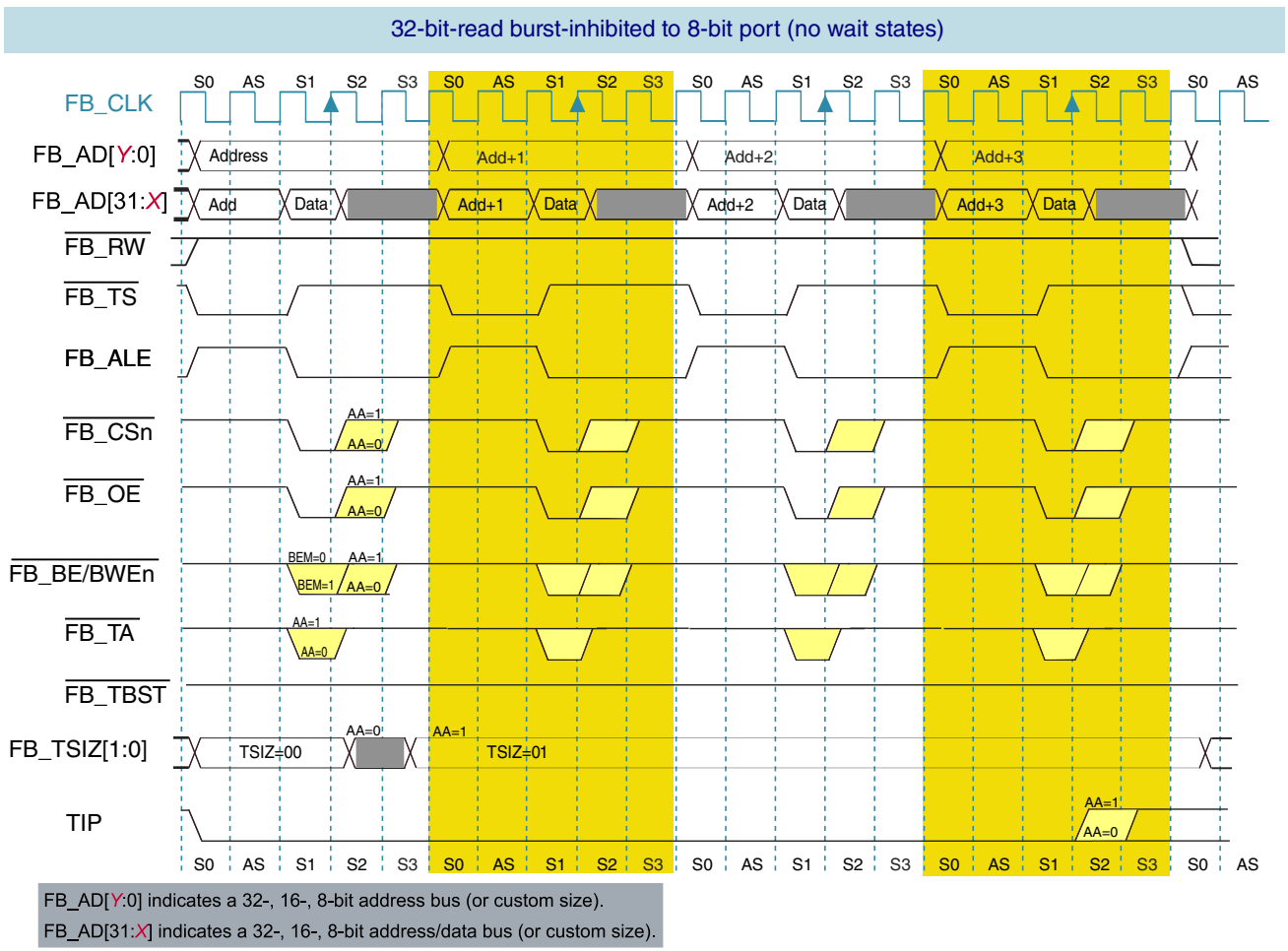
Figure 21-27. 32-bit-Write burst to 8-Bit port 3-1-1-1 (no wait states)

21.5.12.5 32-bit-read burst-inhibited from 8-bit port (no wait states)

The following figure shows a 32-bit read from an 8-bit device with burst inhibited. The transfer results in four individual transfers. The transfer size is driven at 32-bit (00b) during the first transfer and at byte (01b) during the next three transfers.

Note

There is an extra clock of address setup (AS) for each burst-inhibited transfer between states S0 and S1.



21.5.12.6 32-bit-write burst-inhibited to 8-bit port (no wait states)

The following figure shows a 32-bit write to an 8-bit device with burst inhibited. The transfer results in four individual transfers. The transfer size is driven at 32-bit (00b) during the first transfer and at byte (01b) during the next three transfers.

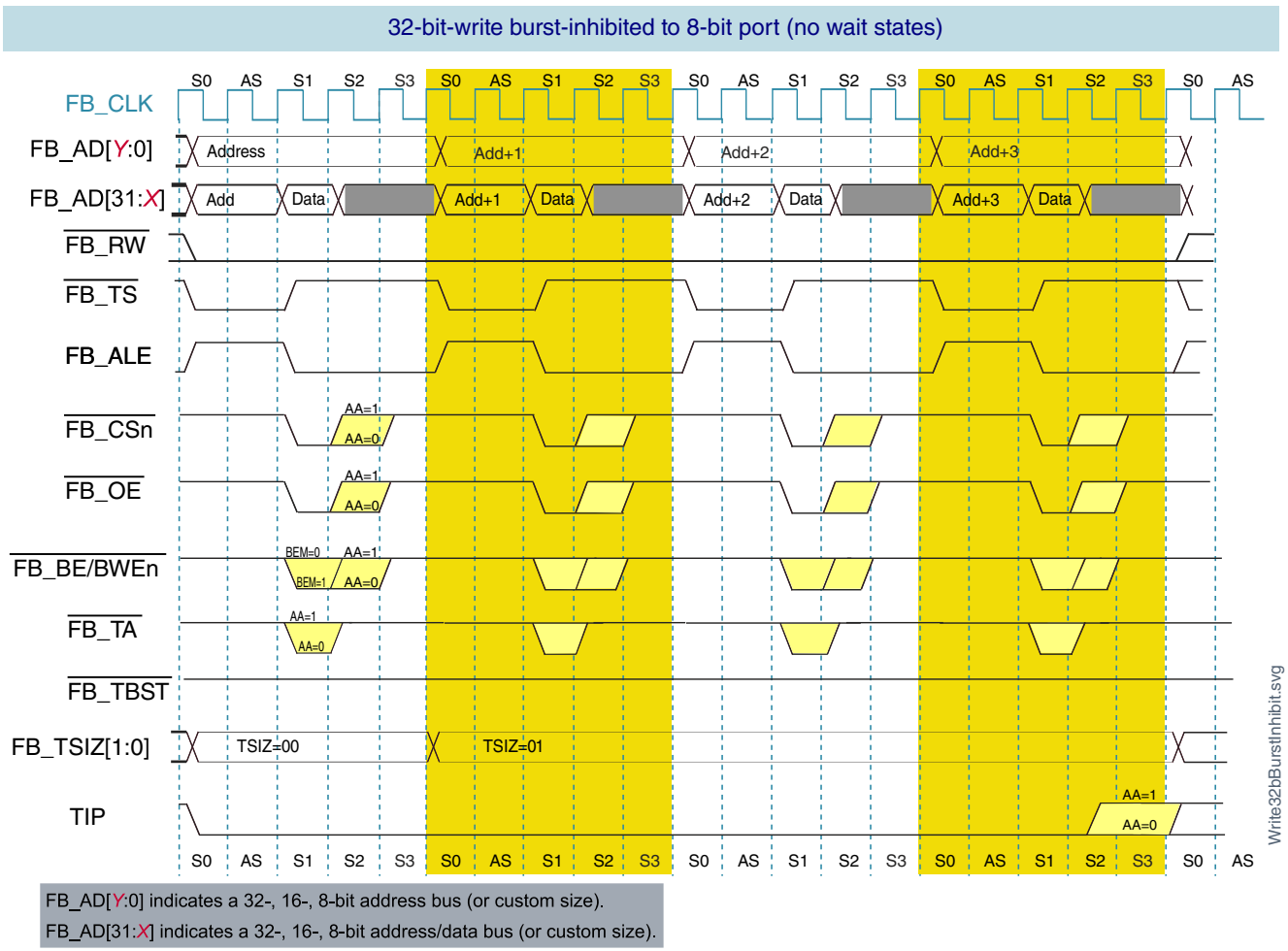


Figure 21-28. 32-bit-write burst-inhibited to 8-bit port (no wait states)

21.5.12.7 32-bit-read burst from 8-bit port 3-2-2-2 (one wait state)

The following figure illustrates another read burst transfer, but in this case a wait state is added between individual beats.

Note

CSCRn[WS] determines the number of wait states in the first beat. However, for subsequent beats, the CSCRn[WS] (or CSCRn[SWS] if CSCRn[SWSEN] = 1b) determines the number of wait states.

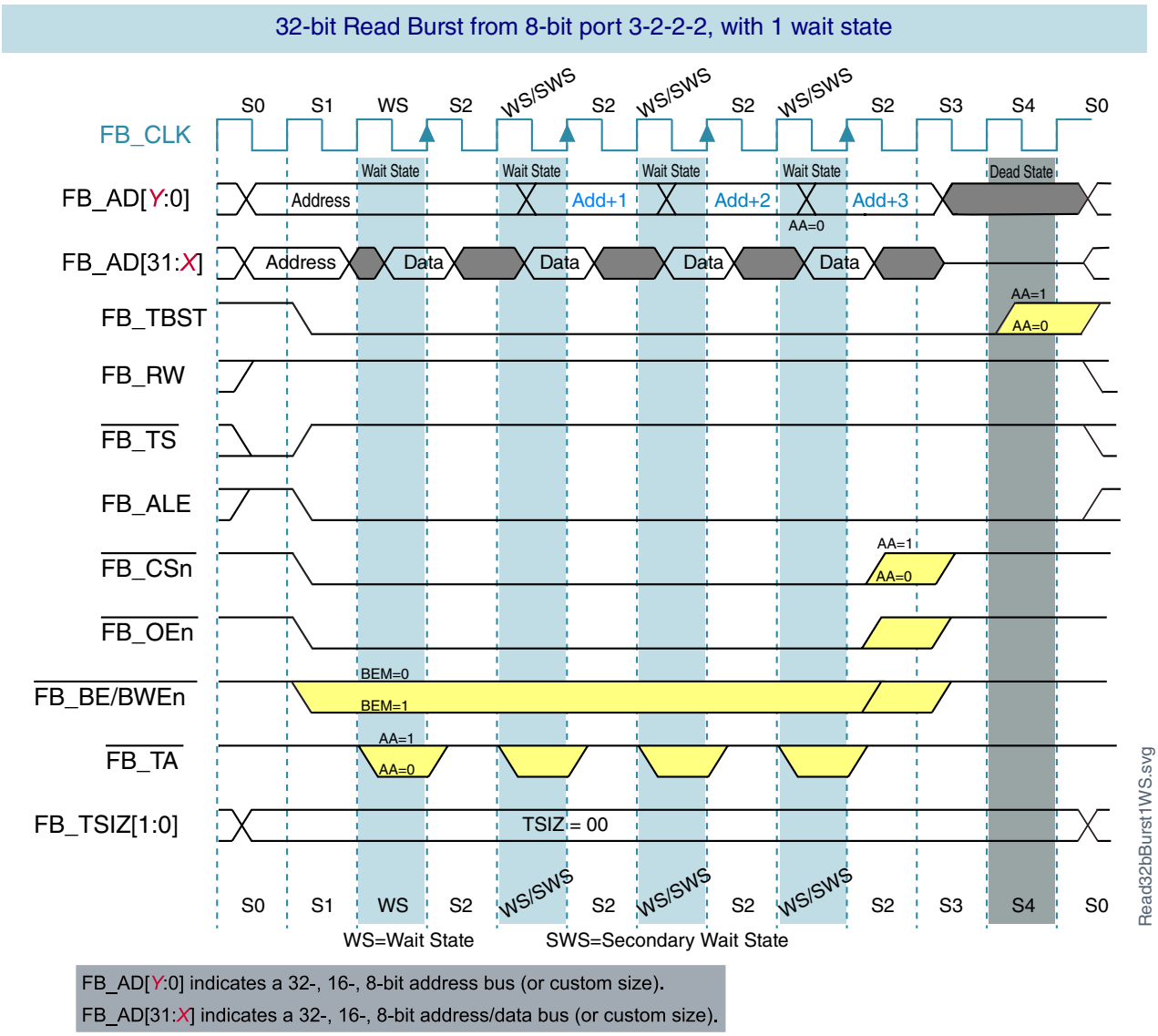


Figure 21-29. 32-bit-read burst from 8-bit port 3-2-2-2 (one wait state)

21.5.12.8 32-bit-write burst to 8-bit port 3-2-2-2 (one wait state)

The following figure illustrates a write burst transfer with one wait state.

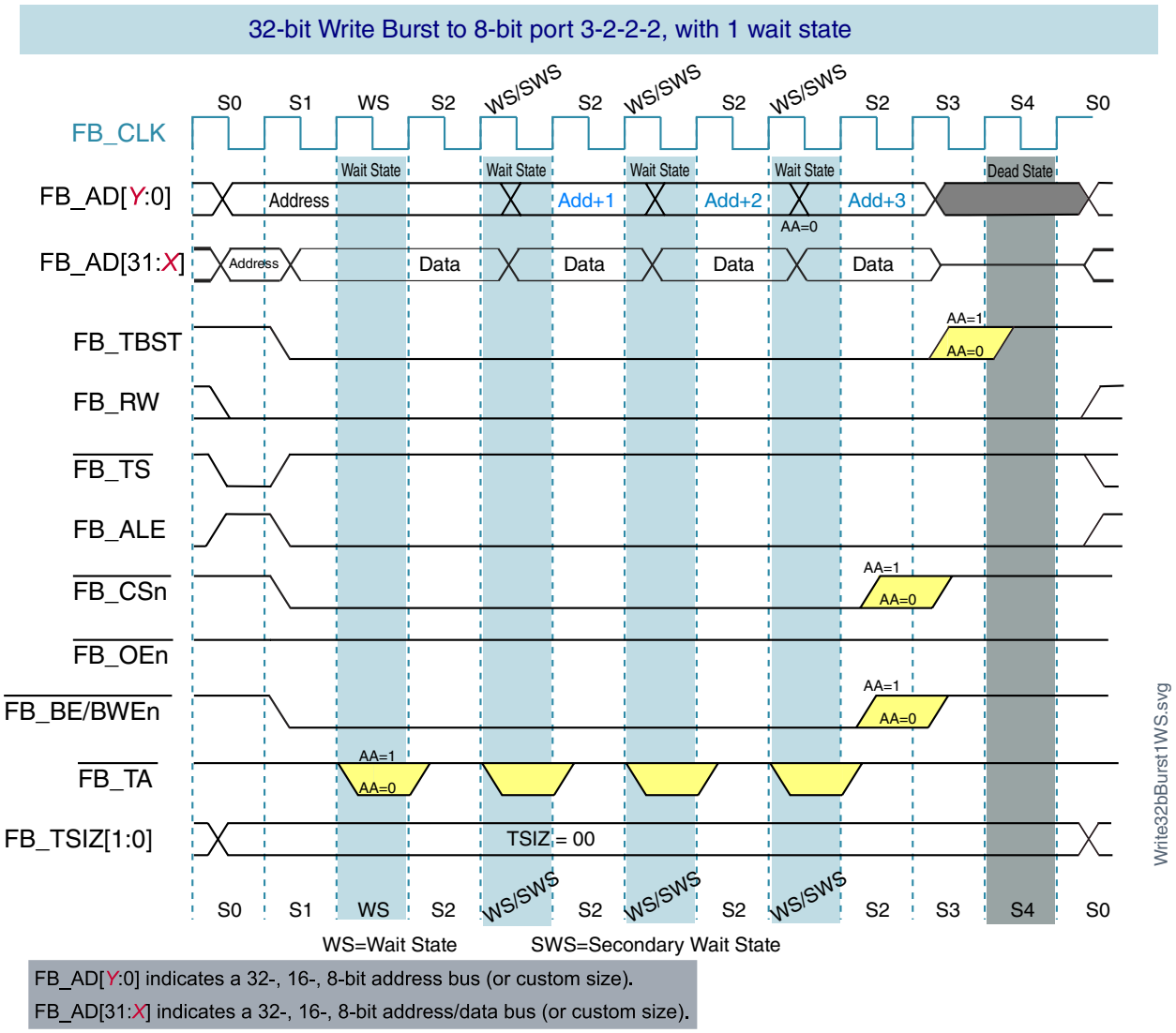


Figure 21-30. 32-bit-write burst to 8-bit port 3-2-2-2 (one wait state)

21.5.12.9 32-bit-read burst from 8-bit port 3-1-1-1 (address setup and hold)

If address setup and hold are used, only the first and last beat of the burst cycle are affected. The following figure shows a read cycle with one clock of address setup and address hold.

Note

- In **multiplexed address/data mode:** the address is driven on FB_AD only during the first cycle for internally- and externally-terminated cycles.

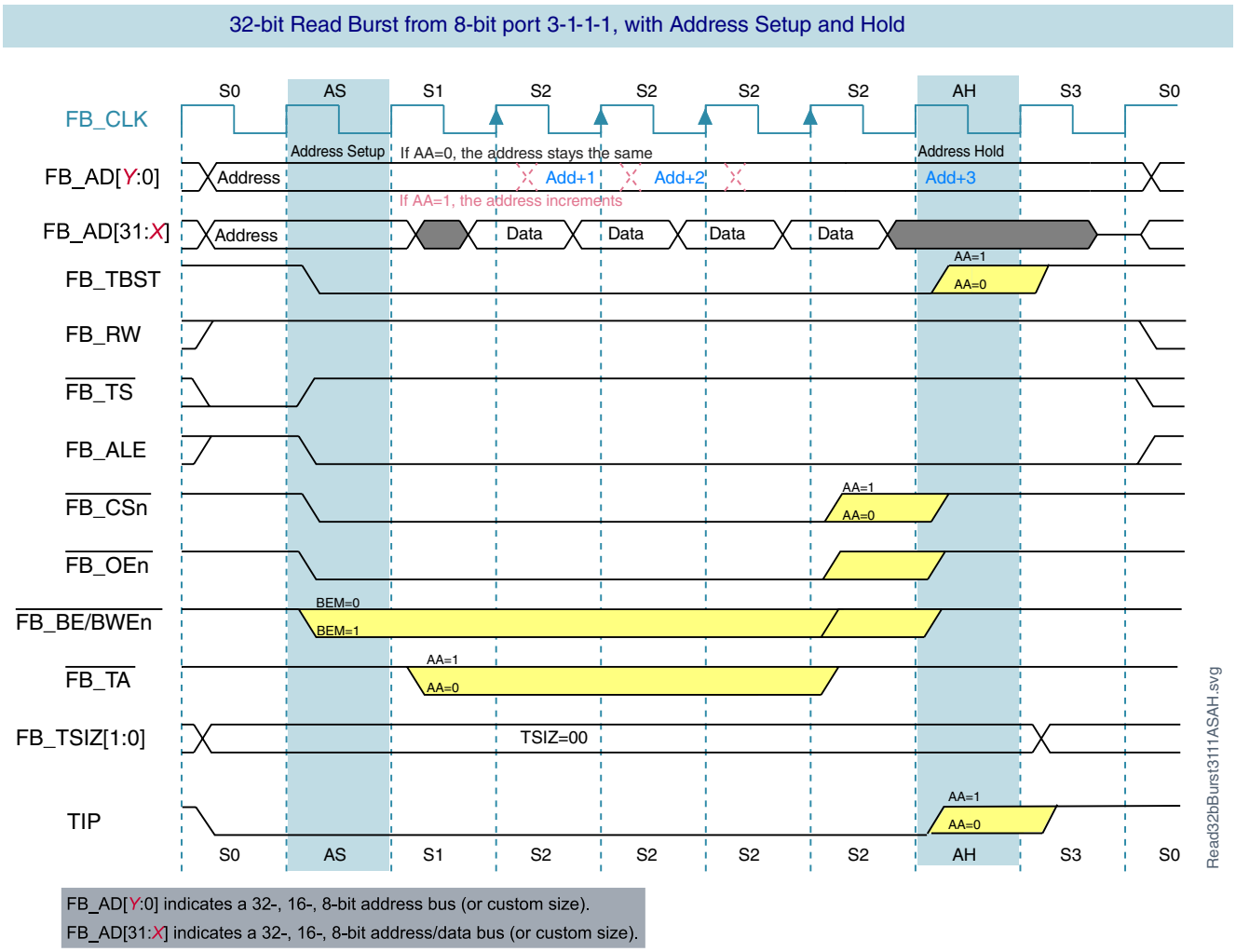


Figure 21-31. 32-bit-read burst from 8-bit port 3-1-1-1 (address setup and hold)

21.5.12.10 32-bit-write burst to 8-bit port 3-1-1-1 (address setup and hold)

The following figure shows a write cycle with one clock of address setup and address hold.

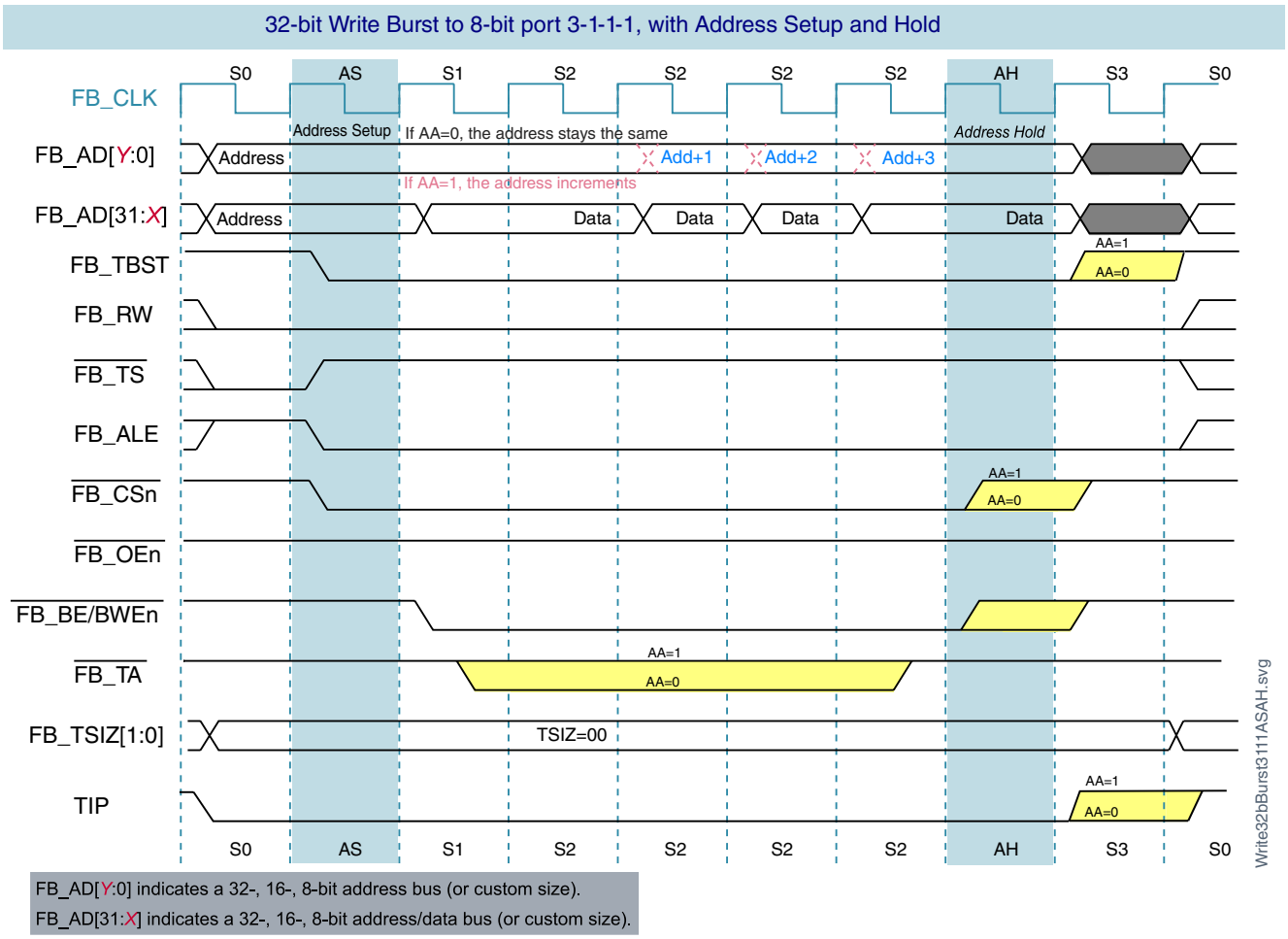


Figure 21-32. 32-bit-write burst to 8-bit port 3-1-1-1 (address setup and hold)

21.5.13 Extended Transfer Start/Address Latch Enable

The FB_TS_b/FB_ALE signal indicates that a bus transaction has begun and the address and attributes are valid. By default, the FB_TS_b/FB_ALE signal asserts for a single bus clock cycle. When CSCR*n*[EXTS] is set, the FB_TS_b/FB_ALE signal asserts and remain asserted until the first positive clock edge after FB_CS_b*n* asserts. See the following figure.

NOTE

When EXTS is set, CSCR*n*[WS] must be programmed to have at least one primary wait state.

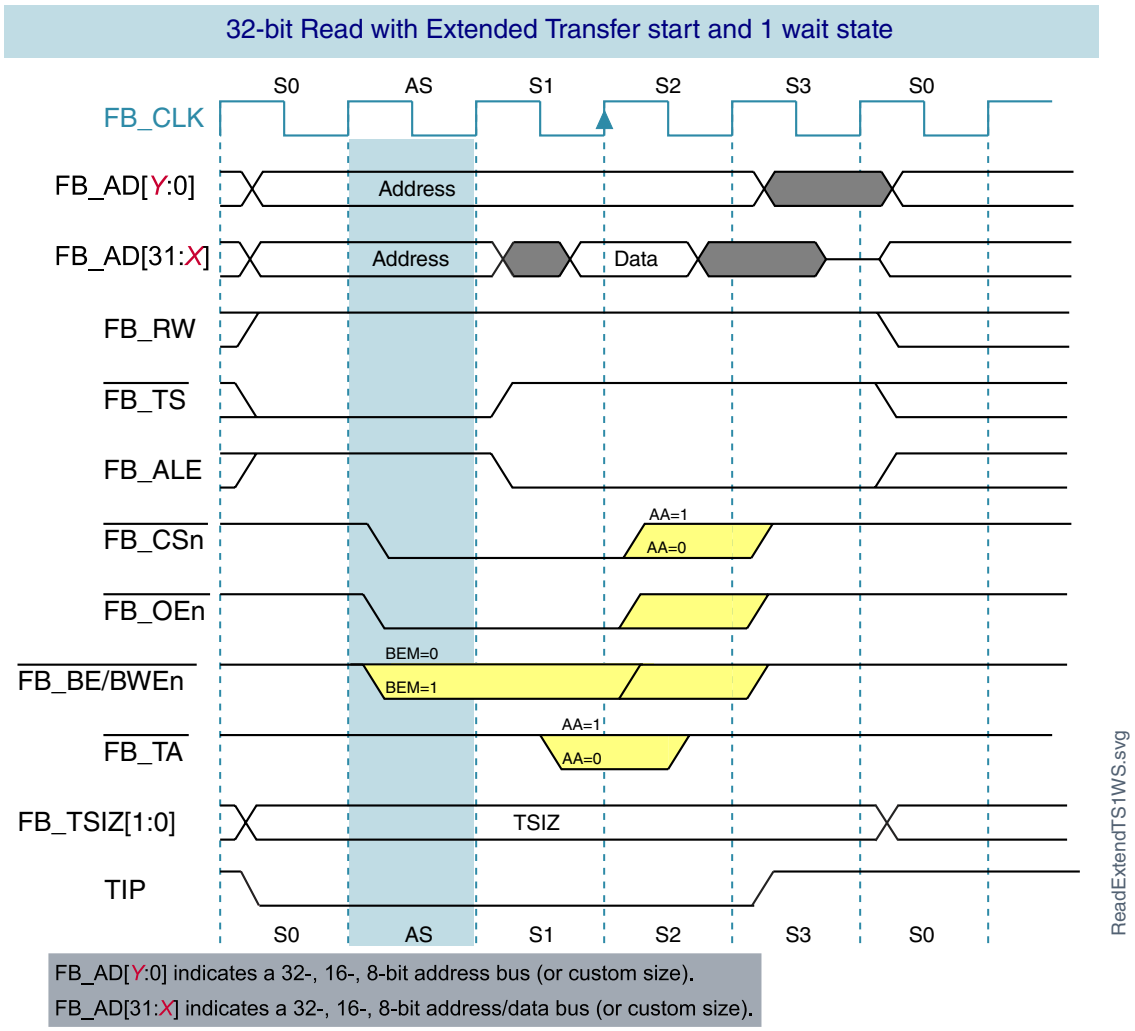


Figure 21-33. Read-Bus Cycle with CSCR*n*[EXTS] = 1 (One Wait State)

21.5.14 Bus errors

These types of accesses cause a transfer to terminate with a bus error:

- Any access when the clock to the module is disabled
- A write to a write-protected address range
- An access whose address is not in a range covered by a chip-select
- An access whose address is in a range covered by more than one chip-selects
- A write to a reserved address in the memory map
- A write to a reserved field in the CSPMCR
- Any FlexBus accesses when FlexBus is secure

If the auto-acknowledge feature is disabled (CSCR[AA] is 0) for an address that generates an error, then the transfer can be terminated by asserting FB_TA_b. If the processor must manage a bus error differently, then asserting an interrupt to the core along with FB_TA when the bus error occurs can invoke an interrupt handler.

The device can hang if FlexBus is configured for external termination and the CSPMCR is not configured for FB_TA.

21.6 Initialization/Application Information

21.6.1 Initializing a chip-select

Before using any other chip select, chip select 0 (CS0) should be initialized, to take it out of global chip select mode.

To initialize a chip-select:

1. Write to the associated CSAR.
2. Write to the associated CSCR.
3. Write to the associated CSMR, including writing "1b" to the Valid field (CSMRn[V]).

21.6.2 Reconfiguring a chip-select

To reconfigure a previously-used chip-select:

1. Invalidate the chip-select by writing 0b to the associated CSMR's Valid field (CSMRn[V]).
2. Write to the associated CSAR.
3. Write to the associated CSCR.
4. Write to the associated CSMR, including writing "1b" to the Valid field (CSMRn[V]).

Chapter 22
Multi Mode DDR Controller (MMDC)

22.1 Chip-specific MMDC information

Table 22-1. Reference links to related information

Topic	Related module	Reference
Full description	MMDC	MMDC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

22.1.1 Multi Mode DDR Controller (MMDC)

The MMDC is a configurable and high performance DDR controller that supports various configurations of LPDDR2 and LPDDR3 memory types. The MMDC consists of a core and PHY. The following table shows the configuration of the MMDC.

Table 22-2. MMDC configuration

Parameter	Description
Name	Multi Mode DDR Controller
Instances	1
Configurable features	• LPDDR2 x32, x16 • LPDDR3 x32
Interface speed	380 MHz
External I/O pins	See the attached IOMUXC pinout spreadsheet for pin details on this device

22.1.2 Fine tuning control

In i.MX 7ULP, write fine tuning, read fine tuning, and ZQ fine tuning are controlled through IOMUXC DDR registers. See the IOMUXC chapter and IOMUXC pin spreadsheet attached for register details.

22.1.3 MMDC register implementation

In this device, the following must be taken into consideration for register implementation.

- MDSCR[CON_ACK] bitfield: After a reset sequence, CON_REQ will be asserted, so as there are no pending AXI accesses to be cleaned, CON_ACK will be asserted indicating that MMDC registers configuration is permitted. See MMDC Core Special Command Register (MMDC_MDSCR)
- MDSCR[WL-EN]: Write leveling feature is not supported in LPDDR3. So, this bit is reserved. See MMDC Core Special Command Register (MMDC_MDSCR)
- MAPSR[4] behaves writeable by the CA7
- MAPSR[LPMD] behaves non-writeable by the CA7
- MAPSR[DVFS] behaves non-writeable by the CA7

22.2 Overview

MMDC is a multi-mode DDR controller that supports LPDDR2/LPDDR3/x32 memory types. MMDC is configurable, high performance, and optimized.

The following figure shows the MMDC block diagram.

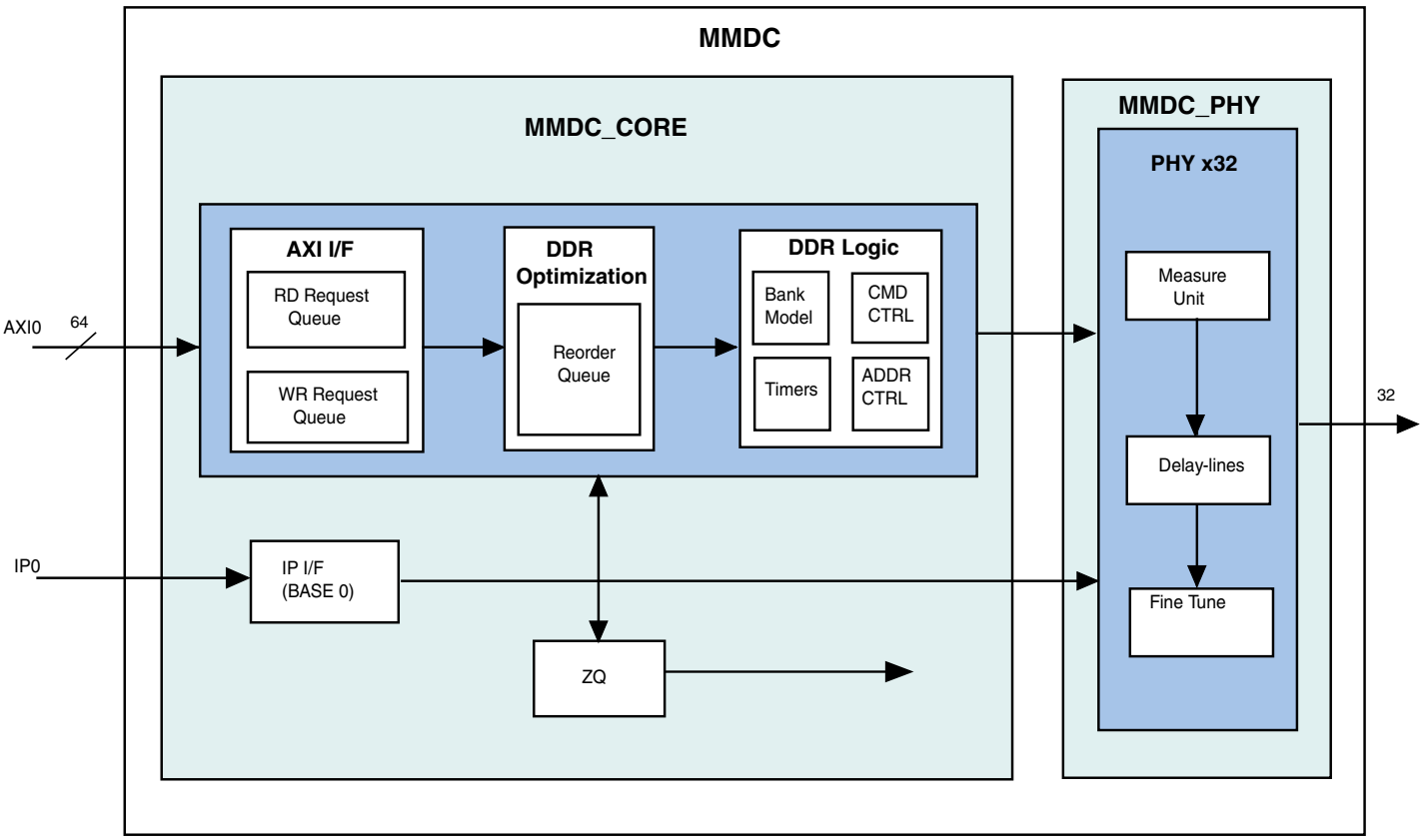


Figure 22-1. MMDC block diagram

MMDC consists of a core (MMDC_CORE) and PHY (MMDC_PHY).

- The core is responsible for communication with the system through an AXI interface, DDR command generation, DDR command optimizations, and a read/write data path.
- The PHY is responsible for the timing adjustment; it uses special calibration mechanisms to ensure data capture margin at a clock rate of up to 380 MHz.

The internal memory map (configuration registers) of the MMDC can be configured through an IP channel (IP0).

22.2.1 MMDC feature summary

The table found here summarizes the MMDC features.

Table 22-3. MMDC feature summary

Feature	Details
DDR standards	• LPDDR2/LPDDR3 x16, x32 • Does not support LPDDR1MDDR or DDR2

Table continues on the next page...

Table 22-3. MMDC feature summary (continued)

Feature	Details

DDR interface	• x32 data bus width
	• Density per DDR device of 256 Mbits–8 Gbits with the following column and row combinations:
	  • Column size of 8–12 bits
	  • Row size of 11–16 bits
	• Two chip selects (LPDDR2/LPDDR3)
	• Supports burst length of 4 for LPDDR2
	• Supports burst length of 8 for LPDDR3

DDR performance	• MMDC running at up to 380 MHz (760MT/s), see SCG block for actual clock frequencies supported. • Supports Real-Time priority by means of QoS sideband priority signals from the chip to enable various priority levels in the re-ordering mechanism: real-time, latency sensitive, normal priority. • Page hit/page miss optimizations • Consecutive read/write access optimizations • Supports deep read and write request queues to enable bank prediction. • Drives back the critical word in a read transaction as soon as it is received by the DDR device (does not wait until the whole data phase has been completed). • Keeps tracking of open memory pages • Supports bank interleaving **NOTE:** Due to reordering and optimization mechanisms (per different AXI Identifier (ID)), the transactions towards the DDR device may be driven in a different ID order than was received by the AXI master. In a similar fashion, the write response, read response or read data may be driven to the AXI master in a different ID order.

AXI interface	• AXI bus compliant
	• Supports bus transfers of 8, 16 , 32 , 64 bits (single accesses and bursts) running at 380 MHz.
	• Supports AXI bursts length of up to 16
	• Supports burst types of WRAP, INCR and FIXED
	• Supports 16 bits AXI ID
	• Write data interleave depth is 1 (no support for Write Data Interleave)
	• Supports write data before address
	• Supports buffered/non-buffered accesses (AWCACHE[0] = 0b means a non-bufferable access and AWCACHE[0] = 1b means a bufferable access). The rest of the CACHE options are not supported
	  • To keep data access coherency between write and read access of the same master, the response signal is sent as follows:
	  • Bufferable write access—BRESP will be sent when last data of the access has entered the MMDC.
	  • Non-bufferable write access—BRESP will be sent when the data was physically written into the external memory device.
	• Supports four exclusive monitors per configurable ID for only a single access with a size of up to 64 bits
	• Supports AXI responses as follows:
	  • Okay in case the access has been successful or exclusive access failure
	  • Slave error in case of security violation
	  • Exclusive okay in case the read or the write portion of an exclusive access has been successful

DDR calibration and delay-lines.	• Supports various calibration processes which can be performed either automatically (hardware) or manually (software) towards either CS0 or CS1. (At the end of the process the delay-lines will work with one set of results.) The following calibration processes are supported:

Table continues on the next page...

Table 22-3. MMDC feature summary (continued)

Feature	Details

	• ZQ calibration for external DDR device (in LPDDR2/LPDDR3 through MRW command)
	  • Can be handled automatically for ZQ Short (periodically) and ZQ Long (at exit from self-refresh)
	  • Can be handled manually at ZQ INIT
	• ZQ calibration for DDR I/O pads for calibrating the DDR driving strength
	  • The sequence can be handled automatically by hardware
	  • The sequence can be handled step by step manually by software
	• Read data calibration. Adjustment of read DQS with read data byte.
	• Write data calibration. Adjustment of write DQS with write data byte.
	• Write leveling calibration. Adjustment of write DQS with CK (DDR differential clock).
	• Read fine tuning. Adjustment of up to 7 delay-line units for each read data bit.
	• Write fine tuning. Adjustment of up to 3 delay-line units for each read data bit.
	• Periodic delay-line measurement for keeping its accuracy during refresh interval.
	• Additional fine tuning delay lines to adjust DDR clock delay, DDR clock duty cycle, DQS duty cycle.

Power saving	• Support of dynamic voltage, frequency change and self-refresh mode entry through hardware and software negotiation with the system (request/acknowledge handshake)
	  • Upon hardware or software self-refresh request assertion, further AXI requests are blocked (even before the assertion of the acknowledge).
	  • During self-refresh mode the system may deassert the operating clock of the MMDC for power saving.
	  • During self-refresh mode the clock (CK) that is driven to the DDR device will be gated for power saving.
	• Supports automatic self-refresh and power down entry and exit
	  • In automatic self-refresh, the internal operating clock will be gated for power saving.
	• Automatic active and precharge power down timer per chip select (one chip select can enter power down while the other is still working)
	• While CS (chip-select) is inactive (high) the command and address buses are not toggling for power saving.

DDR general	• Configurable timing parameters • Configurable refresh scheme • Page boundary crossing support • Automatically generates precharge command and activates the next row • Supports MRW and MRR commands for LPDDR2/LPDDR3 • Software control in LPDDR2/LPDDR3 mode for switching to derated timing parameters and/or update the refresh rate according to temperature sensor • Debug and profiling capabilities

22.3 External Signals

The table found here describes the external signals of MMDC.

Table 22-4. MMDC external signals[1]

Signal name	Description	Direction
DDR_CA0	MMDC Address Bus Signal	I/O
DDR_CA1	MMDC Address Bus Signal	I/O
DDR_CA2	MMDC Address Bus Signal	I/O
DDR_CA3	MMDC Address Bus Signal	I/O
DDR_CA4	MMDC Address Bus Signal	I/O
DDR_CA5	MMDC Address Bus Signal	I/O
DDR_CA6	MMDC Address Bus Signal	I/O
DDR_CA7	MMDC Address Bus Signal	I/O
DDR_CA8	MMDC Address Bus Signal	I/O
DDR_CA9	MMDC Address Bus Signal	I/O
DDR_CKE0	MMDC Clock Enable 0	I/O
DDR_CKE1	MMDC Clock Enable 1	I/O
DDR_CLK0	MMDC Positive Clock Signal	I/O
DDR_CLK0_b	MMDC Negative Clock Signal	I/O
DDR_CS0_b	MMDC Chip Select 0	I/O
DDR_CS1_b	MMDC Chip Select 1	I/O
DDR_DQ0	MMDC Data Bus Signal	I/O
DDR_DQ1	MMDC Data Bus Signal	I/O
DDR_DQ10	MMDC Data Bus Signal	I/O
DDR_DQ11	MMDC Data Bus Signal	I/O
DDR_DQ12	MMDC Data Bus Signal	I/O
DDR_DQ13	MMDC Data Bus Signal	I/O
DDR_DQ14	MMDC Data Bus Signal	I/O
DDR_DQ15	MMDC Data Bus Signal	I/O
DDR_DQ16	MMDC Data Bus Signal	I/O
DDR_DQ17	MMDC Data Bus Signal	I/O
DDR_DQ18	MMDC Data Bus Signal	I/O
DDR_DQ19	MMDC Data Bus Signal	I/O
DDR_DQ2	MMDC Data Bus Signal	I/O
DDR_DQ20	MMDC Data Bus Signal	I/O
DDR_DQ21	MMDC Data Bus Signal	I/O
DDR_DQ22	MMDC Data Bus Signal	I/O
DDR_DQ23	MMDC Data Bus Signal	I/O

Table continues on the next page...

Table 22-4. MMDC external signals[1] (continued)

Signal name	Description	Direction
DDR_DQ24	MMDC Data Bus Signal	I/O
DDR_DQ25	MMDC Data Bus Signal	I/O
DDR_DQ26	MMDC Data Bus Signal	I/O
DDR_DQ27	MMDC Data Bus Signal	I/O
DDR_DQ28	MMDC Data Bus Signal	I/O
DDR_DQ29	MMDC Data Bus Signal	I/O
DDR_DQ3	MMDC Data Bus Signal	I/O
DDR_DQ30	MMDC Data Bus Signal	I/O
DDR_DQ31	MMDC Data Bus Signal	I/O
DDR_DQ4	MMDC Data Bus Signal	I/O
DDR_DQ5	MMDC Data Bus Signal	I/O
DDR_DQ6	MMDC Data Bus Signal	I/O
DDR_DQ7	MMDC Data Bus Signal	I/O
DDR_DQ8	MMDC Data Bus Signal	I/O
DDR_DQ9	MMDC Data Bus Signal	I/O
DDR_DQM0	MMDC Data Mask Signal	I/O
DDR_DQM1	MMDC Data Mask Signal	I/O
DDR_DQM2	MMDC Data Mask Signal	I/O
DDR_DQM3	MMDC Data Mask Signal	I/O
DDR_DQS0	MMDC Positive DQS Signal 0	I/O
DDR_DQS0_b	MMDC Negative DQS Signal 0	I/O
DDR_DQS1	MMDC Positive DQS Signal 1	I/O
DDR_DQS1_b	MMDC Negative DQS Signal 1	I/O
DDR_DQS2	MMDC Positive DQS Signal 2	I/O
DDR_DQS2_b	MMDC Negative DQS Signal 2	I/O
DDR_DQS3	MMDC Positive DQS Signal 3	I/O
DDR_DQS3_b	MMDC Negative DQS Signal 3	I/O
DDR_ZQ0	MMDC ZQ	I/O
DDR_ZQ1	HSIC ZQ	I/O

1. For details, see the IOMUXC and Pinout spreadsheet attached with the RM.

22.4 Clocks

The table found here describes the clock sources for MMDC.

See the chip-specific clocking information for details.

NOTE

The terms *clocks* and *cycles* are used interchangeably and refer to the clock period of the main ddr clock , commonly referred to as the DDR frequency.

Table 22-5. MMDC Clocks

Clock name	Clock Root	Description
aclk_fast_core_p0	mmdc_axi_clk_root	Fast clock (channel 1)
ipg_clk_p0	ipg_clk_root	Peripheral clock (channel 1)
aclk_fast_phy_p0	mmdc_axi_clk_root	Fast clock (channel 1 - PHY)

22.5 Functional Description

This section provides a complete functional description of the block.

22.5.1 Write/Read data flow

22.5.1.1 Write data flow

1. Write requests are received into an 8 entry request FIFO. Access is received only when there are at least two available entries. Each entry holds all of the AXI attributes.
   - If the burst length is greater than 8, the access splits into two accesses: one with burst length 8 and the other with the remainder.
   - The access can be performed as soon as the entire data phase of the associated write request is completed (all data beats were received).
2. A simple round-robin arbitration between the pending read and write accesses is performed, and the pointer to this stage's winner access is sent to the re-ordering buffer.
3. The reordering mechanism is activated to find the winner access, which is the access that best utilizes the DDR bus, based on its dynamic score. For further information see Dynamic scoring mode (Arbitration Winning Conditions).
4. The winner write access at the previous stage is received and is held for dispatch to the DDR logic.
5. When the DDR command control unit is ready to accept the write request, it issues (if needed) a precharge/active command to the DDR device according to the status of the bank model and the parameters of the timers.
6. The DDR logic drives the associated data to the DDR device through the DDR PHY.

22.5.1.2 Read data flow

1. Read requests are received into a 16 entry request FIFO in MMDC if there are at least two available entries. Each entry holds all of the AXI attributes.

NOTE

If the burst length is greater than 8, the access splits into 2 accesses (one with burst length 8 and the other with the remainder).

2. A simple round-robin arbitration between the pending read and write accesses is performed and the pointer to this phase's winner access is sent to the re-ordering buffer.
3. The reordering mechanism is activated to find the winner access, which is the access that best utilizes the DDR bus, based on its dynamic score. For further information see Dynamic scoring mode (Arbitration Winning Conditions).
4. The winner read access at the previous stage is sampled and is held for dispatch to the DDR logic. This read access will be dispatched when there is at least one free slot in the read data buffer to store the data.
5. When the DDR command control unit is ready to accept the read request, it issues (if needed) a precharge/active command to the DDR device according to the status of the bank model and the parameters of the timers.
6. The MMDC PHY samples the read data, and the DDR logic transfers the data to the associated slot in the read data buffer.
7. MMDC transfers the data back to the master.

22.5.2 MMDC initialization

Because the MMDC is disabled when the chip exits reset, no clock is driven to the DDR device and the whole interface towards the DDR device is inactive. The following steps are required to activate the MMDC properly.

NOTE

To guarantee that the DRAM_SDCKE signals are kept low during the power-up and reset sequences of the chip (as defined by JEDEC), you must connect those signals to pull-down resistors.

1. Set MDSCR[CON_REQ], which sets the configuration request; note that because the MMDC is disabled, there is no need to poll the configuration acknowledge bit at MDSCR[CON_ACK].

2. Configure the desired timing parameters at the MDCFG0, MDCFG1, and MDCFG2 registers.
3. Configure the DDR type and other miscellaneous parameters at the MDMISC register.
4. Configure the required delay while leaving reset, at the MDOR register.
5. Configure the DDR physical parameters (density and burst length) at the MDCTL register.
6. Perform a ZQ calibration of the MMDC module to correctly initialize drive strengths.
7. Enable MMDC with the desired chip select at MDCTL[SDE_0] (for chip select 0) and MDCTL[SDE_1] (for chip select 1). At this point, MMDC starts the reset and initialization sequence related to DRAM_SDCKE as defined by JEDEC.
8. Complete the initialization sequence as defined by JEDEC by issuing MRS/MRW commands for (ZQ, PRE, and so on). To issue those commands, configure the appropriate command and address at the MDSCR register.

NOTE

A Precharge All command must be issued prior to the MRW command to ensure robust DDR initialization. This command is required to be issued to both chip selects if two chip selects are utilized in the system.

9. Program the DDR mode registers by configuring the appropriate command and address at the MDSCR register.
10. Configure the power down and self-refresh entry and exit parameters at the MDPDC and MAPSR registers.
11. Configure the ZQ scheme at the MPZQHWCTRL and MPZQLP2CTL registers.
12. Configure and activate the periodic refresh scheme at the MDREF register.
13. Deassert the configuration request by clearing MDSCR[CON_REQ].

NOTE

Steps 1 through 6 are non-blocking and can be done in any order.

Upon completion of these steps, MMDC is ready for work and to process AXI accesses.

NOTE

To achieve better timing and better precision, it is recommended that users configure the MMDC PHY delay parameters by operating either the automatic or manual calibration process. Before starting any calibration process, you must disable the periodic refresh scheme (MDREF[REF_SEL] = 11) and then issue a manual refresh command by configuring

MDSCR[CMD] to 2h. For further information, see Calibration Process.

22.5.3 Configuring the MMDC registers

To safely modify MMDC's internal configuration registers, MMDC must be placed into configuration mode.

Use the following steps to enter configuration mode.

1. Issue a configuration request by setting MDSCR[CON_REQ].
2. Poll on configuration acknowledge until it is set at MDSCR[CON_ACK].

At this point, MMDC enters configuration mode and accessing the MMDC registers is permitted.

NOTE

During configuration mode, MMDC prevents further AXI accesses from being acknowledged.

Upon deassertion of MDSCR[CON_REQ], MMDC leaves configuration mode and AXI accesses are processed.

22.5.4 MMDC Address Space

22.5.4.1 Address decoding

MMDC supports up to two consecutive chip selects, each with the same density.

It is optional to configure the partition between the chip selects through MDASP[CS0_END].

The incoming AXI address bus is 32 bits. MMDC decodes each access as follows:

1. chip select
2. bank number
3. row number
4. column number

The following registers in the MMDC define the DDR address space:

- MDMISC[DDR_4_BANK]—Defines either 4 or 8 banks in the DDR device
- MDCTL[DSIZ]—Defines the DDR data bus width of x16, x32 or x64

- MDMISC[BI]—Defines whether bank interleaving is on or off
- MDCTL[COL]—Defines the column size of the DDR device
- MDCTL[ROW]—Defines the row size of the DDR device

The following tables show address decoding examples for x16 and x32 bit DDR devices when bank interleaving is both on and off. It is assumed that the configuration is as follows: 8 banks (3 bits), 15 bit assignment for the row, and 10 bit assignment for the column. The total density is 256 MWords (512 Mbytes for x16 and 1 Gbyte for x32).

NOTE

Chip selection is done by comparing the 7 most significant address bits (ARADDR[31:25]/AWADDR[31:25]) with MDASP[CS0_END].

Table 22-6. Address decoding—bank interleaving off

AXI ADDRESS	x16 DDR	x32 DDR
A29	—	BANK[2]
A28	BANK[2]	BANK[1]
A27	BANK[1]	BANK[0]
A26	BANK[0]	ROW[14]
A25	ROW[14]	ROW[13]
A24	ROW[13]	ROW[12]
A23	ROW[12]	ROW[11]
A22	ROW[11]	ROW[10]
A21	ROW[10]	ROW[9]
A20	ROW[9]	ROW[8]
A19	ROW[8]	ROW[7]
A18	ROW[7]	ROW[6]
A17	ROW[6]	ROW[5]
A16	ROW[5]	ROW[4]
A15	ROW[4]	ROW[3]
A14	ROW[3]	ROW[2]
A13	ROW[2]	ROW[1]
A12	ROW[1]	ROW[0]
A11	ROW[0]	COL[9]
A10	COL[9]	COL[8]
A9	COL[8]	COL[7]
A8	COL[7]	COL[6]
A7	COL[6]	COL[5]
A6	COL[5]	COL[4]
A5	COL[4]	COL[3]
A4	COL[3]	COL[2]

Table continues on the next page...

Table 22-6. Address decoding—bank interleaving off (continued)

AXI ADDRESS	x16 DDR	x32 DDR
A3	COL[2]	COL[1]
A2	COL[1]	COL[0]
A1	COL[0]	—
A0	—	—

Table 22-7. Address decoding—bank interleaving on

AXI ADDRESS	x16 DDR	x32 DDR
A29	—	ROW[14]
A28	ROW[14]	ROW[13]
A27	ROW[13]	ROW[12]
A26	ROW[12]	ROW[11]
A25	ROW[11]	ROW[10]
A24	ROW[10]	ROW[9]
A23	ROW[9]	ROW[8]
A22	ROW[8]	ROW[7]
A21	ROW[7]	ROW[6]
A20	ROW[6]	ROW[5]
A19	ROW[5]	ROW[4]
A18	ROW[4]	ROW[3]
A17	ROW[3]	ROW[2]
A16	ROW[2]	ROW[1]
A15	ROW[1]	ROW[0]
A14	ROW[0]	BANK[2]
A13	BANK[2]	BANK[1]
A12	BANK[1]	BANK[0]
A11	BANK[0]	COL[9]
A10	COL[9]	COL[8]
A9	COL[8]	COL[7]
A8	COL[7]	COL[6]
A7	COL[6]	COL[5]
A6	COL[5]	COL[4]
A5	COL[4]	COL[3]
A4	COL[3]	COL[2]
A3	COL[2]	COL[1]
A2	COL[1]	COL[0]
A1	COL[0]	—
A0	—	—

NOTE

In cases where this is an access to a non-initialized or disconnected chip select, behavior may be unexpected.

22.5.4.2 Chip select settings

MMDC drives the incoming access to either CS0 or CS1 by comparing the 7 most significant address bits (ARADDR[31:25]/AWADDR[31:25]) with MDASP[CS0_END].

Generally, the total density per chip-select must be a power of two and the total density must be the same for each chip-select.

Creating 4 Gbyte address space with 2 Gbyte CS density and Creating 2 Gbyte address spaces with 1 Gbyte CS density show how to create a continous address space and configure the MMDC accordingly.

22.5.4.2.1 Creating 4 Gbyte address space with 2 Gbyte CS density

If the DDR memory space allocation is 4 Gbytes, only one configuration of chip select partition is allowed. The register MDASP[CS0_END] should be set to 2 Gbytes partition (16Gb).

The figure below shows the associated memory space.

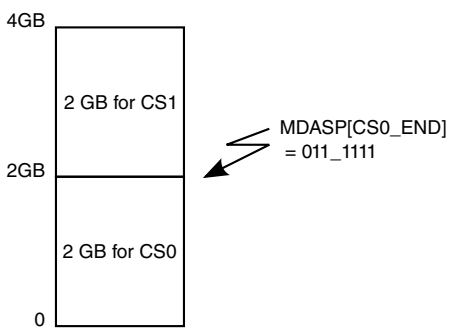


Figure 22-2. Chip select partion—2 Gbytes per chip select

22.5.4.2.2 Creating 2 Gbyte address spaces with 1 Gbyte CS density

If the DDR memory space allocation is 2 Gbytes, there are three options for configuring the chip select partition:MDASP[CS0_END] to 001_1111 (1 Gbyte), MDASP[CS0_END] to 011_1111 (2 Gbytes), and MDASP[CS0_END] to 101_1111 (3 Gbytes).

If DDR memory space allocation is 2 Gbytes, there are three options for configuring the chip select partition:

- MDASP[CS0_END] to 001_1111 (1 Gbyte)
- MDASP[CS0_END] to 011_1111 (2 Gbytes)
- MDASP[CS0_END] to 101_1111 (3 Gbytes)

The figure below shows the associated memory space:

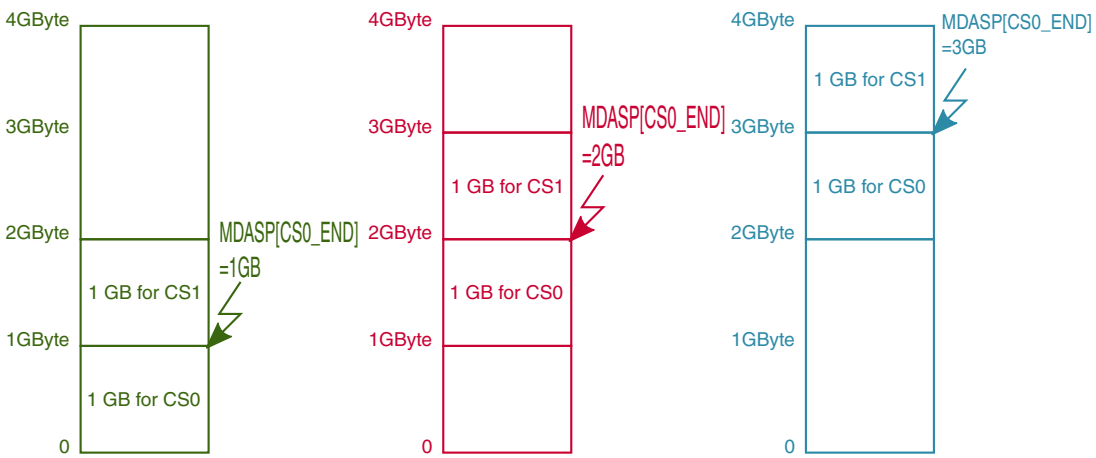


Figure 22-3. Chip select partion—1 Gbyte per chip select

22.5.5 Power Saving and Clock Frequency Change modes

22.5.5.1 Power saving general

MMDC supports multiple DDR power saving modes.

NOTE

At default, the power saving modes are disabled. These modes may dramatically decrease the power consumption of DDR memories.

1. Self-refresh entry to the entire DDR device (for both chip select 0 and 1) can be activated through two mechanisms:
   - LPMD (Low Power Mode)
     - Hardware handshaking (LPMD/LPACK) with the clock module in the system

- Software handshaking by setting the field MAPSR[LPMD] and polling MAPSR[LPACK]
- Automatic entry by configuring the amount of idle cycle for triggering self-refresh entry through MAPSR[PST] and by clearing MAPSR[PSD]

- DVFS (Dynamic Voltage and Frequency Change)
  - Software handshaking by setting the field MAPSR[DVFS] and polling MAPSR[DVACK]

NOTE

If hardware or software requests for self-refresh entry were detected by the MMDC (even before the assertion of the LPACK), no write or read accesses will be acknowledged until the deassertion of those requests.

2. Automatic active/precharge power down entry to a specific chip select can be activated by configuring the ESDPDC register:
   - PWDT_0/PWDT_1 - define the number of idle cycles before entering power down, can be different value per chip select.
   - SLOW_PD
   - BOTH_CS_PS - The MMDC can either set each chip select independently to power down, according to its idle state, or set both chip selects to power down only if both in idle state for the configured period.
   - Few parameters must be configured in addition:
     - Timing parameters at MDCFG0[tXP and tXPDLL].
     - ODT timing at MDOTC[tAOFPD, tAONPD, tANPD and tAXPD]

NOTE

It is possible to enter certain chip selects to low power consumption while the second chip select is activated.

3. Automatic precharge of all DDR banks to a specific chip select. Can be activated by configuring ESDPDC fields: PRCT_0 and PRCT_1. Each field determines a value loaded to a different chip select.

22.5.5.2 Self refresh and Frequency change entry/exit

As described in Power saving general, the MMDC supports two mechanisms that will cause the DDR device to enter self-refresh mode:

- LPMD (Low Power Mode) - For power saving purposes
- DVFS (Dynamic Voltage and Frequency Change) - For clock frequency changes

While the DDR device is in self-refresh mode, there is no need to provide periodic refresh commands.

The MMDC treats hardware/software handshaking of LPMD/DVFS in the same manner:

- Upon the assertion of LPMD/DVFS request, the following is done:
  - The MMDC blocks any further AXI accesses even before the acknowledge is asserted
  - Completes all opened AXI accesses
  - Closes (precharge) all banks in the appropriate timing
  - Drives self-refresh command by deasserting clock enable signal (DRAM_SDCKE is driven to "0") together with a refresh command. This occurs after satisfying tRP/tRPA from the Precharge All command.
  - Deasserts the clock (CK) that is driven to the DDR device
  - Asserts LPMD/DVFS acknowledge (LPACK/DVACK)
  - Allows deassertion of the operating clock of the MMDC (AXI clock)
- Upon the deassertion of LPMD/DVFS request, the following is done:
  - Operating clock of the MMDC must be turned on before LPMD/DVFS is deasserted
  - Starts driving the clock (CK) to the DDR device
  - After satisfying tCKSRX from clock renewal the clock enable signal (DRAM_SDCKE) is asserted
  - LPMD/DVFS acknowledge (LPACK/DVACK) is deasserted
  - After satisfying tXS from the assertion of DRAM_SDCKE, a refresh command is driven to the DDR device.
  - If ZQ calibration is enabled then tRFC is satisfied from the refresh command and a long ZQ command is driven.
  - tZQoper idle cycles are counted after the ZQ command.
  - After satisfying tDLLK from the assertion DRAM_SDCKE, the MMDC returns to normal operation.

The figure below shows the timing diagram of the hardware/software handshaking of LPMD/DVFS:

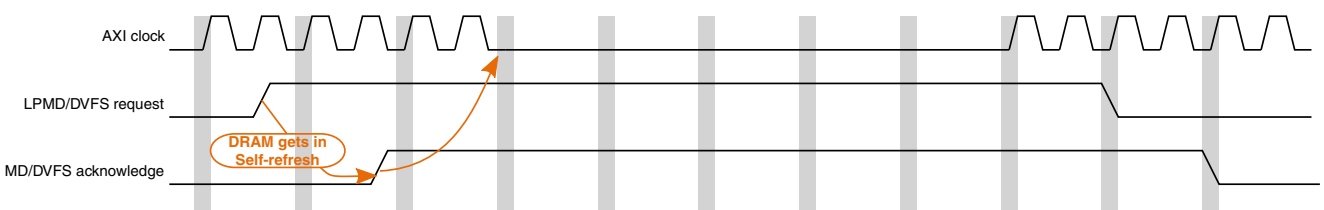


Figure 22-4. LPMD/DVFS Hardware/Software Handshaking

Note for self-refresh:

- As soon as LPMD or DVFS requests are detected by either hardware or software handshaking, the MMDC will deassert the AXI ARREADY/AWREADY signals immediately to block further requests from the system.
- In case of automatic self-refresh, the internal operating clock will be negated to save power.

22.5.6 Reset

22.5.6.1 Hard reset

When hard reset is asserted (aresetn is driven to "0") while warm reset is deasserted (warm_reset is driven to "0"), the entire MMDC will be initialized, including configuration/status registers and state machines.

In order to access the DDR device, the MMDC will then have to be reconfigured.

22.5.6.2 Warm reset

The MMDC supports warm reset signal. The warm reset signal must envelop the hard reset signal and then the MMDC will reset all the internal registers. The only registers that are not reset are those that are essential for returning it to normal operation without repeating the initialization sequence and without losing data stored in the memory (configuration/status registers won't be initialized).

For the successful operation of warm reset, the following steps must be performed:

- The MMDC must enter self-refresh mode. This can be achieved by either LPMD or DFVS requests
- Wait for LPMD or DVFS acknowledge
- Assert warm reset signal (i.e. drive warm_reset to "1")
- Assert hard reset signal (i.e. drive aresetn to "0")
- Deassert hard reset signal
- Deassert warm reset
- Get out of the LPMD/DVFS mode

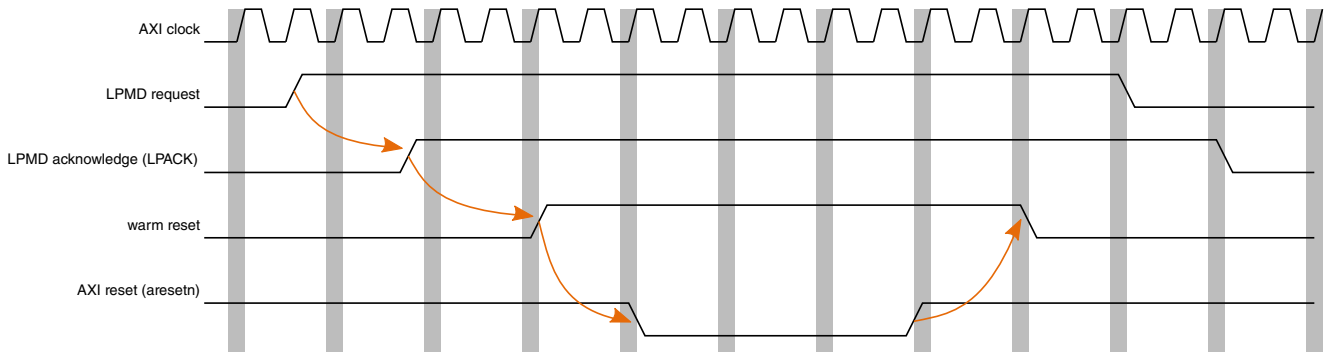


Figure 22-5. Warm Reset Diagram

22.5.6.3 Software reset

The MMDC supports software reset. When software reset is configured then the MMDC will reset all the internal registers except those that are essential for returning to normal operation without repeating the initialization sequence or without losing data stored in the memory (configuration/status registers won't be initialized).

The following steps should be performed for successful operation of software reset:

- The MMDC should enter self-refresh mode. This can be achieved by either LPMD or DFVS request.
- Wait for LPMD or DVFS acknowledge
- Assert software reset, by setting MDMISC[RST]
- Get out of the LPMD/DVFS mode

Normal operation can be resumed.

22.5.7 Refresh Scheme

The MMDC supports various automatic refresh options which can be configured via the MDREF register.

The periodic auto refresh can be triggered by the following clocks:

- 32 kHz clock
- 64 kHz clock
- MMDC operating clock

The refresh scheme of the MMDC is flexible and allows the system to configure the desired AXI accesses delay/latency in each refresh cycle.

The table below shows an example of four configurations of the refresh cycles that will be handled by the MMDC. Each configuration meets a refresh rate of 3.9µs (tREFI, refresh command every 3.9µs).

Table 22-8. MMDC Refresh Scheme

Option number	Description	REFR	REF_SEL	REF_CNT	DDR hang time
1	Issue 8 refresh commands every 31,250 ns	0x7 (8 refreshes)	0x0 (64 kHz)	not needed	tRFC x 8
2	Issue 4 refresh commands every 15,625ns	0x3 (4 refreshes)	0x1(32 kHz)	not needed	tRFC x 4
3	Issue 2 refresh commands every 7800ns	0x1(2 refreshes)	0x2 (REF_CNT)	7800/2.5 = 3120 (0xC30)	tRFC x 2
4	Issue 1 refresh command every 3900 ns	0x0 (1 refresh)	0x2 (REF_CNT)	3900/2.5 = 1560(0x618)	tRFC

22.5.8 Burst Length options towards DDR

The MMDC supports burst lengths which can be configured through MD+CTL[BL] as follows:

- In LPDDR2 mode, only burst length 4 can be used.
- In LPDDR3 mode, only burst length 8 can be used.

In case of AXI INCREMENT, accesses that are not aligned the irrelevant data is masked in write accesses and ignored in read accesses. In case of AXI WRAP accesses, even if the access is not aligned, then the MMDC provides an internal optimization mechanism for better efficiency of the DDR data bus.

22.5.9 Exclusive accesses handling

The MMDC contains four exclusive monitors, each for dedicated ID as configured in MAEXIDR0 and MAEXIDR1.

- If legal read exclusive is received by the MMDC, the associated monitor is turned on.
- While the monitor is turned on upon legal write exclusive, the monitor will be turned off and the write will be completed successfully with EXOKAY.
- The following rules must be met for successful exclusive access:
  - Aligned access (the AXI address is aligned to the AXI size)
  - AXI single access (AXI burst length isn't greater than 1)

- AXI size of up to 64 bits
- AXI non-cachable access (i.e. ARCACHE[1]/AWCACHE[1] is equal "0" or ARCACHE[1]/AWCACHE[1] is equal "1" while ARCACHE[3:2]/ AWCACHE[3:2] are equal "00")
- AXI ID that matches one of the four exclusive IDs

Exclusive read behavior (first bullet also correct for non-exclusive accesses):

- In case of security violation, the read is blocked and is not sent to DDR. There are two options for response:
  - If ARCR_SEC_ERR_EN (MAARCR[30]) is high, SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.
- If AXI exclusive rules violation occurs (as described above), the read access is not blocked and is sent to DDR. The data will be fetched and be driven to the master, but the type of response may be unpredicted.
- If none of the above occurs, the read is sent to the DDR. The exclusive monitor will be turned on and the response is EXOKAY
- If additional legal AXI read exclusive is received with the same ID before the AXI exclusive write, the monitor will be updated with the latest attributes.

Exclusive write behavior (first bullet also correct for non-exclusive accesses):

- In case of security violation, the write is blocked and is not sent to DDR, but the monitor will be kept on. There are two options for response:
  - If ARCR_SEC_ERR_EN (MAARCR[30]) is high then SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.
- In case of AXI exclusive rules violation (as described above), the write is blocked and is not sent to DDR. In that case the type of response may be unpredicted.
- In case the exclusive write access has different AXI attributes, but the same ID as the read exclusive access, the write is blocked and is not sent to DDR and the monitor will be turned off. There are two options for response:
  - If ARCR_EXC_ERR_EN (MAARCR[28]) is high then SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.
- In case of regular (non exclusive) write access is received to the same address or overlapping addresses then the write will be sent to the DDR and the monitor will be turned off.
- In case of legal write exclusive access is received with the same attributes as the read exclusive access while the monitor is on (no write accesses occured to the same address between the read exclusive and write exclusive), then the write is sent to DDR and the response is EXOKAY. But, if the legal write exclusive is received while the monitor is off, the write is blocked and there are two options for response.
  - If ARCR_EXC_ERR_EN (MAARCR[28]) is high then SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.

22.5.10 AXI Error Handling

The MMDC supports the AXI responses listed here.

- In case of AXI exclusive violation there are two options for response:
  - If MAARCR[28] is high then SLV Error is issued towards the Master, Otherwise OKAY response is sent to the Master

NOTE

In case of read error MMDC drives zeros on the read data bus

22.6 Performance

22.6.1 Arbitration and reordering mechanism

22.6.1.1 Arbitration General

The following specifies arbitration and reordering flow in MMDC towards the DDR.

- AXI read and write accesses are sampled in the associated queue.
- Read/write arbitration is handled to select the winning access.
- Winning access is sampled in the reordering queue
- Reordering mechanism is handled between valid requests that reside in the reordering queue to select the access that will be dispatched to the DDR.
  - The reordering is held in order to optimize the accesses and to maximize the utilization of the DDR bus
  - As soon as the reordered access is completed (indicated by end of response or data phase) then it is erased from the associated queue and the MMDC is ready to receive the next available access from the master

In general, the reordering/arbitration mechanism is based on dynamic priority mechanism, which compares dynamic priorities between valid entries in the reordering queue and issues the entry with highest dynamic priority towards the DDR Logic.

The selection of the winning access is based on two modes, which can be activated together, as following:

- Real time channel mode:

    - Accesses with QoS='f' (i.e. awqos[3:0]/arqos[3:0] = "f") will bypass all other requests towards the DDR
- Dynamic scoring mode:
    - The arbitration mechanism is based on dynamic priority. Relevant for the accesses with QoS smaller than 'f' or when real time channel mode is disabled.

NOTE

Due to re-ordering and optimization mechanism (per different AXI ID), the transactions towards the DDR may be driven in a different ID order they were received by the AXI master. In similar way, the write response, read response or read data may be driven to the AXI master in a different ID order.

22.6.1.2 Real time channel mode

When real time mode is enabled (i.e MAARCR[ARCR_RCH_EN] = "1") , all requests with QoS='f' (i.e. awqos[3:0]/aqqos[3:0] = "f") will bypass all other pending accesses towards the DDR. This mode is enabled by default.

22.6.1.3 Dynamic scoring mode (Arbitration Winning Conditions)

The arbitration between pending accesses in the MMDC is handled according to a dynamic priority of each access.

The dynamic priority (may be also called score) is calculated according to a sum of some factors (final_score[3:0]), where part of them may be updated dynamically. The following will specify each scoring factor:

- MAARCR[ARCR_PAG_HIT] (Page hit score) - A static score which is taken into account in case the pending access has a page hit
- MAARCR[ARCR_ACC_HIT] (Access hit score) - A static score, which is taken into account in case the current access type (read/write) is the same as the access that has been dispatched to the DDR previously
- MAARCR[ARCR_DYN_JMP] (Dynamic jump score) - A dynamic score which is given to any pending access in case it was not chosen in the arbitration. The dynamic jump counter is limited by maximum value which is set in MAARCR[ARCR_DYN_MAX] .
- QoS score which is indicated through a sideband 4bits AXI signals (awqos[3:0]/ aqqos[3:0]) and is driven by the AXI master per access

NOTE

In order to prevent an overflow in the total sum of scores, a clipping is held and selects the maximum score value of 'f' once a total scores sum is greater than 'f'.

The figure below shows the dynamic score calculations

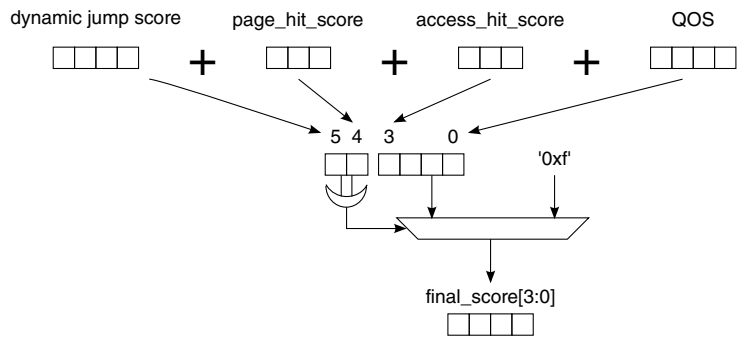


Figure 22-6. Dynamic score/priority calculation

22.6.1.4 Guarding (aging) mechanism

The guarding mechanism (may be also called aging) is used to prevent a starvation of accesses.

As soon as the dynamic jump score reaches its maximum value (MAARCR[ARCR_DYN_MAX]) then each time a pending request was not chosen in the arbitration, the "guarding" counter is incremented by 1. When the "guarding" counter reaches its predefined value, set in MAARCR[ARCR_GUARD], the associated request gets the highest priority and will be chosen in the next arbitration cycle towards the DDR unless a real time channel (i.e access with QoS ="f") is arrived.

Note: In case real time channel has arrived then the dynamic score of the non real time channels won't increment in order to prevent a case where the "guarding" counter of more than one access has reached its limit.

22.6.2 Prediction mechanism

When prediction mechanism is enabled (i.e by configuring MDMISC[MIF3_MODE]) then the MMDC predicts the chip-select, bank address and row address that is going to be issued towards the DDR before the access is physically dispatched towards DDR device.

That mechanism enables to prepare the DDR device with future accesses and improves the overall DDR performance.

This prediction mechanism operates in parallel to the reordering mechanism and may yield a prediction based on 3 levels of pending accesses:

1. Access in first stage of pipeline.
2. Valid access on AXI bus either read channel or write channel.
3. Valid access on special bus from arbitration - this access is chosen by the arbitration as the next miss access in its buffers

22.7 MMDC Debug

22.7.1 Hardware debug monitor

The MMDC has a hardware debugging mechanism that monitors each access that is driven to the MMDC. Every time this mechanism is enabled (setting of MADPCR0[DBG_EN] to "1") then each access that will be dispatched to the DDR will be also observed in the I/O pads (i.e. over ipp_do_ddr_debug[50:0]). The content of this bus is described in the table below.

Table 22-9. Hardware monitor debugging

Signal Name	Number of Bits	Description
acc_addr	[31:0]	AXI ADDRESS of the selected access
acc_type	1	access type of the selected access. "0 indicates write. "1" indicates read.
acc_id	[15:0]	AXI transaction ID of the selected access
valid_strobe	1	indication for a valid request. This signal will be asserted for 1 clock cycle

The fields above are organized as following:

MMDC_DEBUG[50:0] = {1'b0,valid_strobe,acc_id,access_type,addr}

These signals are sent to IOMUX, in IOMUX user can configure it to be output from the chip for debug usage.

22.7.2 Step By Step (SBS) software monitor

The MMDC has a Step By Step (SBS) software debugging mechanism that monitors each access that is driven to the MMDC. Every time this mechanism is trigerred then one AXI access will be dispatched to the DDR and in parallel its attributes will be observed in a status register.

Once the "step by step" is enabled (i.e. MADPCR0[SBS_EN] is "1") then all accesses to the DDR device will be halted. This SBS feature is used during DDR SDCLK frequency scaling to prevent any accesses to the DDR device during the transition. Setting MADPCR0[SBS] to "1" will dispatch the access that is pending in the head of the MMDC queue (read or write). Upon every setting of MADPCR0[SBS]:

- The AXI attributes of the access will be sampled in the associated MASBS0 and MASBS1 fields
- MADPCR0[SBS] will be cleared automatically.

Setting again MADPCR0[SBS] to "1" will dispatch the next pending access in the MMDC queue.

NOTE

Setting the ARCR_ARB_REO_DIS bit in the MMDC0_MAARCR Register disables the SBS function

22.8 MMDC Profiling

The profiling mechanism provides the ability to calculate the DDR utilization together with read and write accesses statistics towards DDR per given period of time.

MMDC supports the following profiling counters:

- MADPSR0 (Total cycles count) - Indicates the total amount of cycles of the profiling period (up to 2^32 cycles)
- MADPSR1 (Busy cycles count) - Indicates the total busy cycles during the profiling period. Busy cycles are any MMDC clock cycles where the internal state machine is not idle. If any read or write requests are pending in the FIFOs, the MMDC is not idle.
- MADPSR2 (Total read accesses count) - Indicates the total read accesses towards MMDC during the profiling period
- MADPSR3 (Total write accesses count) - Indicates the total write accesses towards MMDC during the profiling period
- MADPSR4 (Total read bytes count) - Indicates total bytes that were read from MMDC during the profiling period
- MADPSR5 (Total write bytes count) - Indicates total bytes that were written to MMDC during the profiling period

All profiling items described above are disabled by default. The following describes how to control the profiling mechanism:

- MADPCR0[DBG_EN] enables profiling.

- MADPCR0[PRF_FRZ] stops/freezes the profiling for example in case user wishes to perform DDR profiling per specific task. In order to resume profiling then MADPCR0[PRF_FRZ] should be cleared.
- MADPCR0[DBG_RST] clears all profiling counters
- MADPCR0[CYC_OVF] indicates whether an overflow occured in the total cycles counter (i.e. total amount of cycles are greater than 2^32) . This field can only cleared by writing '0'.

Read/Write statistics can be collected per specific AXI ID (16bits). The following fields in MADPCR1 register determines which AXI-ID or AXI-ID's to monitor:

- PRF_AXI_ID defines which AXI IDs are taken for profiling. Default values is 16'h0.
- PRF_AXI_ID_MASK defines which bits from PRF_AXI_ID will be compared with AXI ID of read/write access. "1" means to monitor the associated bit and "0" means don't care. Default value is 16'h0000, meaning all IDs are not monitored

So the AXI-IDs to be monitored are calculated according to the following equation:

(AXI-ID & PRF_AXI_ID_MASK) Xnor (PRF_AXI_ID & PRF_AXI_ID_MASK)

For example if AXI ID's between A100 till A1FF are wished to be monitored then the following should be configured:

- PRF_AXI_ID = A100
- PRF_AXI_ID_MASK = FF00

22.9 LPDDR2/3 Refresh Rate Update and Timing Derating

LPDDR2 devices may have a temperature sensor that is used to determine an appropriate refresh rate and whether AC timing derating is required. The status of the temperature sensor can be read through MRR command from LPDDR2 MR4 register.

The MMDC supports refresh update and timing derating mechanism on the fly. The following steps specify how to use that mechanism:

- Perform periodic polling on MR4 LPDDR2 register using MRR command
- Read MDMRR register and analyze the MR4 indication
- In case refresh rate update and/or AC timing derating is required then it is needed to update MDREF and/or MDMR4[tRCD_DE, tRC_DE, tRAS_DE, tRP_DE, tRRD_DE] parameters

NOTE

MDMR4[tRCD_DE, tRC_DE, tRAS_DE, tRP_DE, tRRD_DE] are referred to the associated values configured at MDCFG3LP[tRC_LP, tRP_LP, tRCD_LP], MDCFG1[tRAS], MDCFG2[tRRD]

- Assert MDMR4[UPDATE_DE_REQ]
- When the MMDC switch to the new values then an acknowledge will be indicated at MDMR4[UPDATE_DE_ACK]

The enabled power saving features may be interfering with the MR4 reads. When these features are enabled the MMDC will automatically enter self-refresh and hence prevent the MR4 reads. Hence the Power Saving and refresh functionality should be disabled when SW performs the MR4 reads.

To disable the “Power Down” mode, set bit 0 of MAPSR to 1’b.

To disable the “Self-Refresh Power Down Timer” mode, set bits [15:8] of MDPDC to 0x00.

Overall Sequence to Perform MR4 reads

1. Set MAPSR to 0x00010107
2. Set MDPDC to 0x00020076
3. MRR command to read MR4
4. Write MDPDC to 0x00025576 to re-enable this feature
5. Write MAPSR to 0x00010106 to re-enable this feature

22.10 Calibration Process

The MMDC offers various calibration processes that are used to obtain better timing accuracy, board skew compensation and I/O pad driving strength adjustment. Each calibration process can be performed either automatically (hardware) or manually (software), though the manual method is typically reserved for debugging purposes. The following calibration processes are supported:

NOTE

Power saving features must be disabled before beginning the write leveling, DQS gating, and read/write calibration processes. ZQ calibration is allowed with power saving features enabled.

- ZQ calibration for external DDR device (in LPDDR2/LPDDR3 through MRW command)

    - Can be handled automatically for ZQ Short (periodically) and ZQ Long (at exit from self-refresh)
    - Can be handled manually at ZQ INIT
- ZQ calibration for i.MX DDR I/O pads for calibrating the DDR driving strength
    - The sequence can be handled automatically by hardware
    - The sequence can be handled step by step manually by software
- Read DQS gating calibration not supported for LPDDR2/LPDDR3
- Read data calibration. Adjustment of read DQS with read data byte. For further information refer to Read Calibration
- Write data calibration. Adjustment of write DQS with write data byte. For further information refer to Write Calibration
- Read fine tuning. Adjustment of up to 7 delay-line units for each read data bit.
- Write fine tuning. Adjustment of up to 3 delay-line units for each read data bit.

- Disable the periodic refresh scheme (i.e. setting MDREF[REF_SEL] = "11") and then issue manual refresh command burst by configuring MDSCR[CMD]= 0x2. At the end of the calibration it is needed to enable the periodic refresh scheme.
- Disable the automatic power saving mode (i.e set MAPSR[PSD] = "1").

22.10.1 Delay-line

Each of the calibration processes controls several delay-lines for aligning data and strobes.

By default the delay-line is configured to generate 1/4 clock cycle of delay.

Moreover, when the operating clock is at the maximum allowed frequency, as appeared in the features list, then the delay-line is capable to issue a configurable delay of up to 1/2 clock cycle.

NOTE

At the beginning of the calibration process the initial value of the delay-line must be a valid value (i.e. the strobes must be somewhere among the associated data window) though it might not be the optimal value. The delay-line calibration should be done after Read DQS gating.

In order to generate an adequate delay during normal operation of the MMDC the delay-line is going through an automatic measurement process during the refresh period of the DDR device

22.10.2 ZQ calibration

The MMDC supports ZQ calibration process to calibrate the driving strength of the DDR I/O pads as well as driving ZQ commands to calibrate the external DDR device driving strength.

The first ZQ calibration (after booting the processor) is performed prior to turning on the MMDC. Subsequent ZQ calibrations may be executed in parallel to the DDR ZQ calibration. The MMDC supports 2 types of ZQ calibration commands: short and long.

The ZQ long calibration is executed during power up sequence, when exiting self-refresh mode or when exiting slow precharge power down (DLL lock can be done in parallel). The ZQ short calibration is executed periodically according to a configurable timer defined by MPZQHWCTRL[ZQ_HW_PER].

The field MPZQHWCTRL[ZQ_MODE] determines whether the MMDC will execute ZQ calibration to DDR I/O pads and/or issue ZQ short/long command to the DDR device.

The MMDC supports both automatic (hardware) and manual (software) ZQ calibration process for the DDR I/O pads.

It is possible to perform automatic (hardware) ZQ calibration only once (i.e. non-periodical) by asserting MPZQHWCTRL[ZQ_HW_FOR].

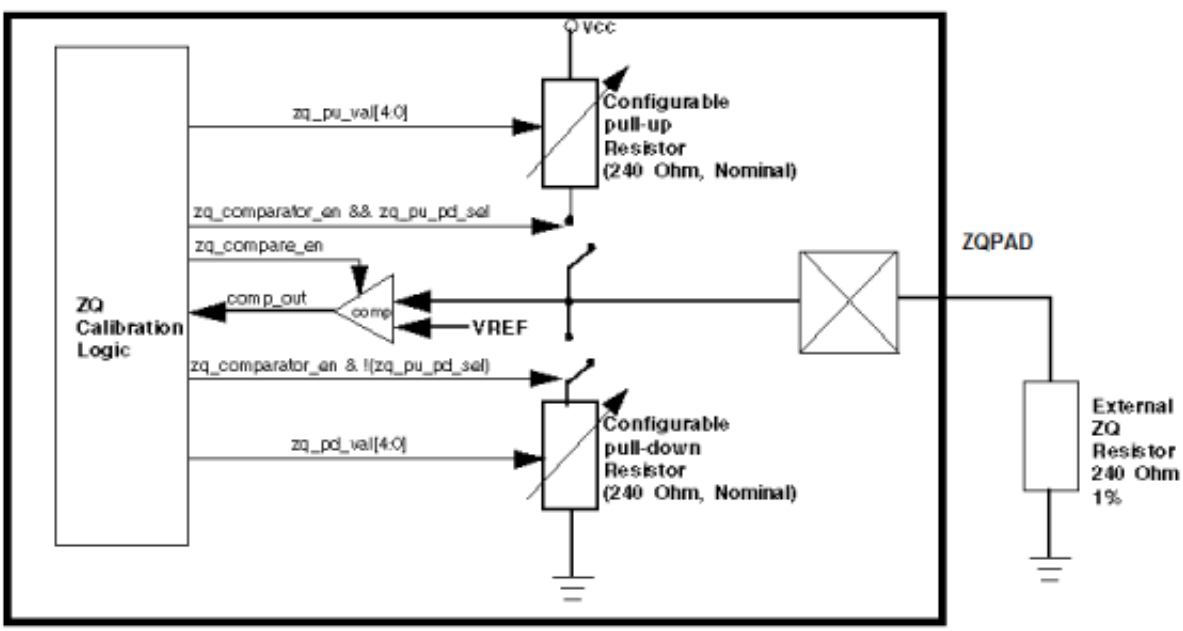


Figure 22-7. MMDC ZQ IF with PAD

22.10.2.1 ZQ automatic (hardware) calibration process

The ZQ automatic calibration occurs over multiple steps isolated by pull-up resistor calibration and by pull-down resistor calibration as follows.

22.10.2.1.1 ZQ automatic Pull-up calibration

The MMDC automatically performs a handshaking mechanism with the ZQ calibration pad as follows:

1. The MMDC drives zq_comparator_en to "1"
2. The MMDC waits few cycles according to MPZQHWCTRL[ZQ_EARLY_COMPARATOR_EN_TIMER]
3. The MMDC drives zq_pu_pd_sel to "1" for indication of pull-up calibration and drives zq_pu_val[4:0] = 5'b00000
4. MMDC drives zq_pu_val[4] to "1"
5. MMDC asserts zq_compare_en
6. MMDC waits few cycles according to MPZQSWCTRL[ZQ_CMP_OUT_SMP] before sampling the comparator output (i.e zq_comp_out). If zq_comp_out is "1" then it means that the output voltage is greater than Vdd/2 (i.e. internal resistor is less than 240 ohm) and drives bit zq_pu_val[4] to "1" else it drives zq_pu_val[4] to "0"
7. MMDC deasserts zq_compare_en
8. MMDC repeats steps 4-7 for zq_pu_val bits 3 to 0
9. MMDC drives ZQ calibration result to MPZQHWCTRL[ZQ_HW_PU_RES]
10. MMDC advances to pull-down calibration

22.10.2.1.2 ZQ automatic Pull-down calibration

1. The MMDC drives zq_pu_pd_sel to "0" for indication of pull-down calibration and drives zq_pd_val[4:0] = 5'b00000
2. MMDC drives zq_pd_val[4] to "1"
3. MMDC asserts zq_compare_en
4. MMDC waits few cycles according to MPZQSWCTRL[ZQ_CMP_OUT_SMP] before sampling the comparator output (i.e zq_comp_out). If zq_comp_out is "1" then it means that the output voltage is greater than Vdd/2 (i.e. internal resistor is less than 240 ohm) and drives bit zq_pd_val[4] to "0" else it drives zq_pd_val[4] to "1"
5. MMDC deasserts zq_compare_en
6. MMDC repeats steps 2-5 for zq_pd_val bits 3 to 0
7. MMDC drives ZQ calibration result to MPZQHWCTRL[ZQ_HW_PD_RES]
8. MMDC deassert zq_comparator_en to indicate the completion of the ZQ calibration

22.10.2.2 ZQ software calibration process

The ZQ calibration can be done also in software. However since software ZQ calibration is much slower than hardware calibration it should be used mainly for debugging.

Software should configure the ZQ calibration parameters (Pull-up or Pull-down and their value) then assert the MPZQSWCTRL[ZQ_SW_FOR] bit. Then software should wait till ZQ_SW_FOR is de-asserted and use ZQ_SW_RES status bit in order to calculate the next ZQ calibration parameters.

22.10.2.3 ZQ calibration commands

Before the MMDC can issue a ZQCL/ZQCS command to the memory it should precharge all memory banks and wait tRP period. A single ZQ command can be issued to all devices as long as the devices don't share the same ZQ resistor.

When the MMDC issues the ZQ command it should also drive A10 (long or short command) and CS (0, 1 or both).

The MMDC must keep the memory lines quiet (except for CK) for the ZQ calibration time as defined in the Jedec (512 cycles for ZQCL after reset, 256 for other ZQCL and 64 for ZQCS).

22.10.3 Read Calibration

The read calibration is used to adjust the read DQS with read data byte. It is assumed that the read DQS gating calibration process is completed prior to the read calibration.

NOTE

In LP2/3_x16, LP2/3_x32 the activation of the calibration is done by setting MPRDDLHWCTL[HW_RD_DL_EN].

22.10.3.1 Hardware (automatic) Read Calibration

There are two modes of operations:

- Calibration with the MPR (Multi Purpose Register)/DQ calibration(LPDDR2/ LPDDR3)
- Calibration with MMDC pre-defined values

22.10.3.1.1 Hardware (automatic) Calibration with DQ Calibration

Execute the following steps:

1. Precharge all active banks (can be done through MDSCR) as required.
2. Enter the DDR device into DQ calibration mode through MRS/MRW commands.
3. Configure the MMDC to work with DQ calibration mode by asserting MPPDCMPR2[MPR_CMP].
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (MPRDDLCTL[RD_DL_ABS_OFFSET#]) will place the read DQS somewhere inside the read DQ window.
5. Start the calibration process by asserting MPRDDLHWCTL[HW_RD_DL_EN].

22.10.3.1.2 Hardware (automatic) Calibration with pre-defined value

In case pre-defined mode is used, i.e. MPPDCMPR2[MPR_CMP] is cleared, then the following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as required.
2. Configure the pre-defined value, which reflects the value that will be written and compared through the read calibration, to MPPDCMPR1[PDV1, PDV2]
3. Issue write access to the external DDR device by setting MPSWDAR0[SW_DUMMY_WR] = 1 (MMDC will generate internally write access without intervention of the system towards bank 0, row 0, column 0)
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. MPRDDLCTL[RD_DL_ABS_OFFSETn]) will place the read DQS somewhere inside the read DQ window
5. Start the calibration process by asserting MPRDDLHWCTL[HW_RD_DL_EN]

The following steps will be executed automatically by the MMDC for both modes (MPR and Pre-defined value):

1. MMDC waits till the read delay-line is updated with the absolute delay value for all bytes at MPRDDLCTL[RD_DL_ABS_OFFSETn] and also satisfying the Tmod + 4 requirement
2. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to MPRDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device.
3. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial read DQS isn't inside the read DQ window and the MMDC generates an error for the associated byte at MPRDDLHWCTL[HW_RD_DL_ERRn] . If the comparison passes then MMDC advances to next step.
4. MMDC resets the rd fifo (to the inverted pre-defined/MPR value) and it's pointers by setting MPDGCTRL[RST_RD_FIFO] = 1

5. MMDC decrements the read delay line absolute offset of each byte by 1 (i.e. MPRDDLCTL[RD_DL_ABS_OFFSETn]) and issue measurement process of the read delay-line to update itself with the new value.
6. MMDC drives read command to the DDR external devices and waits 16 or 32 cycles (according to MPRDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device
7. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the low read boundary of the associated byte for each byte at MPRDDLHWST0/1[HW_RD_DL_LOWn] . If the comparison passes then MMDC repeats steps 4-6. If all read data comparisons fail then the MMDC advances to the next step
8. The MMDC start seeking the upper boundary and sets the read delay line absolute offset of each byte to the initial value + 1 as determined at step 4 and issue measurement process of the read delay-line to update itself with the new value
9. MMDC resets the rd fifo (to the inverted pre-defined value) and it's pointers by setting MPDGCTRL[RST_RD_FIFO] = 1
10. MMDC drives read command to the DDR external devices and waits 16 or 32 cycles (according to MPRDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device
11. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the upper read boundary of the associated byte for each byte at MPRDDLHWST0/1[HW_RD_DL_UPn] . If the comparison passes then MMDC increments the read delay line absolute offset of each byte by 1 (i.e. MPRDDLCTL[RD_DL_ABS_OFFSETn]) and issue measurement process of the read delay-line to update itself with the new value.
12. If all read data comparisons fail then the MMDC advances to the next step. otherwise, MMDC repeats steps 9-11.
13. After the MMDC finds the window boundary (lower and upper) of each read data byte then it stores the average between lower and upper boundaries at the associated MPRDDLCTL[RD_DL_ABS_OFFSETn] and issue measurement process of the read delay-line to update itself with the new value.
14. MMDC indicates that the read data calibration had finished by setting MPRDDLHWCTL[HW_RD_DL_EN] = 0
15. Exit the DDR device from MPR/DQ calibration mode through MRS/MRW commands

22.10.3.2 SW Read Calibration

There are two modes of operations:

- Calibration with the MPR (Multi Purpose Register)/DQ calibration (LPDDR2/ LPDDR3)
- Calibration with MMDC pre-defined values

22.10.3.2.1 Calibration with DQ calibration

Execute the following steps:

1. Precharge all active banks (Can be done through MDSCR) as required.
2. Enter the DDR device into DQ calibration mode through MRS/MRW commands
3. Configure the MMDC to work with DQ calibration mode by asserting MPPDCMPR2[MPR_CMP]
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. MPRDDLCTL[RD_DL_ABS_OFFSETn]) will place the read DQS somewhere inside the read DQ window

22.10.3.2.2 Calibration with pre-defined value

In case pre-defined mode is used, i.e. MPPDCMPR2[MPR_CMP] is cleared, then the following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as requried by the standard.
2. Configure the pre-defined value, which reflects the value that will be written and compared through the read calibration, to MPPDCMPR1[PDV1, PDV2]
3. Issue write access (with any legal DDR address) to external DDR device.
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. MPRDDLCTL[RD_DL_ABS_OFFSETn]) will place the read DQS somewhere inside the read DQ window

The following steps will be executed manually by SW for both modes (MPR/DQ calibration and Pre-defined value):

1. Force the delay line to measure itself and to issue the requested read delay by configuring MPMUR[FRC_MSR] = 1
2. Wait 16 DDR cycles till the read delay-line is updated with the absolute delay value for all bytes
3. Issue read command (with any legal DDR address) from the external DDR device
4. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial read DQS isn't inside the read DQ window. If the comparison passes then advance to next step.
5. Reset the rd fifo (to the inverted pre-defined/MPR value) and it's pointers by setting MPDGCTRL[RST_RD_FIFO] = 1

6. Decrement the read delay line absolute offset of each byte by 1 (i.e. MPRDDLCTL[RD_DL_ABS_OFFSETn])
7. Force the delay line to measure itself and to issue the requested read delay by configuring MPMUR[FRC_MSR] = 1
8. Issue read command (with the legal DDR address chosen in step 7) from the external DDR device and waits 16 or 32 cycles (according to MPRDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device
9. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the low read boundary of the associated byte at of each byte . If the comparison passes then repeat steps 5-8. If all read data comparisons fail then advance to the next step.
10. Start seeking the upper boundary and set the read delay line absolute offset of each byte to the initial value + 1 as determined at step 4
11. Force the delay line to measure itself and to issue the requested read delay by configuring MPMUR[FRC_MSR] = 1
12. Resets the rd fifo (to the inverted pre-defined/MPR value) and it's pointers by setting MPDGCTRL[RST_RD_FIFO] = 1
13. Issue read command (with the legal DDR address chosen in step 7) from the external DDR device
14. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the upper read boundary of the associated byte at of each byte. If the comparison passes then increment the read delay line absolute offset of each byte by 1 (i.e. MPRDDLCTL[RD_DL_ABS_OFFSETn])
15. Force the delay line to measure itself and to issue the requested read delay by configuring MPMUR[FRC_MSR] = 1
16. If all read data comparisons fail then advance to the next step, else repeat steps 12-15.
17. After finding the window boundary (lower and upper) of each read data byte then calculate the average between lower and upper boundaries and store the associated average at MPRDDLCTL[RD_DL_ABS_OFFSETn]
18. Force the delay line to measure itself and to issue the requested read delay by configuring MPMUR[FRC_MSR] = 1
19. Exit the DDR device from MPR/DQ calibration mode through MRS/MRW commands.

22.10.4 Write Calibration

The write calibration is used to adjust the write DQS with write data byte. It is assumed that the read calibration process is completed prior to the write calibration.

NOTE

In LP2/3_x16, LP2/3_x32 the activation of the calibration of each channel is done by setting MPWRDLHWCTL0[HW_WR_DL_EN].

22.10.4.1 HW (automatic) Write Calibration

The following steps should be executed:

1. Make sure that the initial value that is configured in the write delay line absolute offset of each byte (i.e. MPWRDLCTL[WR_DL_ABS_OFFSET#]) will place the write DQS somewhere inside the write DQ window
2. Configure the pre-defined value, which reflects the value that will be written and compared through the write calibration, to MPPDCMPR1[PDV1, PDV2]
3. Assert MPWRDLHWCTL0[HW_WR_DL_EN]

The following steps will be executed automatically:

1. MMDC waits till the write delay-line is updated with the absolute delay value for all bytes at MPWRDCTL[WR_DL_ABS_OFFSET#]
2. MMDC drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to MPWRDLHWCTL[HW_WR_DL_CMP_CYC]) assuming that the data has arrived to the DDR device.
3. MMDC drives read command to the same address from the external DDR
4. MMDC compares the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial write DQS isn't inside the write DQ window and the MMDC generates an error for the associated byte at MPWRDLHWCTL[HW_WR_DL_ERR#] . If the comparison passes then MMDC advances to next step.
5. MMDC resets the rd fifo (to the inverted pre-defined value) and it's pointers by setting MPDGCTRL[RST_RD_FIFO] = 1
6. MMDC decrements the write delay line absolute offset of each byte by 1 (i.e. MPWRDLCTL[WR_DL_ABS_OFFSET#]) and issue measurement process of the write delay-line to update itself with the new value.
7. MMDC drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to MPWRDLHWCTL[HW_WR_DL_CMP_CYC]) assuming that the data has arrived to the DDR device

8. MMDC drives read command to the same address from the external DDR
9. MMDC compares the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the low write boundary of the associated byte of each byte at MPWRDLHWST0/1[HW_WR_DL_LOW#] . If the comparison passes then MMDC repeats steps 8-11. If all data comparisons fail then the MMDC advances to the next step
10. The MMDC start seeking the upper boundary and sets the write delay line absolute offset of each byte to the initial value + 1 as determined at step 4 and issue measurement process of the write delay-line to update itself with the new value
11. MMDC resets the rd fifo (to the inverted pre-defined value) and its pointers by setting MPDGCTRL[RST_RD_FIFO] = 1
12. MMDC drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to MPWRDLHWCTL[HW_WR_DL_CMP_CYC]) assuming that the data has arrived to the DDR device.
13. MMDC drives read command to the same address from the external DDR
14. MMDC compares the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the upper write boundary of the associated byte of each byte at MPWRDLHWST0/1[HW_WR_DL_UP#] . If the comparison passes then MMDC increments the write delay line absolute offset of each byte by 1 (i.e. MPWRDLCTL[WR_DL_ABS_OFFSET#]) and issue measurement process of the write delay-line to update itself with the new value.
15. MMDC repeats steps 14-17. If all data comparisons fail then the MMDC advances to the next step
16. .After the MMDC finds the window boundary (lower and upper) of each write data byte then it stores the average between lower and upper boundaries at the associated MPWRDLCTL[WR_DL_ABS_OFFSET#] and issue measurement process of the write delay-line to update itself with the new value.
17. MMDC indicates that the write data calibration had finished by setting MPWRDLHWCTL[HW_WR_DL_EN] = 0

22.10.4.2 SW Write Calibration

The following steps should be executed:

NOTE

It is recommended to perform the write calibration using the HW method. The SW method is provided for debug purposes only.

1. Make sure that the initial value that is configured in the write delay line absolute offset of each byte (i.e. MPWRDLCTL[WR_DL_ABS_OFFSETn]) will place the write DQS somewhere inside the write DQ window
2. Configure the pre-defined value, which reflects the value that will be written and compared through the write calibration, to MPPDCMPR1[PDV1, PDV2]
3. Force the delay line to measure itself and to issue the requested write delay by configuring MPMUR0[FRC_MSR] = 1
4. Wait 16 DDR cycles till the write delay-line is updated with the absolute delay value for all bytes
5. Issue write command to any legal DDR address of the external DDR device
6. Issue read command, to the address written previously, from the external DDR device
7. Compare the read data byte to the associated byte in the pre-defined value for all bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial write DQS isn't inside the write DQ window. If the comparison passes then advance to next step.
8. MMDC resets the rd fifo (to the inverted pre-defined value) and it's pointers by setting MPDGCTRL[RST_RD_FIFO] = 1
9. Decrement the write delay line absolute offset of each byte by 1 (i.e. MPWRDLCTL[WR_DL_ABS_OFFSETn])
10. Force the delay line to measure itself and to issue the requested write delay by configuring MPMUR0[FRC_MSR] = 1
11. Issue write command to any legal DDR address of the external DDR device
12. Issue read command, to the address written previously, from the external DDR device
13. Compare the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the low write boundary of the associated byte of each byte at MPWRDLHWST0/1[HW_WR_DL_LOWn] . If the comparison passes then repeat steps 8-12. If all data comparisons fail then advance to the next step.
14. Start seeking the upper boundary and set the write delay line absolute offset of each byte to the initial value + 1
15. Force the delay line to measure itself and to issue the requested write delay by configuring MPMUR0[FRC_MSR] = 1
16. Reset the rd fifo (to the inverted pre-defined value) and it's pointers by setting MPDGCTRL[RST_RD_FIFO] = 1
17. Issue write command to any legal DDR address of the external DDR device
18. Issue read command, to the address written previously, from the external DDR device
19. Compare the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed

to store the upper write boundary of the associated byte of each byte at MPWRDLHWST0/1[HW_WR_DL_UPn]. If the comparison passes then increment the write delay line absolute offset of each byte by 1.

20. Force the delay line to measure itself and to issue the requested write delay by configuring MPMUR0[FRC_MSR] = 1
21. If all read data comparisons fail then advance to the next step else repeat steps 16-20.
22. .After finding the window boundary (lower and upper) of each write data byte then calculate the average between lower and upper boundaries and store the associated average at MPWRDLCTL[WR_DL_ABS_OFFSETn]
23. Force the delay line to measure itself and to issue the requested write delay by configuring MPMUR0[FRC_MSR] = 1

22.10.5 Write fine tuning

Write fine tuning is an additional circuit that provides the ability to fine tune the timing of each of the DQ/DM bits (relative to DQS) by up to 100 ps. To use the write fine tuning, select the number of delay units in each DQ/DM I/O (maximum 3 delay units of around 30-35 ps each). The delay can be configured independently for each DQ/DM. This configuration is controlled by the IOMUXC.

22.10.6 Read fine tuning

Read fine tuning is an additional circuit that provides the ability to fine tune the timing of each coming dq bits (relative to coming dqs) by up to +/-100 ps.

This is done by reducing the delay between the incoming rd_dqs by 100 ps and adding a configurable delay of up to 200 ps (6 delay units of around 30-35 ps each) for each DQ input. The delay can be configured independently for each DQ. The calibration of this mechanism can be done only by writing and reading data from the memory. Controlled by register MPRDDQBY#DL.

22.10.7 ZQ Fine Tuning

An offset can be added to the PU /PD values determined by the ZQ calibration process. The offset range is programmable from -7 to +7, controlled by the MMDC_MPPDCMPR2[ZQ_PU_OFFSET] and MMDC_MPPDCMPR2[ZQ_PD_OFFSET] fields. The offset is enabled/disabled by MMDC_MPPDCMPR2[ZQ_OFFSET_EN]. See MMDC_MPPDCMPR2 register for more information.

22.10.8 Duty cycle adjustment

Duty cycle adjustments can be made to the SDCLKx and SDQSx signals, see the MMDCx_MPDCCR register for more information.

22.11 MMDC Memory Map/Register Definition

The Memory Map is shown below.

NOTE

The terms *clocks* and *cycles* are used interchangeably and refer to the clock period of the main ddr clock , commonly referred to as the DDR frequency.

MMDC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AB_0000	MMDC Core Control Register (MMDC0_MDCTL)	32	R/W	0311_0000h	22.11.1/ 620
40AB_0004	MMDC Core Power Down Control Register (MMDC0_MDPDC)	32	R/W	0003_0012h	22.11.2/ 622
40AB_000C	MMDC Core Timing Configuration Register 0 (MMDC0_MDCFG0)	32	R/W	3236_22D3h	22.11.3/ 624
40AB_0010	MMDC Core Timing Configuration Register 1 (MMDC0_MDCFG1)	32	R/W	B6B1_8A23h	22.11.4/ 626
40AB_0014	MMDC Core Timing Configuration Register 2 (MMDC0_MDCFG2)	32	R/W	00C7_0092h	22.11.5/ 628
40AB_0018	MMDC Core Miscellaneous Register (MMDC0_MDMISC)	32	R/W	0000_1600h	22.11.6/ 630
40AB_001C	MMDC Core Special Command Register (MMDC0_MDSCR)	32	R/W	0000_0000h	22.11.7/ 633
40AB_0020	MMDC Core Refresh Control Register (MMDC0_MDREF)	32	R/W	0000_C000h	22.11.8/ 636
40AB_002C	MMDC Core Read/Write Command Delay Register (MMDC0_MDRWD)	32	R/W	0F9F_26D2h	22.11.9/ 638
40AB_0030	MMDC Core Out of Reset Delays Register (MMDC0_MDOR)	32	R/W	009F_0E0Eh	22.11.10/ 640
40AB_0034	MMDC Core MRR Data Register (MMDC0_MDMRR)	32	R	0000_0000h	22.11.11/ 641

Table continues on the next page...

MMDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AB_0038	MMDC Core Timing Configuration Register 3 (MMDC0_MDCFG3LP)	32	R/W	0000_0000h	22.11.12/ 642
40AB_003C	MMDC Core MR4 Derating Register (MMDC0_MDMR4)	32	R/W	0000_0000h	22.11.13/ 644
40AB_0040	MMDC Core Address Space Partition Register (MMDC0_MDASP)	32	R/W	0000_003Fh	22.11.14/ 646
40AB_0400	MMDC Core AXI Reordering Control Register (MMDC0_MAARCR)	32	R/W	5142_01F0h	22.11.15/ 647
40AB_0404	MMDC Core Power Saving Control and Status Register (MMDC0_MAPSR)	32	R/W	0000_1007h	22.11.16/ 649
40AB_0408	MMDC Core Exclusive ID Monitor Register0 (MMDC0_MAEXIDR0)	32	R/W	0020_0000h	22.11.17/ 651
40AB_040C	MMDC Core Exclusive ID Monitor Register1 (MMDC0_MAEXIDR1)	32	R/W	0060_0040h	22.11.18/ 652
40AB_0410	MMDC Core Debug and Profiling Control Register 0 (MMDC0_MADPCR0)	32	R/W	0000_0000h	22.11.19/ 653
40AB_0414	MMDC Core Debug and Profiling Control Register 1 (MMDC0_MADPCR1)	32	R/W	0000_0000h	22.11.20/ 654
40AB_0418	MMDC Core Debug and Profiling Status Register 0 (MMDC0_MADPSR0)	32	R	0000_0000h	22.11.21/ 655
40AB_041C	MMDC Core Debug and Profiling Status Register 1 (MMDC0_MADPSR1)	32	R	0000_0000h	22.11.22/ 655
40AB_0420	MMDC Core Debug and Profiling Status Register 2 (MMDC0_MADPSR2)	32	R	0000_0000h	22.11.23/ 656
40AB_0424	MMDC Core Debug and Profiling Status Register 3 (MMDC0_MADPSR3)	32	R	0000_0000h	22.11.24/ 656
40AB_0428	MMDC Core Debug and Profiling Status Register 4 (MMDC0_MADPSR4)	32	R	0000_0000h	22.11.25/ 657
40AB_042C	MMDC Core Debug and Profiling Status Register 5 (MMDC0_MADPSR5)	32	R	0000_0000h	22.11.26/ 657
40AB_0430	MMDC Core Step By Step Address Register (MMDC0_MASBS0)	32	R	0000_0000h	22.11.27/ 658
40AB_0434	MMDC Core Step By Step Address Attributes Register (MMDC0_MASBS1)	32	R	0000_0000h	22.11.28/ 658
40AB_0440	MMDC Core General Purpose Register (MMDC0_MAGENP)	32	R/W	0000_0000h	22.11.29/ 659
40AB_0800	MMDC PHY ZQ HW control register (MMDC0_MPZQHWCTRL)	32	R/W	A138_0000h	22.11.30/ 660
40AB_0804	MMDC PHY ZQ SW control register (MMDC0_MPZQSWCTRL)	32	R/W	0000_0000h	22.11.31/ 663
40AB_081C	MMDC PHY Read DQ Byte0 Delay Register (MMDC0_MPRDDQBY0DL)	32	R/W	0000_0000h	22.11.32/ 665
40AB_0820	MMDC PHY Read DQ Byte1 Delay Register (MMDC0_MPRDDQBY1DL)	32	R/W	0000_0000h	22.11.33/ 668

Table continues on the next page...

MMDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AB_0824	MMDC PHY Read DQ Byte2 Delay Register (MMDC0_MPRDDQBY2DL)	32	R/W	0000_0000h	22.11.34/ 671
40AB_0828	MMDC PHY Read DQ Byte3 Delay Register (MMDC0_MPRDDQBY3DL)	32	R/W	0000_0000h	22.11.35/ 673
40AB_083C	MMDC PHY Read DQS Gating Control Register 0 (MMDC0_MPDGCTRL0)	32	R/W	0000_0000h	22.11.36/ 676
40AB_0848	MMDC PHY Read delay-lines Configuration Register (MMDC0_MPRDDLCTL)	32	R/W	4040_4040h	22.11.37/ 677
40AB_084C	MMDC PHY Read delay-lines Status Register (MMDC0_MPRDDLST)	32	R	0000_0000h	22.11.38/ 678
40AB_0850	MMDC PHY Write delay-lines Configuration Register (MMDC0_MPWRDLCTL)	32	R/W	4040_4040h	22.11.39/ 679
40AB_0854	MMDC PHY Write delay-lines Status Register (MMDC0_MPWRDLST)	32	R	0000_0000h	22.11.40/ 681
40AB_085C	MMDC ZQ LPDDR2 HW Control Register (MMDC0_MPZQLP2CTL)	32	R/W	1B5F_0109h	22.11.41/ 682
40AB_0860	MMDC PHY Read Delay HW Calibration Control Register (MMDC0_MPRDDLHWCTL)	32	R/W	0000_0000h	22.11.42/ 684
40AB_0864	MMDC PHY Write Delay HW Calibration Control Register (MMDC0_MPWRDLHWCTL)	32	R/W	0000_0000h	22.11.43/ 686
40AB_0868	MMDC PHY Read Delay HW Calibration Status Register 0 (MMDC0_MPRDDLHWST0)	32	R	0000_0000h	22.11.44/ 687
40AB_086C	MMDC PHY Read Delay HW Calibration Status Register 1 (MMDC0_MPRDDLHWST1)	32	R	0000_0000h	22.11.45/ 688
40AB_0870	MMDC PHY Write Delay HW Calibration Status Register 0 (MMDC0_MPWRDLHWST0)	32	R	0000_0000h	22.11.46/ 689
40AB_0874	MMDC PHY Write Delay HW Calibration Status Register 1 (MMDC0_MPWRDLHWST1)	32	R	0000_0000h	22.11.47/ 690
40AB_088C	MMDC PHY Pre-defined Compare Register 1 (MMDC0_MPPDCMPR1)	32	R/W	0000_0000h	22.11.48/ 691
40AB_0890	MMDC PHY Pre-defined Compare and CA delay-line Configuration Register (MMDC0_MPPDCMPR2)	32	R/W	0040_0000h	22.11.49/ 692
40AB_0894	MMDC PHY SW Dummy Access Register (MMDC0_MPSWDAR0)	32	R/W	0000_0000h	22.11.50/ 695
40AB_0898	MMDC PHY SW Dummy Read Data Register 0 (MMDC0_MPSWDRDR0)	32	R	FFFF_FFFFh	22.11.51/ 696
40AB_089C	MMDC PHY SW Dummy Read Data Register 1 (MMDC0_MPSWDRDR1)	32	R	FFFF_FFFFh	22.11.52/ 697
40AB_08A0	MMDC PHY SW Dummy Read Data Register 2 (MMDC0_MPSWDRDR2)	32	R	FFFF_FFFFh	22.11.53/ 697
40AB_08A4	MMDC PHY SW Dummy Read Data Register 3 (MMDC0_MPSWDRDR3)	32	R	FFFF_FFFFh	22.11.54/ 697
40AB_08A8	MMDC PHY SW Dummy Read Data Register 4 (MMDC0_MPSWDRDR4)	32	R	FFFF_FFFFh	22.11.55/ 698

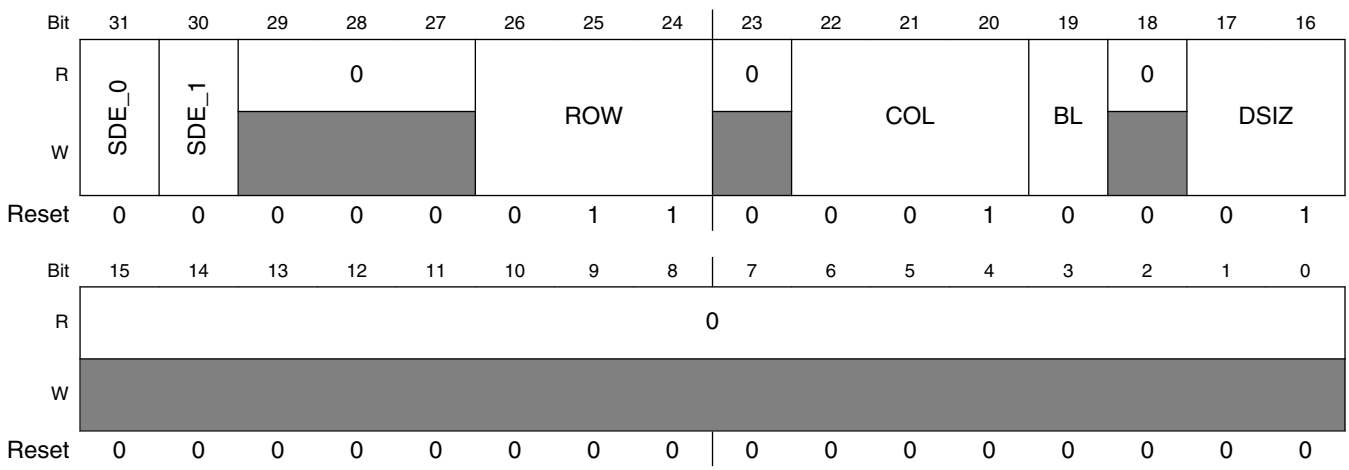
Table continues on the next page...

MMDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AB_08AC	MMDC PHY SW Dummy Read Data Register 5 (MMDC0_MPSWDRDR5)	32	R	FFFF_FFFFh	22.11.56/ 698
40AB_08B0	MMDC PHY SW Dummy Read Data Register 6 (MMDC0_MPSWDRDR6)	32	R	FFFF_FFFFh	22.11.57/ 699
40AB_08B4	MMDC PHY SW Dummy Read Data Register 7 (MMDC0_MPSWDRDR7)	32	R	FFFF_FFFFh	22.11.58/ 699
40AB_08B8	MMDC PHY Measure Unit Register (MMDC0_MPMUR0)	32	R/W	0000_0000h	22.11.59/ 700
40AB_08C0	MMDC Duty Cycle Control Register (MMDC0_MPDCCR)	32	R/W	2492_2492h	22.11.60/ 701

22.11.1 MMDC Core Control Register (MMDCx_MDCTL)

Address: 40AB_0000h base + 0h offset = 40AB_0000h



MMDCx_MDCTL field descriptions

Field	Description
31 SDE_0	MMDC Enable CS0. This bit enables/disables accesses from the MMDC toward Chip Select 0. The reset value of this bit is "0" (i.e. No clocks and clock enable will be driven to the memory). At the enabling point the MMDC will perform an initialization process (including a delay on RESET and/or CKE) for both chip selects. The initialization length depends on the configured memory type. 0 Disabled 1 Enabled
30 SDE_1	MMDC Enable CS1. This bit enables/disables accesses from the MMDC toward Chip Select 1. The reset value of this bit is "0" (i.e. No clocks and clock enable will be driven to the memory). At the enabling point the MMDC will perform an initialization process (including a delay on RESET and/or CKE) for both chip selects. The initialization length depends on the configured memory type.

Table continues on the next page...

MMDCx_MDCTL field descriptions (continued)

Field	Description
	0 Disabled 1 Enabled
29–27 Reserved	This read-only field is reserved and always has the value 0.
26–24 ROW	Row Address Width. This field specifies the number of row addresses used by the memory array. It will affect the way an incoming address will be decoded. Settings 110-111 are reserved 000 11 bits Row 001 12 bits Row 010 13 bits Row 011 14 bits Row 100 15 bits Row 101 16 bits Row
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 COL	Column Address Width. This field specifies the number of column addresses used by the memory array. It will determine how an incoming address will be decoded. 0x0 9 bits column 0x1 10 bits column 0x2 11 bits column 0x3 8 bits column 0x4 12 bits column 0x5-0xF Reserved
19 BL	Burst Length. This field determines the burst length of the DDR device. In LPDDR2 mode the MMDC supports burst length 4. In LPDDR3 mode the MMDC supports burst length 8. 0 Burst Length 4 is used 1 Burst Length 8 is used
18 Reserved	This read-only field is reserved and always has the value 0.
17–16 DSIZ	DDR data bus size. This field determines the size of the data bus of the DDR memory 0 16-bit data bus 1 32-bit data bus — — — 2-3 Reserved
Reserved	This read-only field is reserved and always has the value 0.

22.11.2 MMDC Core Power Down Control Register (MMDCx_MDPDC)

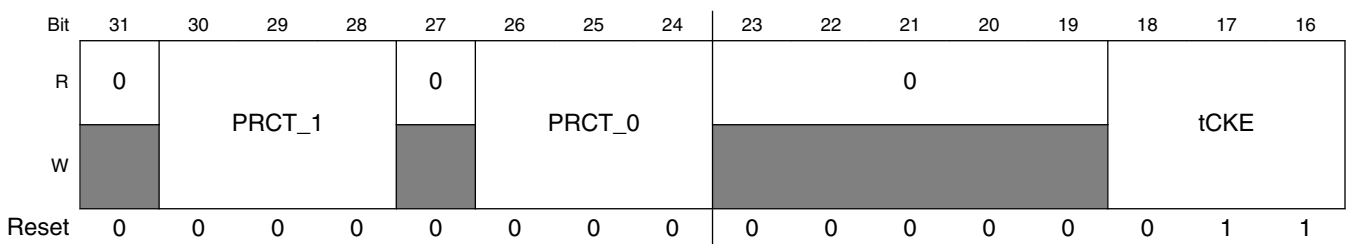
Table 22-10. PRCT field encoding

PRCT[2:0]	Precharge Timer
000	Disabled (Bit field reset value)
001	2 clocks
010	4 clocks
011	8 clocks
100	16 clocks
101	32 clocks
110	64 clocks
111	128 clocks

Table 22-11. PWDT field encoding

PWDT[3:0]	Power Down Time-out
0000	Disabled (bit field reset value)
0001	16 cycles
0010	32 cycles
0011	64 cycles
0100	128 cycles
0101	256 cycles
0110	512 cycles
0111	1024 cycles
1000	2048 cycles
1001	4096 cycles
1010	8196 cycles
1011	16384 cycles
1100	32768 cycles
1101-1111	Reserved

Address: 40AB_0000h base + 4h offset = 40AB_0004h



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PWDT_1				PWDT_0				SLOW_PD	BOTH_CS_PD	tCKSRX			tCKSRE		
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

MMDCx_MDPDC field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 PRCT_1	Precharge Timer - Chip Select 1. This field determines the amount of idle cycle for which chip select 1 will be automatically precharged. The amount of cycles are determined according to the PRCT Field Encoding table above.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 PRCT_0	Precharge Timer - Chip Select 0. This field determines the amount of idle cycle for which chip select 0 will be automatically precharged. The amount of cycles are determined according to the table below.
23–19 Reserved	This read-only field is reserved and always has the value 0.
18–16 tCKE	CKE minimum pulse width. This field determines the minimum pulse width of CKE. 0x0 1 cycle 0x1 2 cycles 0x6 7 cycles 0x7 8 cycles
15–12 PWDT_1	Power Down Timer - Chip Select 1. This field determines the amount of idle cycle for which chip select 1 will be automatically get into precharge/active power down. The amount of cycles are determined according to the PWDT Field Encoding table above.
11–8 PWDT_0	Power Down Timer - Chip Select 0. This field determines the amount of idle cycle for which chip select 0 will be automatically get into precharge/active power down. The amount of cycles are determined according to the PWDT Field Encoding table above.
7 SLOW_PD	Slow/fast power down. LPDDR2 mode: This field is not relevant and should remain in its default reset value. LPDDR3 mode: This field is not relevant and should remain in its default reset value. **NOTE:** Memory should be configured the same. 0 Fast mode. 1 Slow mode.
6 BOTH_CS_PD	Parallel power down entry to both chip selects. When power down timer is used for both chip-selects (i.e. PWDT_0 and PWDT1 don't equal "0") , then if this bit is enabled, the MMDC will enter power down only if the amount of idle cycles of both chip selects was obtained.

Table continues on the next page...

MMDCx_MDPDC field descriptions (continued)

Field	Description
	0 Each chip select can enter power down independently according to its configuration. 1 Chip selects can enter power down only if the amount of idle cycles of both chip selects was obtained.
5–3 tCKSRX	Valid clock cycles before self-refresh exit. This field determines the amount of clock cycles before self-refresh exit. 0x0 0 cycle 0x1 1 cycles 0x6 6 cycles 0x7 7 cycles
tCKSRE	Valid clock cycles after self-refresh entry. This field determines the amount of clock cycles after self-refresh entry. 0x0 0 cycle 0x1 1 cycles 0x6 6 cycles 0x7 7 cycles

22.11.3 MMDC Core Timing Configuration Register 0 (MMDCx_MDCFG0)

Address: 40AB_0000h base + Ch offset = 40AB_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	tRFC								tXS								tXP			tXPDLL				tFAW							tCL			
Reset	0	0	1	1	0	0	1	0	0	0	1	1	0	1	1	0	0	0	1	0	0	0	1	0	1	1	0	1	0	0	1	1

MMDCx_MDCFG0 field descriptions

Field	Description
31–24 tRFC	Refresh command to Active or Refresh command time. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xFE 255 clocks 0xFF 256 clocks
23–16 tXS	Exit self refresh to non READ command. In LPDDR2 it is called tXSR, self-refresh exit to next valid command delay. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 - 0x15 reserved 0x16 23 clocks

Table continues on the next page...

MMDCx_MDCFG0 field descriptions (continued)

Field	Description
	0x17 24 clocks 0xFE 255 clocks 0xFF 256 clocks
15–13 tXP	Exit power down with DLL-on to any valid command. Exit power down with DLL-frozen to commands not requiring a locked DLL In LPDDR2/LPDDR3 mode this field is referred to Exit power-down to next valid command delay. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 cycle 0x1 2 cycles 0x6 7 cycles 0x7 8 cycles
12–9 tXPDLL	Exit precharge power down with DLL frozen to commands requiring DLL. LPDDR2 mode: This field is not relevant and should remain in its default reset value. LPDDR3 mode: This field is not relevant and should remain in its default reset value. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF 16 clocks
8–4 tFAW	Four Active Window (all banks). See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x1E 31 clocks 0x1F 32 clocks
tCL	CAS Read Latency. In LPDDR2 mode this field is referred to RL. In LPDDR3 mode this field is referred to RL. **NOTE:** In LPDDR2/LPDDR3 mode only the RL/WL pairs are allowed as specified in MR2 register. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 3 cycles 0x1 4 cycles 0x2 5 cycles 0x3 6 cycles 0x4 7 cycles

Table continues on the next page...

MMDCx_MDCFG0 field descriptions (continued)

Field	Description
	0x5 8 cycles 0x6 9 cycles 0x7 10 cycles 0x8 11 cycles 0x9 - 0xF Reserved

22.11.4 MMDC Core Timing Configuration Register 1 (MMDCx_MDCFG1)

Address: 40AB_0000h base + 10h offset = 40AB_0010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	tRCD			tRP			tRC					tRAS				
Reset	1	0	1	1	0	1	1	0	1	0	1	1	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	tRPA	0			tWR				tMRD			0		tCWL		
Reset	1	0	0	0	1	0	1	0	0	0	1	0	0	0	1	1

MMDCx_MDCFG1 field descriptions

Field	Description
31–29 tRCD	Active command to internal read or write delay time (same bank). In LPDDR2/LPDDR3 mode, this parameter should be configured at tRCD_LP. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x3 4 clocks 0x4 5 clocks 0x5 6 clocks 0x6 7 clocks 0x7 8 clocks
28–26 tRP	Precharge command period (same bank). In LPDDR2 mode this parameter should be configured at tRPpb_LP. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x3 4 clocks

Table continues on the next page...

MMDCx_MDCFG1 field descriptions (continued)

Field	Description
	0x4 5 clocks 0x5 6 clocks 0x6 7 clocks 0x7 8 clocks
25–21 tRC	Active to Active or Refresh command period (same bank). In LPDDR2/LPDDR3 mode, this parameter should be configured at tRC_LP. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x1E 31 clocks 0x1F 32 clocks
20–16 tRAS	Active to Precharge command period (same bank). See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x1E 31 clocks 0x1F Reserved
15 tRPA	Precharge-all command period. In LPDDR2/LPDDR3 mode, this parameter should be configured at tRPab_LP. See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. **NOTE:** A Precharge All command must be issued prior to the MRW command to ensure robust DDR initialization. This command is required to be issued to both chip selects if two chip selects are utilized in the system. 0 Will be equal to: tRP. 1 Will be equal to: tRP+1.
14–12 Reserved	This read-only field is reserved and always has the value 0.
11–9 tWR	WRITE recovery time (same bank). See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1cycle 0x1 2cycles 0x2 3cycles 0x3 4cycles 0x4 5cycles 0x5 6cycles

Table continues on the next page...

MMDCx_MDCFG1 field descriptions (continued)

Field	Description
	0x6 7cycles 0x7 8 cycles
8–5 tMRD	Mode Register Set command cycle (all banks). In LPDDR2/LPDDR3 mode this field should be set to max(tMRR,tMRW). See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF 16 clocks
4–3 Reserved	This read-only field is reserved and always has the value 0.
tCWL	CAS Write Latency. In LPDDR2/LPDDR3 mode this field is referred to WL. 0x0 1 cycles (LPDDR2/LPDDR3) 0x1 2 cycles (LPDDR2/LPDDR3) 0x2 3 cycles (LPDDR2/LPDDR3) 0x3 4 cycles (LPDDR2/LPDDR3) 0x4 5 cycles (LPDDR2/LPDDR3) 0x5 6 cycles (LPDDR2/LPDDR3) 0x6 7 cycles (LPDDR2/LPDDR3) 0x7 Reserved

22.11.5 MMDC Core Timing Configuration Register 2 (MMDCx_MDCFG2)

Address: 40AB_0000h base + 14h offset = 40AB_0014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							tDLLK									0							tRTP			tWTR			tRRD		
W																																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0

MMDCx_MDCFG2 field descriptions

Field	Description
31–25 Reserved	This read-only field is reserved and always has the value 0.
24–16 tDLLK	DLL locking time. LPDDR2 mode: This field is not relevant and should remain in its default reset value. LPDDR3 mode: This field is not relevant and should remain in its default reset value.

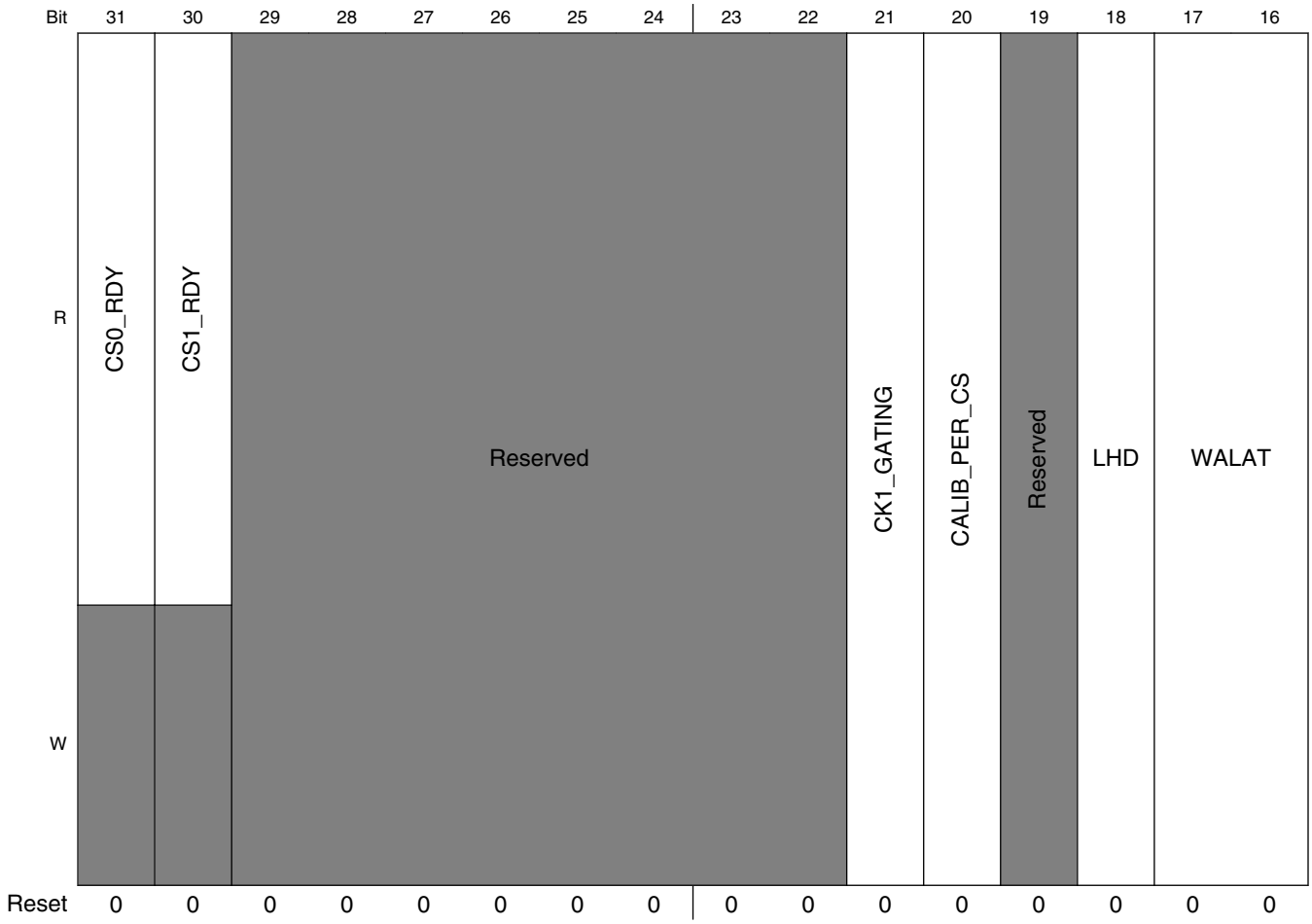
Table continues on the next page...

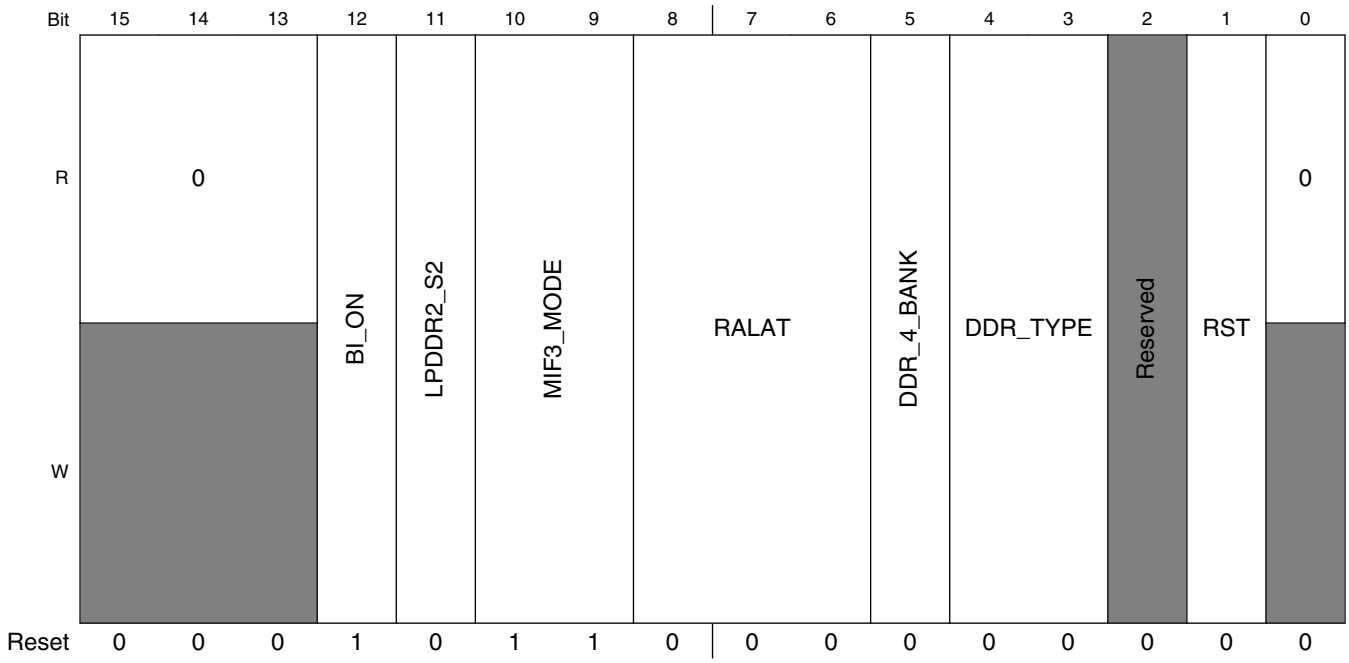
MMDCx_MDCFG2 field descriptions (continued)

Field	Description
	See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1 cycle. 0x1 2 cycles. 0x2 3 cycles. 0xC7 200 cycles 0x1FE 511 cycles. 0x1FF 512 cycles
15–9 Reserved	This read-only field is reserved and always has the value 0.
8–6 tRTP	Internal READ command to Precharge command delay (same bank). See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1cycle 0x1 2cycles 0x2 3cycles 0x3 4cycles 0x4 5cycles 0x5 6cycles 0x6 7cycles 0x7 8 cycles
5–3 tWTR	Internal WRITE to READ command delay (same bank). See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1cycle 0x1 2cycles 0x2 3cycles 0x3 4cycles 0x4 5cycles 0x5 6cycles 0x6 7cycles 0x7 8 cycles
tRRD	Active to Active command period (all banks). See LPDDR2 SDRAM Specification JESD209-2B (February 2010) , LPDDR3 SDRAM Specification JESD209-3C (August 2015) for a detailed description of this parameter. 0x0 1cycle 0x1 2cycles 0x2 3cycles 0x3 4cycles 0x4 5cycles 0x5 6cycles 0x6 7cycles 0x7 Reserved

22.11.6 MMDC Core Miscellaneous Register (MMDCx_MDMISC)

Address: 40AB_0000h base + 18h offset = 40AB_0018h





MMDCx_MDMISC field descriptions

Field	Description
31 CS0_RDY	External status device on CS0. This is a read-only status bit, that indicates whether the external memory is in wake-up period. 0 Device in wake-up period. 1 Device is ready for initialization.
30 CS1_RDY	External status device on CS1. This is a read-only status bit, that indicates whether the external memory is in wake-up period. 0 Device in wake-up period. 1 Device is ready for initialization.
29–22 -	This field is reserved.
21 CK1_GATING	Gating the secondary DDR clock. When this bit is asserted then the MMDC will disable the secondary DDR clock 0 MMDC drives two clocks toward the DDR memory 1 MMDC drives only one clock toward the DDR memory (CK0)
20 CALIB_PER_CS	Number of chip-select for calibration process. This bit determines the chip-select index that the associated calibration is targeted to. Relevant for read, write, write leveling and read DQS gating calibrations 0 Calibration is targeted to CS0 1 Calibration is targeted to CS1
19 -	This field is reserved. Reserved
18 LHD	Latency hiding disable. This is a debug feature. When set to "1" the MMDC will handle one read/write access at a time. Meaning that the MMDC pipe-line will be limited to 1 open access (next AXI address phase will be acknowledged if the current AXI data phase had finished)

Table continues on the next page...

MMDCx_MDMISC field descriptions (continued)

Field	Description
	0 Latency hiding on. 1 Latency hiding disable.
17–16 WALAT	Write Additional latency. **NOTE:** The purpose of WALAT is to add time delay at the end of a burst write operation to ensure that the JEDEC time specification for Write Post Amble Delay (tWPST) is met (DQS strobe is held low at the end of a write burst for > 30% a clock cycle before it is released). If the value of any of the WL_DL_ABS_OFFSETn register fields are greater than '1F', WALAT should be set to '1' (cycle additional delay). WALAT should be further increased for any full cycle delays added by the WL_CYC_DELn register fields. 0x0 No additional latency required. 0x1 1 cycle additional delay 0x2 2 cycles additional delay 0x3 3 cycles additional delay
15–13 Reserved	This read-only field is reserved and always has the value 0.
12 BI_ON	Bank Interleaving On. This bit controls the organization of the bank, row and column address bits. For further information see Address decoding . 0 Banks are not interleaved, and address will be decoded as bank-row-column 1 Banks are interleaved, and address will be decoded as row-bank-column
11 LPDDR2_S2	LPDDR2 S2 device type indication. In case LPDDR2 device is used (DDR_TYPE = 0x1), this bit will indicate whether S2 or S4 device is used. 0x0 LPDDR2-S4 device is used. 0x1 LPDDR2-S2 device is used.
10–9 MIF3_MODE	Command prediction working mode. This field determines the level of command prediction that will be used by the MMDC 00 Disable prediction. 01 Enable prediction based on : Valid access on first pipe line stage. 10 Enable prediction based on: Valid access on first pipe line stage, Valid access on axi bus. 11 Enable prediction based on: Valid access on first pipe line stage, Valid access on axi bus, Next miss access from access queue.
8–6 RALAT	Read Additional Latency. This field determines the additional read latency which is added to CAS latency and internal delays for which the MMDC will retrieve the read data from the internal FIFO. This field is used to compensate on board/chip delays. **NOTE:** In LPDDR2 mode 2 extra cycles will be added internally in order to compensate tDQSCK delay. 0x0 no additional latency. 0x1 1 cycle additional latency. 0x2 2 cycles additional latency. 0x3 3 cycles additional latency. 0x4 4 cycles additional latency. 0x5 5 cycles additional latency. 0x6 6 cycles additional latency. 0x7 7 cycles additional latency.

Table continues on the next page...

MMDCx_MDMISC field descriptions (continued)

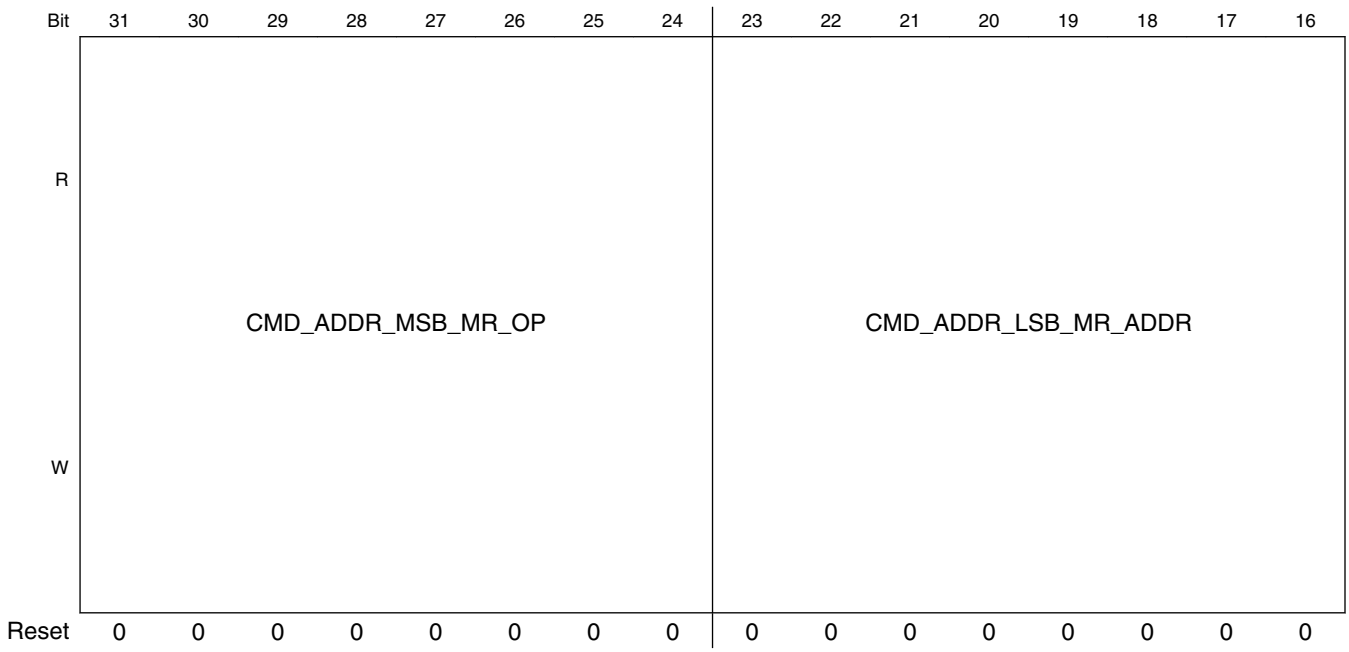
Field	Description
5 DDR_4_BANK	Number of banks per DDR device. When this bit is set to "1" then the MMDC will work with DDR device of 4 banks. 0 8 banks device is being used. (Default) 1 4 banks device is being used
4–3 DDR_TYPE	DDR TYPE. This field determines the type of the external DDR device. 0x0 Reserved 0x1 LPDDR2 device is used. 0x2 Reserved 0x3 LPDDR3 device is used.
2 -	This field is reserved.
1 RST	Software Reset. When this bit is asserted then the internal FSMs and registers of the MMDC will be initialized. **NOTE:** This bit once asserted gets deasserted automatically. 0 Do nothing. 1 Assert reset to the MMDC.
0 Reserved	This read-only field is reserved and always has the value 0.

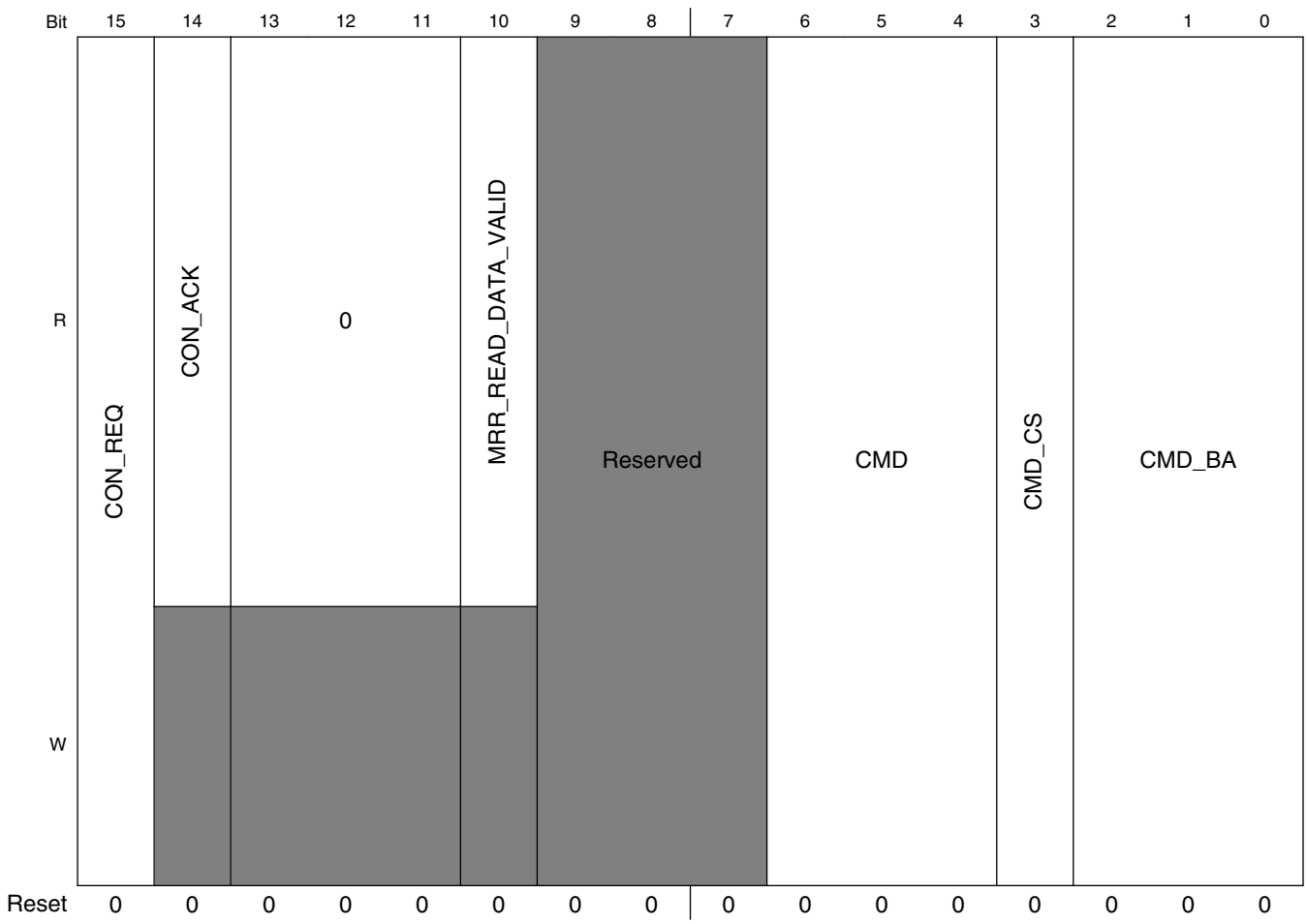
22.11.7 MMDC Core Special Command Register (MMDCx_MDSCR)

This register is used to issue special commands manually toward the external DDR device (such as load mode register, manual self refresh, manual precharge and so on). Every write to this register will be interpreted as a command, and a read from this register will show the last command that was executed.

Every write to this register will result in one special command, and the IP bus will assert ips_xfr_wait as long as the special command is being carried out.

Address: 40AB_0000h base + 1Ch offset = 40AB_001Ch





MMDCx_MDSCR field descriptions

Field	Description
31–24 CMD_ADDR_ MSB_MR_OP	Command/Address MSB. This field indicates the MSB of the command/Address. In LPDDR2/LPDDR3 this field indicates the MRW operand
23–16 CMD_ADDR_ LSB_MR_ADDR	Command/Address LSB. This field indicates the LSB of the command/Address In LPDDR2/LPDDR3 this field indicates the MRR/MRW address
15 CON_REQ	Configuration request. When this bit is set then the MMDC will clean the pending AXI accesses and will prevent from further AXI accesses to be acknowledged. This field guarantee safe configuration (or change configuration) of the MMDC while no access is in process and prevents an unexpected behaviour. After setting this bit, it is needed to poll on CON_ACK until it is set to "1". When CON_ACK is asserted then configuration is permitted. After configuration is completed then this bit must be deasserted in order to process further AXI accesses. **NOTE:** This bit is asserted at the end of the reset sequence, meaning that the MMDC is waiting to configure and initialize the external memory before accepting any AXI accesses. Configuration request/acknowledge mechanism should be used for the following procedures: changing of timing parameters , during calibration process or driving commands via MDSCR[CMD] 0 No request to configure MMDC. 1 A request to configure MMDC is valid
14 CON_ACK	Configuration acknowledge. Whenever this bit is set, it is permitted to configure MMDC IP registers. 0 Configuration of MMDC registers is forbidden. 1 Configuration of MMDC registers is permitted.
13–11 Reserved	This read-only field is reserved and always has the value 0.
10 MRR_READ_ DATA_VALID	MRR read data valid. This field indicates that read data is valid at MDMRR register **NOTE:** This field is relevant only for LPDDR2/LPDDR3 mode 0 Cleared upon the assertion of MRR command 1 Set after MRR data is valid and stored at MDMRR register.
9–7 -	This field is reserved.
6–4 CMD	Command. This field contains the command to be executed. This field will be automatically cleared after the command will be send to the DDR memory. **NOTE:** Only the 0x5 Precharge All command should be used, memory operation is unpredictable when using the 0x1 command **NOTE:** A Precharge All command must be issued prior to the MRW command to ensure robust DDR initialization. This command is required to be issued to both chip selects if two chip selects are utilized in the system. 0x0 Normal operation 0x1 Precharge All, command is sent independently of bank status (set correct CMD_CS). Will be issued even if banks are closed. Primarily used for initialization sequence purposes. Not to be used during run-time operation. 0x2 Auto-Refresh Command (set correct CMD_CS).

Table continues on the next page...

MMDCx_MDSCR field descriptions (continued)

Field	Description
	0x3 Load Mode Register Command (LPDDR2/LPDDR3, set correct CMD_CS, MR_OP, MR_ADDR) 0x4 ZQ calibration 0x5 Precharge All, only if banks open (set correct CMD_CS). 0x6 MRR command (LPDDR2/LPDDR3, set correct CMD_CS, MR_ADDR) 0x7 Reserved
3 CMD_CS	Chip Select. This field determines which chip select the command is targeted to 0 to Chip-select 0 1 to Chip-select 1
CMD_BA	Bank Address. This field determines the address of the bank within the selected chip-select to where the command is targeted. 0x0 bank address 0 0x1 bank address 1 0x2 bank address 2 0x7 bank address 7

22.11.8 MMDC Core Refresh Control Register (MMDCx_MDREF)

This register determines the refresh scheme that will be executed toward the DDR device. It specifies how often a refresh cycle occurs and how many refresh commands will be executed every refresh cycle.

For further information see Refresh Scheme .

The following tables show examples of possible refresh schemes.

Table 22-12. Refresh rate example for REF_SEL = 0

REFR[2:0]	Number of refresh commands every 64KHz	Average periodic refresh rate (tREFI)	System Refresh period
0x0	1	15.6 µs	tRFC
0x1	2	7.8 µs	2*tRFC
0x3	4	3.9µs	4*tRFC
0x7	8	1.95 µs	8*tRFC

Table 22-13. Refresh rate example for REF_SEL = 1

REFR[2:0]	Number of refresh commands every 32KHz	Average periodic refresh rate (tREFI)	System Refresh period
0x1	2	15.6 µs	2*tRFC

Table continues on the next page...

Table 22-13. Refresh rate example for REF_SEL = 1 (continued)

REFR[2:0]	Number of refresh commands every 32KHz	Average periodic refresh rate (tREFI)	System Refresh period
0x3	4	7.8 µs	4*tRFC
0x7	8	3.9µs	8*tRFC

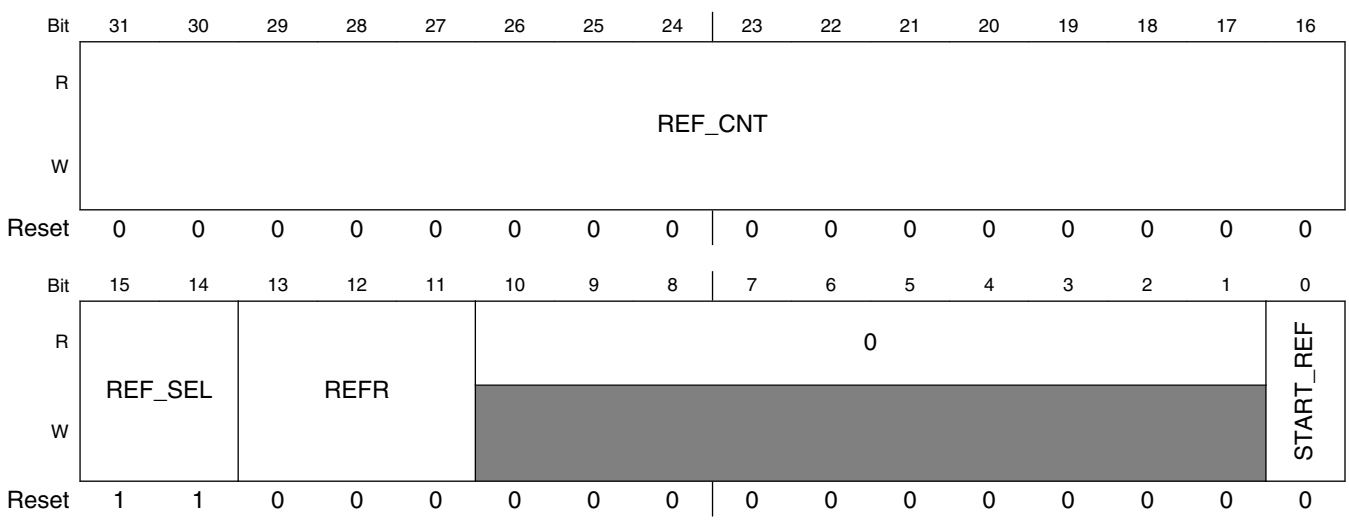
Table 22-14. Refresh rate example for REF_SEL = 2@ 400MHz

REFR[2:0]	Number of refresh commands every refresh cycle	REF_CNT	Average periodic refresh rate (tREFI)	System Refresh period
0x0	1	0x618	3.9 µs	tRFC
0x1	2	0xC30	3.9 µs	2*tRFC
0x2	3	0x1248	3.9µs	3*tRFC
0x3	4	0x1860	3.9 µs	4*tRFC

Other refresh configurations are also allowed; the configuration values in the tables above are only examples for obtaining the desired average periodic refresh rate.

If the required average periodic refresh rate (tREFI) is kept, all of the rows will be refreshed in every refresh window. Because the memory device issues additional refresh commands for every refresh it receives, the tREFI remains the same across the device, regardless of its number of rows. This is particularly relevant in the tRFC parameter, which becomes bigger as the density increases.

Address: 40AB_0000h base + 20h offset = 40AB_0020h



MMDCx_MDREF field descriptions

Field	Description
31–16 REF_CNT	Refresh Counter at DDR clock period If REF_SEL equals '2' a refresh cycle will begin every amount of DDR cycles configured in this field. 0x0 Reserved. 0x1 1 cycle. 0xFFFE 65534 cycles. 0xFFFF 65535 cycles.
15–14 REF_SEL	Refresh Selector. This bit selects the source of the clock that will trigger each refresh cycle: 0 Periodic refresh cycles will be triggered in frequency of 64KHz. 1 Periodic refresh cycles will be triggered in frequency of 32KHz. 2 Periodic refresh cycles will be triggered every amount of cycles that are configured in REF_CNT field. 3 No refresh cycles will be triggered.
13–11 REFR	Refresh Rate. This field determines how many refresh commands will be issued every refresh cycle. After every refresh command the MMDC won't drive any command to the DDR device until satisfying tRFC period 0x0 1 refresh 0x1 2 refreshes 0x2 3 refreshes 0x3 4 refreshes 0x4 5 refreshes 0x5 6 refreshes 0x6 7 refreshes 0x7 8 refreshes
10–1 Reserved	This read-only field is reserved and always has the value 0.
0 START_REF	Manual start of refresh cycle. When this field is set to '1' the MMDC will start a refresh cycle immediately according to number of refresh commands that are configured in 'REFR' field. This bit returns to zero automatically. 0 Do nothing. 1 Start a refresh cycle.

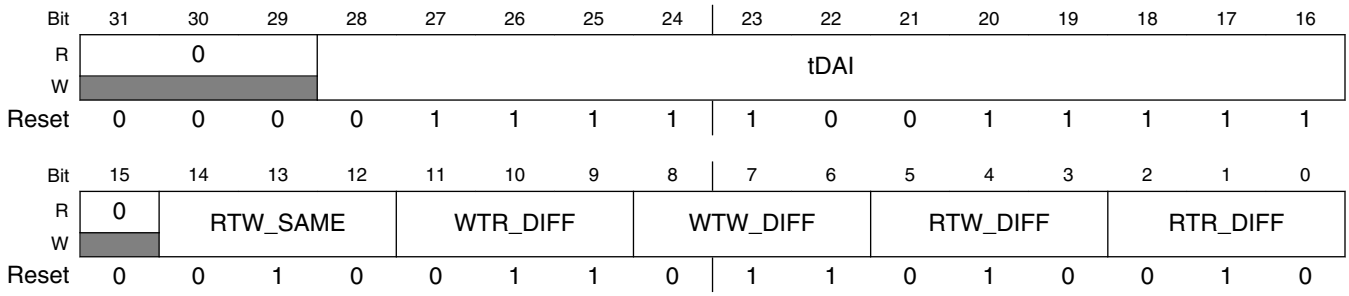
22.11.9 MMDC Core Read/Write Command Delay Register (MMDCx_MDRWD)

This register determines the delay between back to back read and write accesses. The register reset values are set to the minimum required value.

NOTE

As the default values are set to achieve optimal results, changing them is discouraged.

Address: 40AB_0000h base + 2Ch offset = 40AB_002Ch



MMDCx_MDRWD field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28–16 tDAI	Device auto initialization period.(maximum) **NOTE:** This field is relevant only to LPDDR2/3 mode 0x0 1 cycle 0xF9F 4000 cycles (Default, JEDEC value for LPDDR2, gives 10us at 400MHz clock). 0x1FFF 8192 cycles
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 RTW_SAME	Read to write delay for the same chip-select. This field controls the delay between read to write commands toward the same chip select. The total delay is calculated according to: BL/2 + RTW_SAME + (tCL-tCWL) + RALAT 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles (Default) 0x3 3 cycles 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
11–9 WTR_DIFF	Write to read delay for different chip-select. This field controls the delay between write to read commands toward different chip select. The total delay is calculated according to: BL/2 + WTR_DIFF + (tCL-tCWL) + RALAT 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles 0x3 3 cycles (Default) 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
8–6 WTW_DIFF	Write to write delay for different chip-select. This field controls the delay between write to write commands toward different chip select.

Table continues on the next page...

MMDCx_MDRWD field descriptions (continued)

Field	Description
	The total delay is calculated according to: BL/2 + WTW_DIFF 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles 0x3 3 cycles (Default) 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
5–3 RTW_DIFF	Read to write delay for different chip-select. This field controls the delay between read to write commands toward different chip select. The total delay is calculated according to: BL/2 + RTW_DIFF + (tCL - tCWL) + RALAT 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles (Default) 0x3 3 cycles 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
RTR_DIFF	Read to read delay for different chip-select. This field controls the delay between read to read commands toward different chip select. The total delay is calculated according to: BL/2 + RTR_DIFF 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles (Default) 0x3 3 cycles 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles

22.11.10 MMDC Core Out of Reset Delays Register (MMDCx_MDOR)

This register defines delays that must be kept when MMDC exits reset.

Address: 40AB_0000h base + 30h offset = 40AB_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								tXPR								0		SDE_to_RST						0		RST_to_CKE					
W																																
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1	0

MMDCx_MDOR field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value 0.
23–16 tXPR	LPDDR2/3: Not relevant to this mode and should remain in default reset value. 0x0 Reserved 0x1 2 cycles 0x2 3 cycles 0xFE 255 cycles 0xFF 256 cycles
15–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 SDE_to_RST	LPDDR2 mode: This field is not relevant and should remain in its default reset value. **NOTE:** Each cycle in this field is 15.258 us. 0x0 Reserved 0x1 Reserved 0x2 Reserved 0x3 1 cycles 0x4 2 cycles 0x10 14 cycles - total of 200 us 0x3E 60 cycles 0x3F 61 cycles
7–6 Reserved	This read-only field is reserved and always has the value 0.
RST_to_CKE	LPDDR2: Idle time after first CKE assertion (JEDEC value is 200 us). **NOTE:** Each cycle in this field is 15.258 us. 0x0 Reserved 0x1 Reserved 0x2 Reserved 0x3 1 cycles 0x10 14 cycles (JEDEC value for LPDDR2) - total of 200 us 0x23 33 cycles - total of 500 us 0x3E 60 cycles 0x3F 61 cycles

22.11.11 MMDC Core MRR Data Register (MMDCx_MDMRR)

This register contains data that was collected after issuing MRR command. The data in this register is valid only when MDSCR[MRR_READ_DATA_VALID] is set to "1".

This register is relevant only in LPDDR2/3 mode. For further information see LPDDR2/3 Refresh Rate Update and Timing Derating.

Address: 40AB_0000h base + 34h offset = 40AB_0034h

Bit	31–24	23–16	15–8	7–0
R	MRR_READ_DATA3	MRR_READ_DATA2	MRR_READ_DATA1	MRR_READ_DATA0
W				
Reset	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0

MMDCx_MDMRR field descriptions

Field	Description
31–24 MRR_READ_ DATA3	MRR DATA that arrived on DQ[31:24]
23–16 MRR_READ_ DATA2	MRR DATA that arrived on DQ[23:16]
15–8 MRR_READ_ DATA1	MRR DATA that arrived on DQ[15:8]
MRR_READ_ DATA0	MRR DATA that arrived on DQ[7:0]

22.11.12 MMDC Core Timing Configuration Register 3 (MMDCx_MDCFG3LP)

This register is relevant only for LPDDR2/3 mode.

Address: 40AB_0000h base + 38h offset = 40AB_0038h

Bit	31–22	21–16	15–12	11–8	7–4	3–0
R	0	RC_LP	0	tRCD_LP	tRPpb_LP	tRPab_LP
W						
Reset	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0

MMDCx_MDCFG3LP field descriptions

Field	Description
31–22 Reserved	This read-only field is reserved and always has the value 0.
21–16 RC_LP	Active to Active or Refresh command period (same bank). This field is valid only for LPDDR2/3 memories 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x3E 63 clocks 0x3F Reserved

Table continues on the next page...

MMDCx_MDCFG3LP field descriptions (continued)

Field	Description
15–12 Reserved	This read-only field is reserved and always has the value 0.
11–8 tRCD_LP	Active command to internal read or write delay time (same bank). This field is valid only for LPDDR2 memories 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF Reserved
7–4 tRPpb_LP	Precharge (per bank) command period (same bank). This field is valid only for LPDDR2 memories 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF Reserved
tRPab_LP	Precharge (all banks) command period. This field is valid only for LPDDR2 memories 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF Reserved

22.11.13 MMDC Core MR4 Derating Register (MMDCx_MDMR4)

This register is relevant only for LPDDR2 mode. It is used to dynamically change certain values depending on MR4 read result, which is based on memory temperature sensor result.

Address: 40AB_0000h base + 3Ch offset = 40AB_003Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							tRRD_DE	tRP_DE	tRAS_DE	tRC_DE	tRCD_DE	0		UPDATE_DE_ACK	UPDATE_DE_REQ
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MDMR4 field descriptions

Field	Description
31–9 Reserved	This read-only field is reserved and always has the value 0.
8 tRRD_DE	tRRD derating value. 0 Original tRRD is used. 1 tRRD is derated in 1 cycle.
7 tRP_DE	tRP derating value. 0 Original tRP is used. 1 tRP is derated in 1 cycle.
6 tRAS_DE	tRAS derating value. 0 Original tRAS is used. 1 tRAS is derated in 1 cycle.
5 tRC_DE	tRC derating value. 0 Original tRC is used. 1 tRC is derated in 1 cycle.
4 tRCD_DE	tRCD derating value. 0 Original tRCD is used. 1 tRCD is derated in 1 cycle.
3–2 Reserved	This read-only field is reserved and always has the value 0.
1 UPDATE_DE_ACK	Update Derated Values Acknowledge. This read only bit will be cleared upon UPDATE_DE_REQ assertion and will be set after the new values are taken.
0 UPDATE_DE_REQ	Update Derated Values Request. This read modify write field is automatically cleared after the request is issued. 0 Do nothing. 1 Request to update the following values: tRRD, tRCD, tRP, tRC, tRAS and refresh related fields(MDREF register): REF_CNT, REF_SEL, REFR

22.11.14 MMDC Core Address Space Partition Register (MMDCx_MDASP)

This register defines the partitioning between chip select 0 and chip select 1. For further information see Chip select settings .

Address: 40AB_0000h base + 40h offset = 40AB_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																									CS0_END						
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

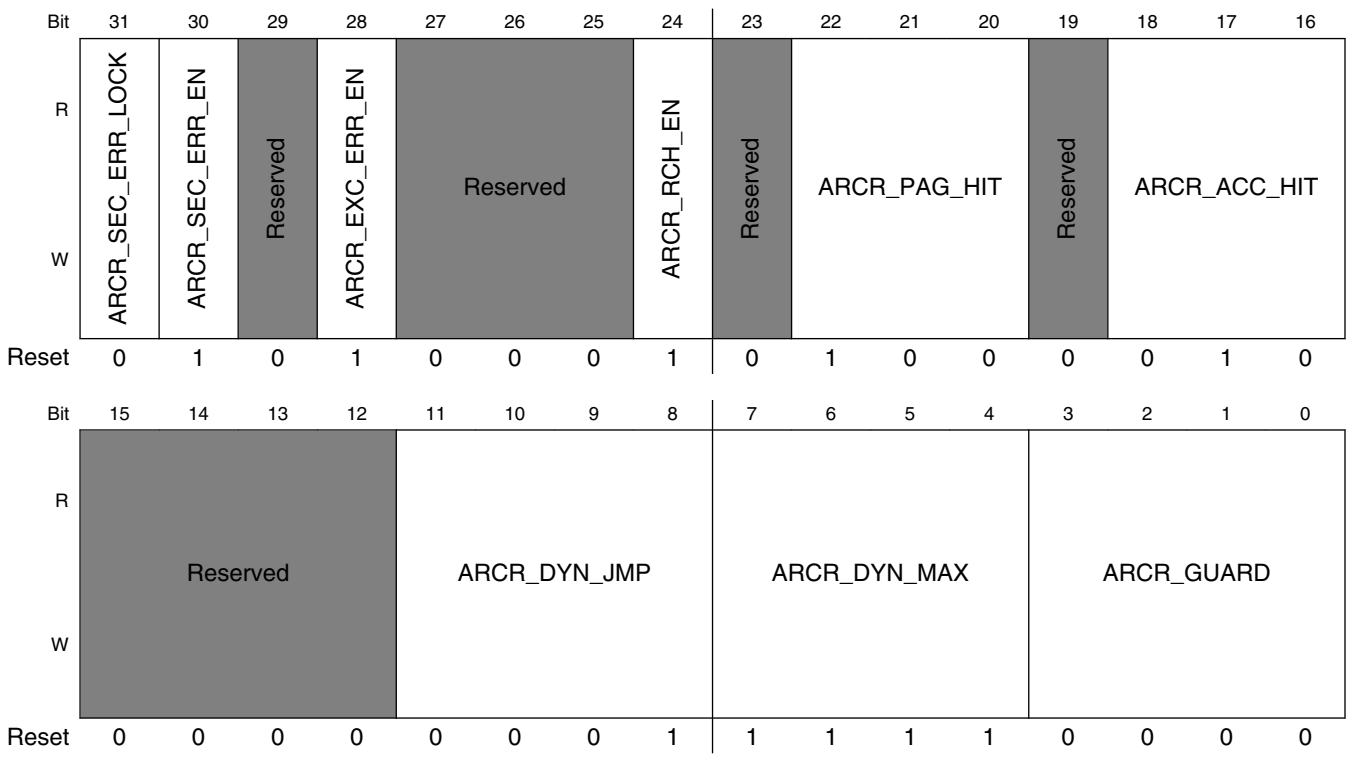
MMDCx_MDASP field descriptions

Field	Description
31–7 Reserved	This read-only field is reserved and always has the value 0.
CS0_END	Defines the absolute last address associated with CS0 with increments of 256Mb. CS0_END=AXI_ADDRESS[31:25] bits. MMDC0_MDASP[CS0_END] should be set to DDR_CS_SIZE/32M + 0x3f (channel 0 base address begins at 0x60000000)

22.11.15 MMDC Core AXI Reordering Control Register (MMDCx_MAARCR)

This register determines the values of the weights used for the re-ordering arbitration engine. For further information see Performance.

Address: 40AB_0000h base + 400h offset = 40AB_0400h



MMDCx_MAARCR field descriptions

Field	Description
31 ARCR_SEC_ ERR_LOCK	Once set, this bit locks ARCR_SEC_ERR_EN and prevents from its updating. This bit can be only cleared by reset Default value is 0x0 - encoding 0 (unlocked) 0 ARCR_SEC_ERR_EN is unlocked, so can be updated any moment 1 ARCR_SEC_ERR_EN is locked, so it can't be updated
30 ARCR_SEC_ ERR_EN	This bit defines whether security read/write access violation result in SLV Error response or in OKAY response Default value is 0x1 - encoding 1(response is SLV Error, rresp/bresp=2'b10) 0 security violation results in OKAY response (rresp/bresp=2'b00) 1 security violation results in SLAVE Error response (rresp/bresp=2'b10)

Table continues on the next page...

MMDCx_MAARCR field descriptions (continued)

Field	Description
29 -	This field is reserved. Reserved
28 ARCR_EXC_ERR_EN	This bit defines whether exclusive read/write access violation of AXI 6.2.4 rule result in SLV Error response or in OKAY response Default value is 0x1 - encoding 1(response is SLV Error) 0 violation of AXI exclusive rules (6.2.4) result in OKAY response (rresp/bresp=2'b00) 1 violation of AXI exclusive rules (6.2.4) result in SLAVE Error response (rresp/bresp=2'b10)
27–25 -	This field is reserved. Reserved
24 ARCR_RCH_EN	This bit defines whether Real time channel is activated and bypassed all other pending accesses, So accesses with QoS=='F' will be granted the highest priority in the optimization/reordering mechanism Default value is 0x1 - encoding 1 (Enabled) 0 normal prioritization, no bypassing 1 accesses with QoS=='F' bypass the arbitration
23 -	This field is reserved. Reserved
22–20 ARCR_PAG_HIT	ARCR Page Hit Rate. This value will be added by the optimization/reordering mechanism to any pending access that is targeted to an open DDR row. Default value of ARCR_PAG_HIT is 0x00100 - encoding 4.
19 -	This field is reserved. Reserved
18–16 ARCR_ACC_HIT	ARCR Access Hit Rate. This value will be added by the optimization/reordering mechanism to any pending access that has the same access type (read/write) as the previous access. Default value of is ARCR_ACC_HIT 0x0010 - encoding 2.
15–12 -	This field is reserved. Reserved
11–8 ARCR_DYN_JMP	ARCR Dynamic Jump. Each time an access is not chosen by the optimization/reordering mechanism then its dynamic score will be incremented by ARCR_DYN_JMP value. **NOTE:** Setting ARCR_DYN_JMP may cause starvation of low priority accesses **NOTE:** ARCR_DYN_JMP must be smaller than ARCR_DYN_MAX Default ARCR_DYN_JMP value is 0x0001 - encoding 1
7–4 ARCR_DYN_MAX	ARCR Dynamic Maximum. ARCR_DYN_MAX is the maximum dynamic score value that each access inside the optimization/reordering mechanism can get. 0000 0 0001 1 1111 15 (default)
ARCR_GUARD	ARCR Guard. After an access reached the maximum dynamic score value, it will wait additional ARCR_GUARD arbitration cycles and then will gain the highest priority in the optimization/reordering mechanism. 0000 15 (default) 0001 16 1111 30

22.11.16 MMDC Core Power Saving Control and Status Register (MMDCx_MAPSR)

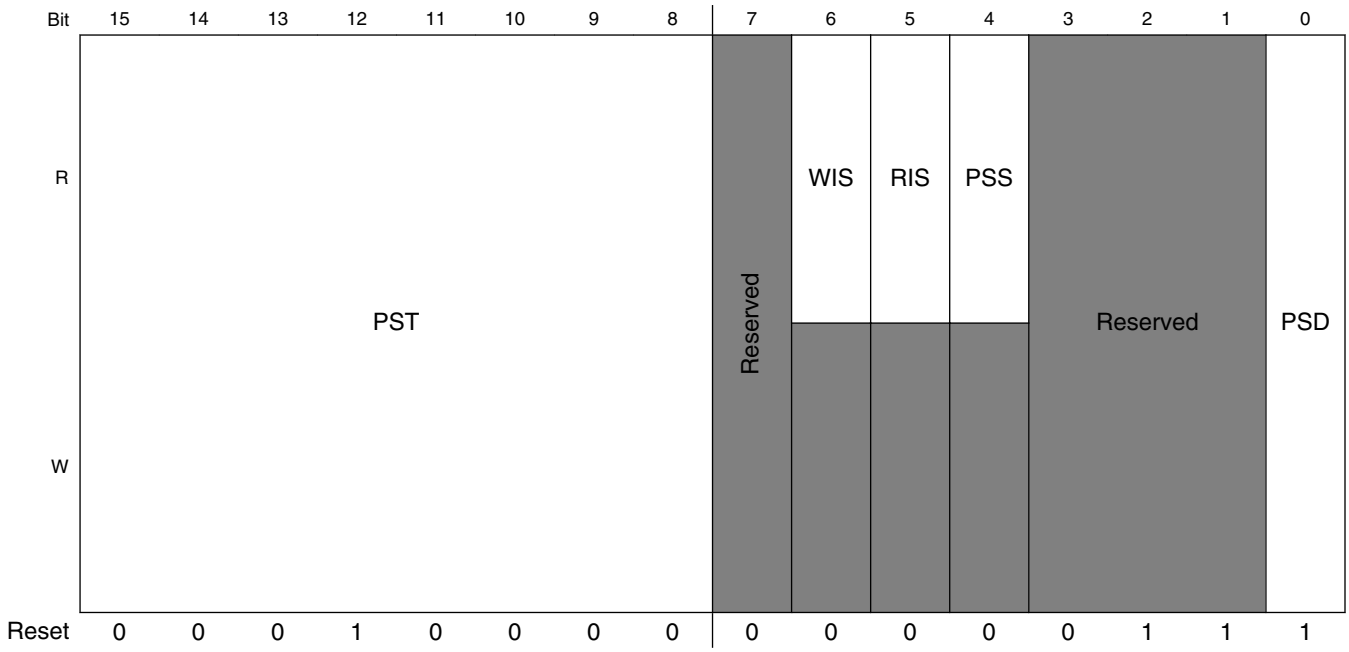
The MAPSR determines the power saving features of MMDC. For further information see Power Saving and Clock Frequency Change modes .

NOTE

See chip-specific information for implementation on your device.

Address: 40AB_0000h base + 404h offset = 40AB_0404h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved						DVACK	LPACK	Reserved		DVFS	LPMD	Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



MMDCx_MAPSR field descriptions

Field	Description
31–26 -	This field is reserved. Reserved
25 DVACK	DVFS/Self-Refresh acknowledge. This read only bit indicates whether a DVFS/self-refresh acknowledge was asserted and that the MMDC is in self-refresh mode.
24 LPACK	General low-power acknowledge. This read only bit indicates whether a low-power acknowledge was asserted and that MMDC is in self-refresh mode
23–22 -	This field is reserved. Reserved
21 DVFS	DVFS/Self-Refresh request. Software request for DVFS/self-refresh. Assertion of this bit will initiate a self-refresh entry sequence. 0 no DVFS/Self-Refresh entry request 1 DVFS/Self-Refresh entry request
20 LPMD	General LPMD request. Software request for LPMD. Assertion of this bit will yield in self-refresh entry sequence 0 no lpmd request 1 lpmd request
19–16 -	This field is reserved. Reserved
15–8 PST	Automatic Power saving timer. Valid only when PSD is set to "0". When the MMDC is idle for amount of cycles specified in that field then the DDR device will be entered automatically into self-refresh mode. The real value which is used is register-value multiplied by 64. 00000000 Reserved - this value is forbidden. 00000001 timer is configured to 64 clock cycles.

Table continues on the next page...

MMDCx_MAPSR field descriptions (continued)

Field	Description
	00000010 timer is configured to 128 clock cycles. 00010000 (Default)- 1024 clock cycles. 11111111 timer clock is configured to 16320 clock cycles.
7 -	This field is reserved. Reserved.
6 WIS	Write Idle Status. This read only bit indicates whether write request buffer is idle (empty) or not. 0 not idle 1 idle
5 RIS	Read Idle Status. This read only bit indicates whether read request buffer is idle (empty) or not. 0 not idle 1 idle
4 PSS	Power Saving Status. This read only bit indicates whether the MMDC is in automatic power saving mode. 0 not in power saving mode 1 power saving mode
3–1 -	This field is reserved. Reserved.
0 PSD	Automatic Power Saving Disable. When the value of PSD is "0" (i.e. automatic power saving is enabled) then the PST is activated and MMDC will enter automatically to self-refresh while the number of idle cycle reached. **NOTE:** This bit must be disabled (i.e. set to "1") during calibration process 0 power saving enabled 1 power saving disabled (default)

22.11.17 MMDC Core Exclusive ID Monitor Register0 (MMDCx_MAEXIDR0)

This register defines the ID to be monitored for exclusive accesses of monitor0 and monitor1. For further information see Exclusive accesses handling .

Address: 40AB_0000h base + 408h offset = 40AB_0408h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EXC_ID_MONITOR1																EXC_ID_MONITOR0															
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MAEXIDR0 field descriptions

Field	Description
31–16 EXC_ID_ MONITOR1	This field defines ID for Exclusive monitor#1. Default value is 0x0020
EXC_ID_ MONITOR0	This field defines ID for Exclusive monitor#0. Default value is 0x0000

22.11.18 MMDC Core Exclusive ID Monitor Register1 (MMDCx_MAEXIDR1)

This register defines the ID to be monitored for exclusive accesses of monitor2 and monitor3. For further information see Exclusive accesses handling .

Address: 40AB_0000h base + 40Ch offset = 40AB_040Ch

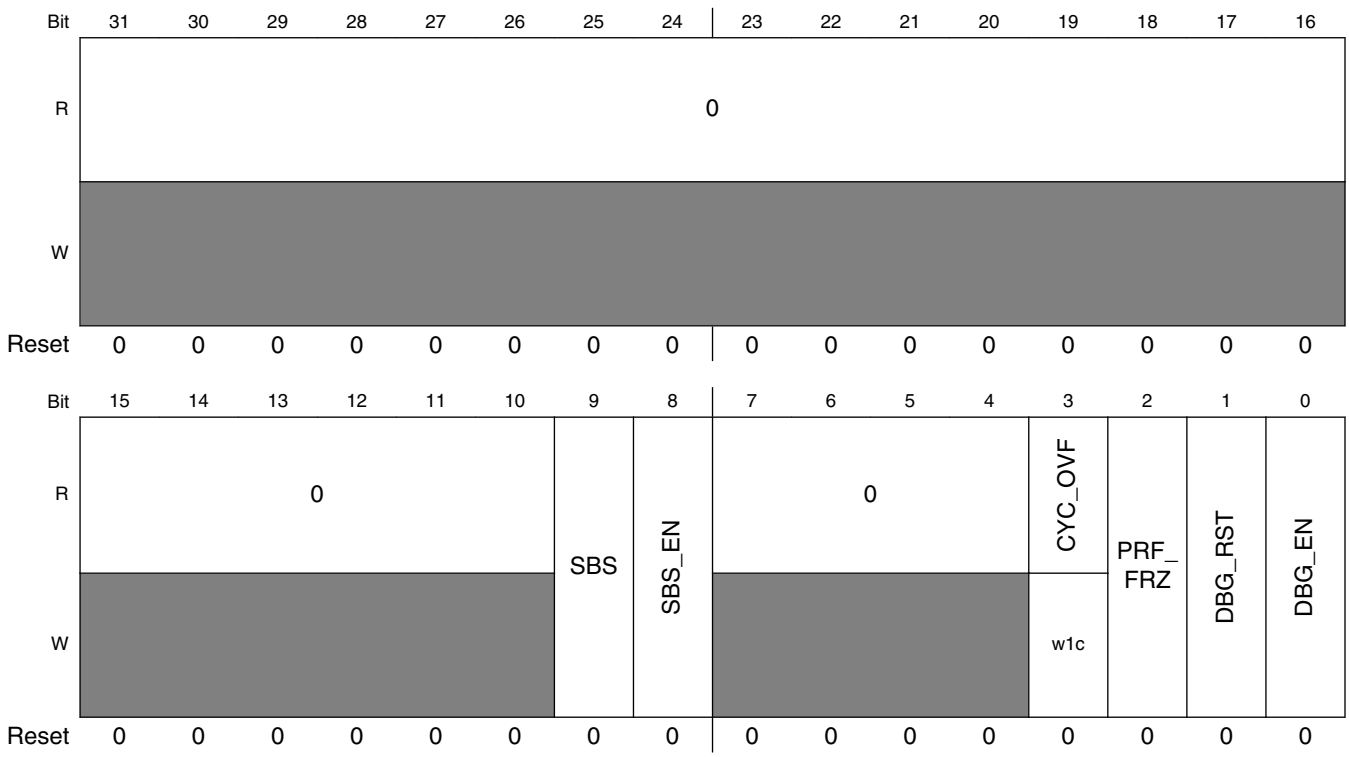
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EXC_ID_MONITOR3																EXC_ID_MONITOR2															
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

MMDCx_MAEXIDR1 field descriptions

Field	Description
31–16 EXC_ID_ MONITOR3	This field defines ID for Exclusive monitor#3. Default value is 0x0060
EXC_ID_ MONITOR2	This field defines ID for Exclusive monitor#2. Default value is 0x0040

22.11.19 MMDC Core Debug and Profiling Control Register 0 (MMDCx_MADPCR0)

Address: 40AB_0000h base + 410h offset = 40AB_0410h



MMDCx_MADPCR0 field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9 SBS	Step By Step trigger. If SBS_EN is set to "1" then dispatching AXI pending access toward the DDR will done only if this bit is set to "1", otherwise no access will be dispatched toward the DDR. This bit is cleared when the pending access has been issued toward the DDR device. 1 Launch AXI pending access toward the DDR 0 No access will be launched toward the DDR
8 SBS_EN	Step By Step debug Enable. Enable step by step mode. Every time this mechanism is enabled then setting SBS to "1" will dispatch one pending AXI access to the DDR and in parallel its attributes will be observed in the status registers (MASBS0 and MASBS1). For further information see Step By Step (SBS) software monitor . 0 disable 1 enable
7–4 Reserved	This read-only field is reserved and always has the value 0.
3 CYC_OVF	Total Profiling Cycles Count Overflow. When profiling mechanism is enabled (DBG_EN is set to "1") then this bit is asserted when overflow of CYC_COUNT occurred. Cleared by writing 1 to it.

Table continues on the next page...

MMDCx_MADPCR0 field descriptions (continued)

Field	Description
	0 no overflow 1 overflow
2 PRF_FRZ	Profiling freeze. When this bit is asserted then the profiling mechanism will be froze and the associated status registers (MADPSR0-MADPSR5) will hold the current profiling values. 0 profiling counters are not frozen 1 profiling counters are frozen
1 DBG_RST	Debug and Profiling Reset. Reset all debug and profiling counters and components. 0 no reset 1 reset
0 DBG_EN	Debug and Profiling Enable. Enable debug and profiling mechanism. When this bit is asserted then the MMDC will perform a profiling based on the ID that is configured to MADPCR1. Upon assertion of PRF_FRZ the profiling will be froze and the profiling results will be sampled to the status registers (MADPSR0-MADPSR5). For further information see MMDC Profiling. 0 disable (default) 1 enable

22.11.20 MMDC Core Debug and Profiling Control Register 1 (MMDCx_MADPCR1)

Address: 40AB_0000h base + 414h offset = 40AB_0414h

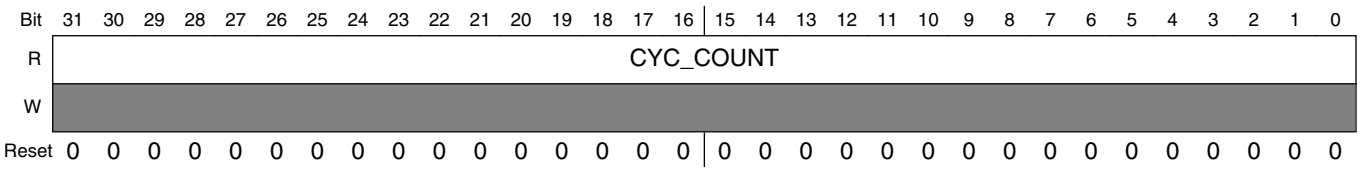
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRF_AXI_ID_MASK																PRF_AXI_ID															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MADPCR1 field descriptions

Field	Description
31–16 PRF_AXI_ID_ MASK	Profiling AXI ID Mask. AXI ID bits which masked by this value are chosen for profiling. 1 AXI ID specific bit is chosen for profiling 0 AXI ID specific bit is ignored (don't care)
PRF_AXI_ID	Profiling AXI ID. AXI IDs that match a bit-wise AND logic operation between PRF_AXI_ID and PRF_AXI_ID_MASK are chosen for profiling. Default value is 0x0, to choose any ID-s for profiling

22.11.21 MMDC Core Debug and Profiling Status Register 0 (MMDCx_MADPSR0)

Address: 40AB_0000h base + 418h offset = 40AB_0418h



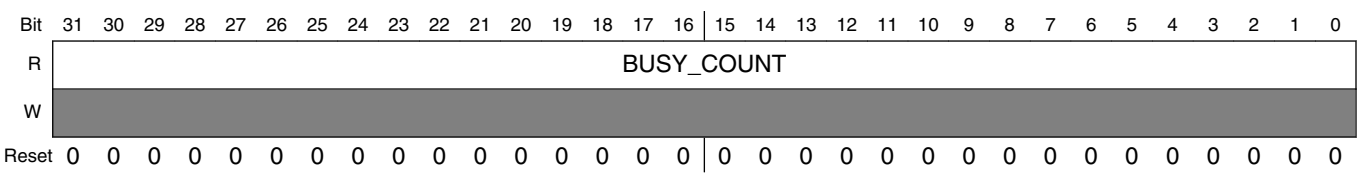
MMDCx_MADPSR0 field descriptions

Field	Description
CYC_COUNT	Total Profiling cycle Count. This field reflects the total cycle count in case the profiling mechanism is enabled from assertion of DBG_EN and until PRF_FRZ is asserted

22.11.22 MMDC Core Debug and Profiling Status Register 1 (MMDCx_MADPSR1)

The register reflects the total cycles during which the MMDC state machines were busy (both writes and reads). This information can be used for DDR Utilization calculation.

Address: 40AB_0000h base + 41Ch offset = 40AB_041Ch



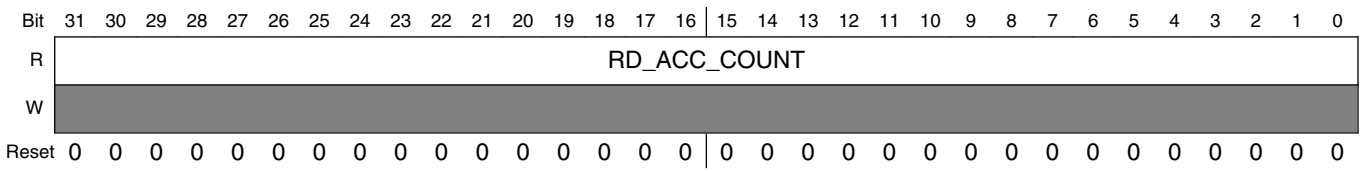
MMDCx_MADPSR1 field descriptions

Field	Description
BUSY_COUNT	Profiling Busy Cycles Count. This field reflects the total number of cycles where the MMDC read and write state machines were busy during the profiling period. Can be used for DDR utilization calculations. Busy cycles are any MMDC clock cycles where the internal state machine is not idle. If any read or write requests are pending in the FIFOs, the MMDC is not idle.

22.11.23 MMDC Core Debug and Profiling Status Register 2 (MMDCx_MADPSR2)

This register reflects the total number of read accesses (per AXI ID) toward MMDC.

Address: 40AB_0000h base + 420h offset = 40AB_0420h



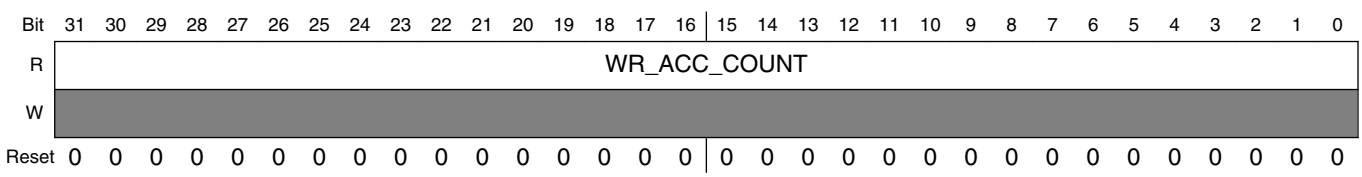
MMDCx_MADPSR2 field descriptions

Field	Description
RD_ACC_ COUNT	Profiling Read Access Count. This register reflects the total number of read accesses (per AXI ID) toward MMDC.

22.11.24 MMDC Core Debug and Profiling Status Register 3 (MMDCx_MADPSR3)

This register reflects the total number of write accesses (per AXI ID) toward MMDC.

Address: 40AB_0000h base + 424h offset = 40AB_0424h



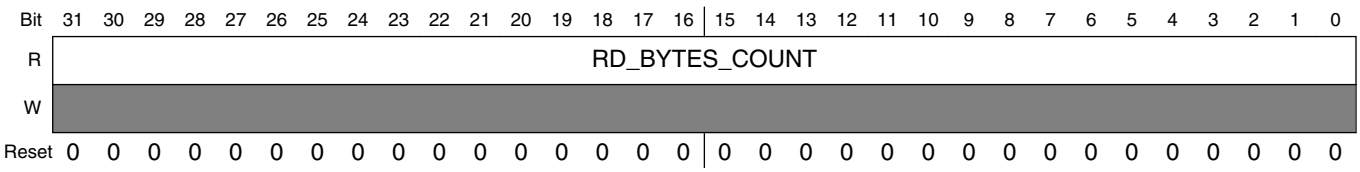
MMDCx_MADPSR3 field descriptions

Field	Description
WR_ACC_ COUNT	Profiling Write Access Count. This register reflects the total number of write accesses (per AXI ID) toward MMDC.

22.11.25 MMDC Core Debug and Profiling Status Register 4 (MMDCx_MADPSR4)

This register reflects the total number of bytes that were transferred during read access (per AXI ID) toward MMDC.

Address: 40AB_0000h base + 428h offset = 40AB_0428h



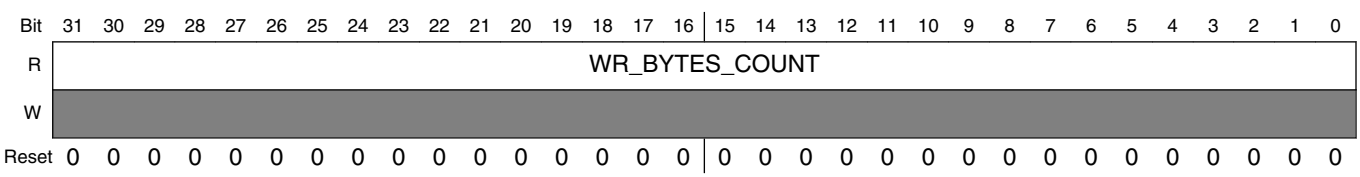
MMDCx_MADPSR4 field descriptions

Field	Description
RD_BYTES_ COUNT	Profiling Read Bytes Count. This register reflects the total number of bytes that were transferred during read access (per AXI ID) toward MMDC.

22.11.26 MMDC Core Debug and Profiling Status Register 5 (MMDCx_MADPSR5)

This register reflects the total number of bytes that were transferred during write access (per AXI ID) toward MMDC.

Address: 40AB_0000h base + 42Ch offset = 40AB_042Ch

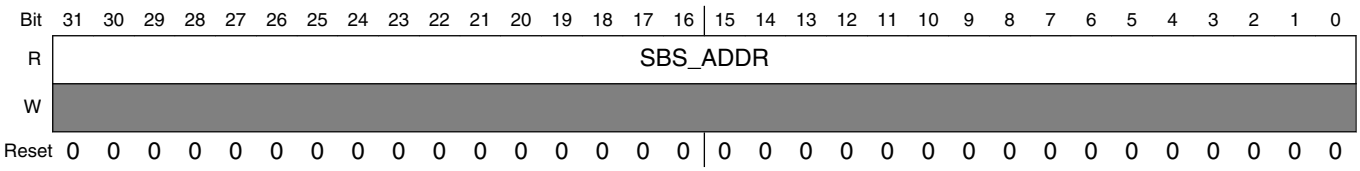


MMDCx_MADPSR5 field descriptions

Field	Description
WR_BYTES_ COUNT	Profiling Write Bytes Count. This register reflects the total number of bytes that were transferred during write access (per AXI ID) toward MMDC.

22.11.27 MMDC Core Step By Step Address Register (MMDCx_MASBS0)

Address: 40AB_0000h base + 430h offset = 40AB_0430h

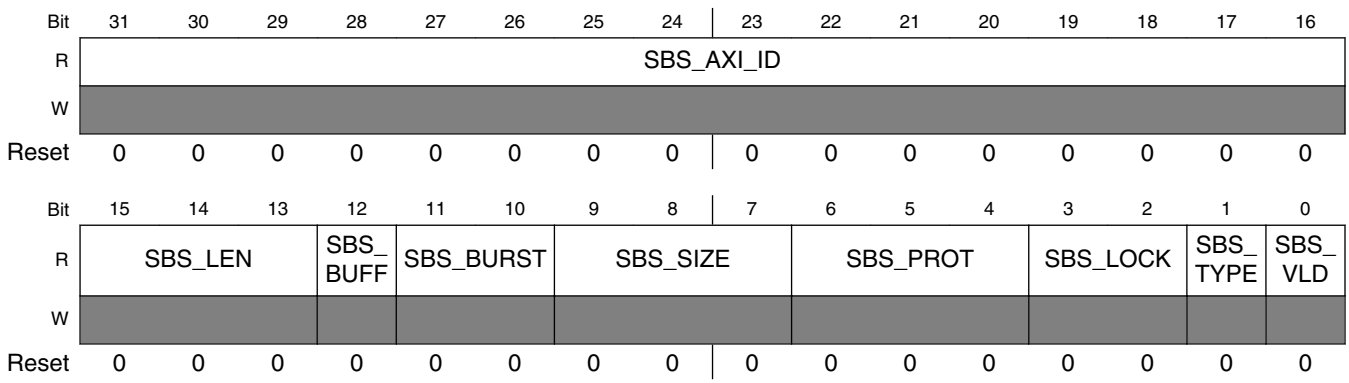


MMDCx_MASBS0 field descriptions

Field	Description
SBS_ADDR	Step By Step Address. These bits reflect the address of the pending request in case of step by step mode.

22.11.28 MMDC Core Step By Step Address Attributes Register (MMDCx_MASBS1)

Address: 40AB_0000h base + 434h offset = 40AB_0434h



MMDCx_MASBS1 field descriptions

Field	Description
31–16 SBS_AXI_ID	Step By Step AXI ID. These bits reflect the AXI ID of the pending request in case of step by step mode.
15–13 SBS_LEN	Step By Step Length. These bits reflect the AXI LENGTH of the pending request in case of step by step mode. 000 burst of length 1 001 burst of length 2 111 burst of length 8
12 SBS_BUFF	Step By Step Buffered. This bit reflect the AXI CACHE[0] of the pending request in case of step by step mode. Relevant only for write requests

Table continues on the next page...

MMDCx_MASBS1 field descriptions (continued)

Field	Description
11–10 SBS_BURST	Step By Step Burst. These bits reflect the AXI BURST of the pending request in case of step by step mode. 00 FIXED 01 INCR burst 10 WRAP burst 11 reserved
9–7 SBS_SIZE	Step By Step Size. These bits reflect the AXI SIZE of the pending request in case of step by step mode. 000 8 bits 001 16 bits 010 32 bits 011 64 bits 100 128bits 101-111 Reserved
6–4 SBS_PROT	Step By Step Protection. These bits reflect the AXI PROT of the pending request in case of step by step mode.
3–2 SBS_LOCK	Step By Step Lock. These bits reflect the AXI LOCK of the pending request in case of step by step mode.
1 SBS_TYPE	Step By Step Request Type. These bits reflect the type (read/write) of the pending request in case of step by step mode. 0 write 1 read
0 SBS_VLD	Step By Step Valid. This bit reflects whether there is a pending request in case of step by step mode. 0 not valid 1 valid

22.11.29 MMDC Core General Purpose Register (MMDCx_MAGENP)

This register is a general 32 bit read/write register.

Address: 40AB_0000h base + 440h offset = 40AB_0440h

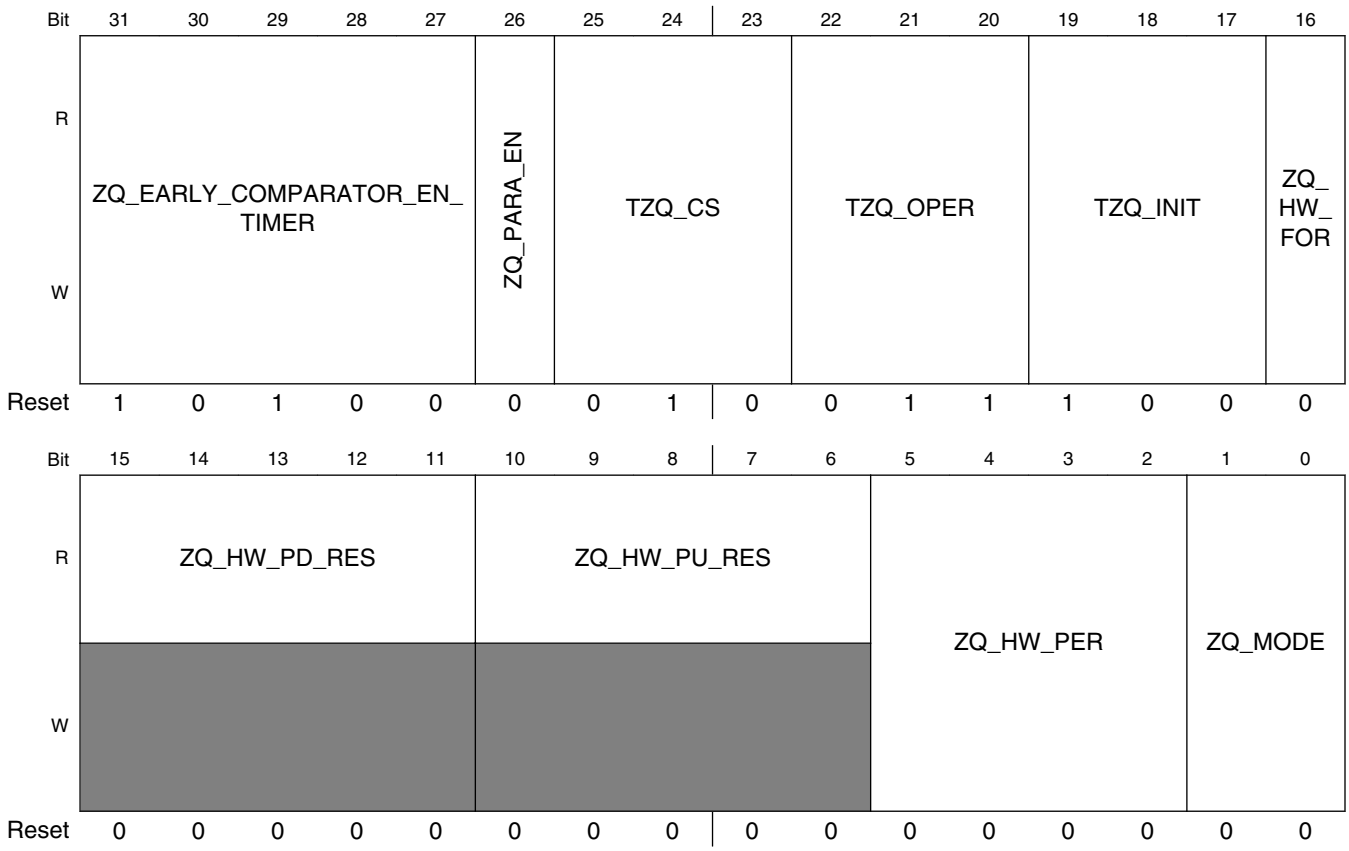
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GP31_GP0																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MAGENP field descriptions

Field	Description
GP31_GP0	General purpose read/write bits.

22.11.30 MMDC PHY ZQ HW control register (MMDCx_MPZQHWCTRL)

Address: 40AB_0000h base + 800h offset = 40AB_0800h



MMDCx_MPZQHWCTRL field descriptions

Field	Description
31–27 ZQ_EARLY_COMPARATOR_EN_TIMER	ZQ early comparator enable timer. This timer defines the interval between the warming up of the comparator of the i.MX ZQ calibration pad and the beginning of the ZQ calibration process with the pad 0x0 - 0x6 Reserved 0x7 8 cycles 0x14 21 cycles (Default) 0x1E 31 cycles 0x1F 32 cycles
26 ZQ_PARA_EN	Device ZQ calibration parallel enable. **NOTE:** If only one chip select is enabled, parallel mode (ZQ_PARA_EN=1) must be selected. 0 Device ZQ calibration is done in serial (CS0 first and then CS1). 1 ZQ calibration of both CS is done in parallel

Table continues on the next page...

MMDCx_MPZQHWCTRL field descriptions (continued)

Field	Description
25–23 TZQ_CS	Device ZQ short time. This field holds the number of cycles that are required by the external DDR device to perform ZQ short calibration. Upon driving the command to the DDR device then no further accesses will be issued to the DDR device till satisfying that time. **NOTE:** In LPDDR2 the ZQ short time is taken from MPZQLP2CTL[ZQ_LP2_HW_ZQCS] **NOTE:** This field should not be update during ZQ calibration. 000 Reserved 001 Reserved 010 128 cycles (Default) 011 256 cycles 100 512 cycles 101 1024 cycles 110- 111 Reserved
22–20 TZQ_OPER	Device ZQ long/oper time. This field holds the number of cycles that are required by the external DDR device to perform ZQ long calibration except the first ZQ long command that is issued after reset. Upon driving the command to the DDR device then no further accesses will be issued to the DDR device till satisfying that time. **NOTE:** In LPDDR2/3 the ZQ oper time is taken from MPZQLP2CTL[ZQ_LP2_HW_ZQCL] **NOTE:** This field should not be update during ZQ calibration. 000 Reserved 001 Reserved 010 128 cycles 011 256 cycles - Default 100 512 cycles 101 1024 cycles 110- 111 Reserved
19–17 TZQ_INIT	Device ZQ long/init time. This field holds the number of cycles that are required by the external DDR device to perform ZQ long calibration right after reset. Upon driving the command to the DDR device then no further accesses will be issued to the DDR device till satisfying that time. **NOTE:** In LPDDR2 the ZQ init time is taken from MPZQLP2CTL[ZQ_LP2_HW_ZQINIT] **NOTE:** This field should not be update during ZQ calibration. 000 Reserved 001 Reserved 010 128 cycles 011 256 cycles 100 512 cycles - Default 101 1024 cycles 110- 111 Reserved
16 ZQ_HW_FOR	Force ZQ automatic calibration process with the i.MX ZQ calibration pad. When this bit is asserted then the MMDC will issue one ZQ automatic calibration process with the i.MX ZQ calibration pad. It is the user responsibility to make sure that all the accesses to DDR will be finished before asserting this bit using CON_REQ/CON_ACK mechanism. HW will negate this bit upon completion of the ZQ calibration process. Upon negation of this bit the ZQ HW calibration pull-up and pull-down results (ZQ_HW_PU_RES and ZQ_HW_PD_RES respectively) are valid

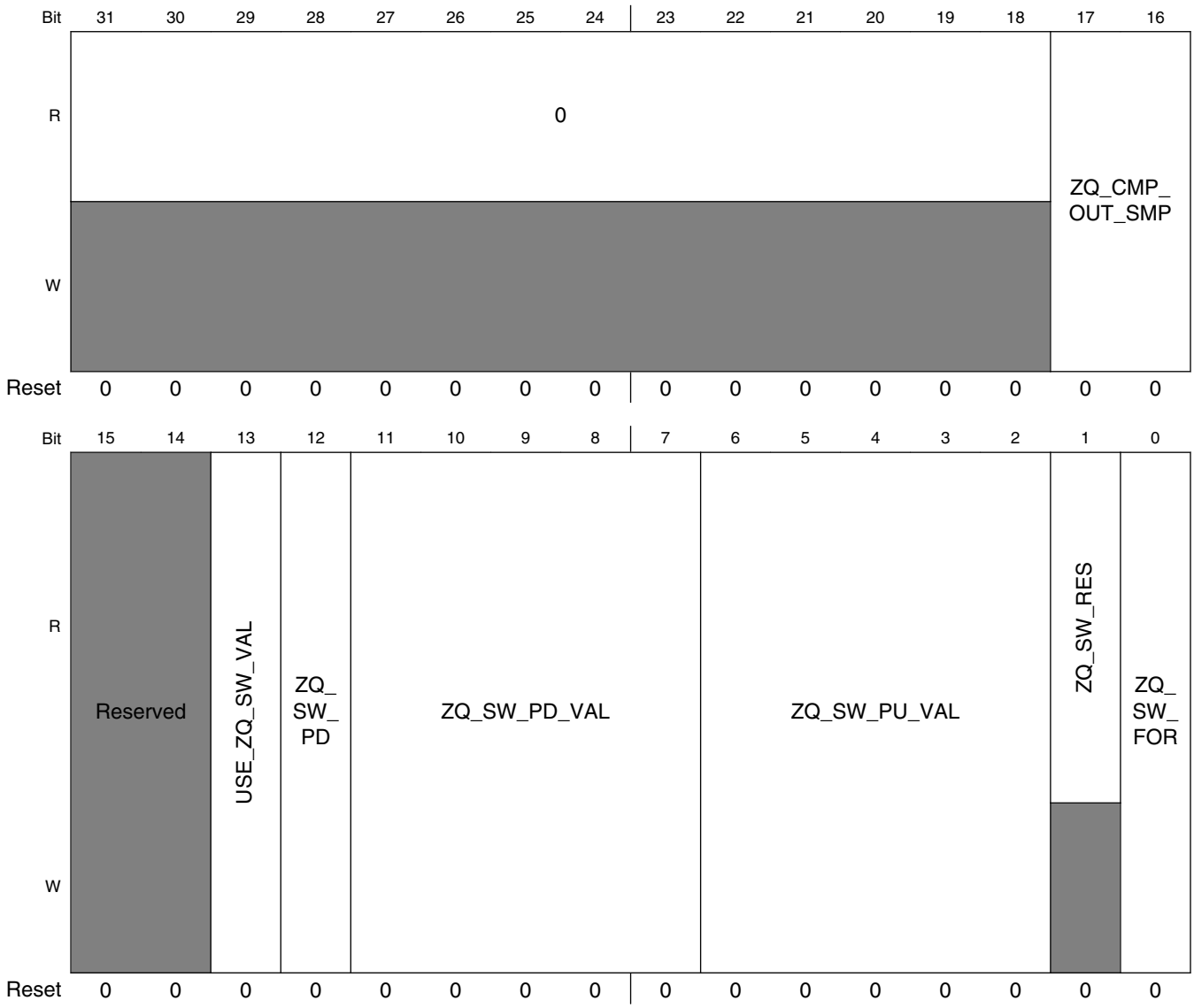
Table continues on the next page...

MMDCx_MPZQHWCTRL field descriptions (continued)

Field	Description
	NOTE: In order to enable this bit ZQ_MODE must be set to either "1" or "3"
15–11 ZQ_HW_PD_RES	ZQ HW calibration pull-down result. This field holds the pull-down resistor value calculated at the end of the ZQ automatic calibration process with the i.MX ZQ calibration pad. **NOTE:** An offset can be applied to this result, see MMDC_MPPDCMPR2[ZQ_PD_OFFSET]. 00000 Max. resistance. 11111 Min. resistance.
10–6 ZQ_HW_PU_RES	ZQ automatic calibration pull-up result. This field holds the pull-up resistor value calculated at the end of the ZQ automatic calibration process with the i.MX ZQ calibration pad. **NOTE:** An offset can be applied to this result, see MMDC_MPPDCMPR2[ZQ_PU_OFFSET] 00000 Min. resistance. 11111 Max. resistance.
5–2 ZQ_HW_PER	ZQ periodic calibration time. This field determines how often the periodic ZQ calibration is performed. This field is applied for both ZQ short calibration and ZQ automatic calibration process with i.MX ZQ calibration pad. Whenever this timer is expired then according to ZQ_MODE the ZQ automatic calibration process with the i.MX ZQ calibration pad will be issued and/or short/long command will be issued to the external DDR device. This field is ignored if ZQ_MODE equals "00" 0000 ZQ calibration is performed every 1 ms. 0001 ZQ calibration is performed every 2 ms. 0010 ZQ calibration is performed every 4 ms. 1010 ZQ calibration is performed every 1 sec. 1110 ZQ calibration is performed every 16 sec. 1111 ZQ calibration is performed every 32 sec.
ZQ_MODE	ZQ calibration mode: 0x0 No ZQ calibration is issued. (Default) 0x1 ZQ calibration is issued to i.MX ZQ calibration pad together with ZQ long command to the external DDR device only when exiting self refresh. 0x2 ZQ calibration command long/short is issued only to the external DDR device periodically and when exiting self refresh 0x3 ZQ calibration is issued to i.MX ZQ calibration pad together with ZQ calibration command long/short to the external DDR device periodically and when exiting self refresh

22.11.31 MMDC PHY ZQ SW control register (MMDCx_MPZQSWCTRL)

Address: 40AB_0000h base + 804h offset = 40AB_0804h



MMDCx_MPZQSWCTRL field descriptions

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value 0.
17–16 ZQ_CMP_OUT_SMP	Defines the amount of cycles between driving the ZQ signals to the ZQ pad and till sampling the comparator enable output while performing ZQ calibration process with the i.MX ZQ calibration pad 00 7 cycles 01 15 cycles

Table continues on the next page...

MMDCx_MPZQSWCTRL field descriptions (continued)

Field	Description
	10 23 cycles 11 31 cycles
15–14 -	This field is reserved. Reserved
13 USE_ZQ_SW_VAL	Use SW ZQ configured value for I/O pads resistor controls. This bit selects whether ZQ SW value or ZQ HW value will be driven to the I/O pads resistor controls. By default this bit is cleared and MMDC drives the HW ZQ status bits on the resistor controls of the I/O pads. **NOTE:** This bit should not be updated during ZQ calibration. 0 Fields ZQ_HW_PD_VAL & ZQ_HW_PU_VAL will be driven to I/O pads resistor controls. 1 Fields ZQ_SW_PD_VAL & ZQ_SW_PU_VAL will be driven to I/O pads resistor controls.
12 ZQ_SW_PD	ZQ software PU/PD calibration. This bit determines the calibration stage (PU or PD). 0 PU resistor calibration 1 PD resistor calibration
11–7 ZQ_SW_PD_VAL	ZQ software pull-down resistance. This field determines the value of the PD resistor during SW ZQ calibration. 00000 Max. resistance. 11111 Min. resistance.
6–2 ZQ_SW_PU_VAL	ZQ software pull-up resistance. This field determines the value of the PU resistor during SW ZQ calibration. 00000 Min. resistance. 11111 Max. resistance.
1 ZQ_SW_RES	ZQ software calibration result. This bit reflects the ZQ calibration voltage comparator value. 0 Current ZQ calibration voltage is less than VDD/2. 1 Current ZQ calibration voltage is more than VDD/2
0 ZQ_SW_FOR	ZQ SW calibration enable. This bit when asserted enables ZQ SW calibration. HW negates this bit upon completion of the ZQ SW calibration. Upon negation of this bit the ZQ SW calibration result (i.e. ZQ_SW_RES) is valid

22.11.32 MMDC PHY Read DQ Byte0 Delay Register (MMDCx_MPRDDQBY0DL)

This register is used to add fine-tuning adjustment to every bit in the read DQ byte0 relative to the read DQS. This delay is in addition to the read data calibration. If operating in 64-bit mode, there is an identical register that is mapped at the second base address.

Address: 40AB_0000h base + 81Ch offset = 40AB_081Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	rd_dq7_del			0	rd_dq6_del			0	rd_dq5_del			0	rd_dq4_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	rd_dq3_del			0	rd_dq2_del			0	rd_dq1_del			0	rd_dq0_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPRDDQBY0DL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq7_del	Read dqs0 to dq7 delay fine-tuning. This field holds the number of delay units that are added to dq7 relative to dqs0. 000 No change in dq7 delay 001 Add dq7 delay of 1 delay unit 010 Add dq7 delay of 2 delay units. 011 Add dq7 delay of 3 delay units. 100 Add dq7 delay of 4 delay units. 101 Add dq7 delay of 5 delay units. 110 Add dq7 delay of 6 delay units. 111 Add dq7 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 rd_dq6_del	Read dqs0 to dq6 delay fine-tuning. This field holds the number of delay units that are added to dq6 relative to dqs0. 000 No change in dq6 delay 001 Add dq6 delay of 1 delay unit 010 Add dq6 delay of 2 delay units. 011 Add dq6 delay of 3 delay units. 100 Add dq6 delay of 4 delay units. 101 Add dq6 delay of 5 delay units.

Table continues on the next page...

MMDCx_MPRDDQBY0DL field descriptions (continued)

Field	Description
	110 Add dq6 delay of 6 delay units. 111 Add dq6 delay of 7 delay units.
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq5_del	Read dqs0 to dq5 delay fine-tuning. This field holds the number of delay units that are added to dq5 relative to dqs0. 000 No change in dq5 delay 001 Add dq5 delay of 1 delay unit 010 Add dq5 delay of 2 delay units. 011 Add dq5 delay of 3 delay units. 100 Add dq5 delay of 4 delay units. 101 Add dq5 delay of 5 delay units. 110 Add dq5 delay of 6 delay units. 111 Add dq5 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 rd_dq4_del	Read dqs0 to dq4 delay fine-tuning. This field holds the number of delay units that are added to dq4 relative to dqs0. 000 No change in dq4 delay 001 Add dq4 delay of 1 delay unit 010 Add dq4 delay of 2 delay units. 011 Add dq4 delay of 3 delay units. 100 Add dq4 delay of 4 delay units. 101 Add dq4 delay of 5 delay units. 110 Add dq4 delay of 6 delay units. 111 Add dq4 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq3_del	Read dqs0 to dq3 delay fine-tuning. This field holds the number of delay units that are added to dq3 relative to dqs0. 000 No change in dq3 delay 001 Add dq3 delay of 1 delay unit 010 Add dq3 delay of 2 delay units. 011 Add dq3 delay of 3 delay units. 100 Add dq3 delay of 4 delay units. 101 Add dq3 delay of 5 delay units. 110 Add dq3 delay of 6 delay units. 111 Add dq3 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq2_del	Read dqs0 to dq2 delay fine-tuning. This field holds the number of delay units that are added to dq2 relative to dqs0. 000 No change in dq2 delay 001 Add dq2 delay of 1 delay unit

Table continues on the next page...

MMDCx_MPRDDQBY0DL field descriptions (continued)

Field	Description
	010 Add dq2 delay of 2 delay units. 011 Add dq2 delay of 3 delay units. 100 Add dq2 delay of 4 delay units. 101 Add dq2 delay of 5 delay units. 110 Add dq2 delay of 6 delay units. 111 Add dq2 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq1_del	Read dqs0 to dq1 delay fine-tuning. This field holds the number of delay units that are added to dq1 relative to dqs0. 000 No change in dq1 delay 001 Add dq1 delay of 1 delay unit 010 Add dq1 delay of 2 delay units. 011 Add dq1 delay of 3 delay units. 100 Add dq1 delay of 4 delay units. 101 Add dq1 delay of 5 delay units. 110 Add dq1 delay of 6 delay units. 111 Add dq1 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
rd_dq0_del	Read dqs0 to dq0 delay fine-tuning. This field holds the number of delay units that are added to dq0 relative to dqs0. 000 No change in dq0 delay 001 Add dq0 delay of 1 delay unit 010 Add dq0 delay of 2 delay units. 011 Add dq0 delay of 3 delay units. 100 Add dq0 delay of 4 delay units. 101 Add dq0 delay of 5 delay units. 110 Add dq0 delay of 6 delay units. 111 Add dq0 delay of 7 delay units.

22.11.33 MMDC PHY Read DQ Byte1 Delay Register (MMDCx_MPRDDQBY1DL)

This register is used to add fine-tuning adjustment to every bit in the read DQ byte1 relative to the read DQS

Address: 40AB_0000h base + 820h offset = 40AB_0820h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	rd_dq15_del			0	rd_dq14_del			0	rd_dq13_del			0	rd_dq12_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	rd_dq11_del			0	rd_dq10_del			0	rd_dq9_del			0	rd_dq8_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPRDDQBY1DL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq15_del	Read dqs1 to dq15 delay fine-tuning. This field holds the number of delay units that are added to dq15 relative to dqs1. 000 No change in dq15 delay 001 Add dq15 delay of 1 delay unit 010 Add dq15 delay of 2 delay units. 011 Add dq15 delay of 3 delay units. 100 Add dq15 delay of 4 delay units. 101 Add dq15 delay of 5 delay units. 110 Add dq15 delay of 6 delay units. 111 Add dq15 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 rd_dq14_del	Read dqs1 to dq14 delay fine-tuning. This field holds the number of delay units that are added to dq14 relative to dqs1. 000 No change in dq14 delay 001 Add dq14 delay of 1 delay unit 010 Add dq14 delay of 2 delay units. 011 Add dq14 delay of 3 delay units. 100 Add dq14 delay of 4 delay units. 101 Add dq14 delay of 5 delay units. 110 Add dq14 delay of 6 delay units. 111 Add dq14 delay of 7 delay units.

Table continues on the next page...

MMDCx_MPRDDQBY1DL field descriptions (continued)

Field	Description
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq13_del	Read dqs1 to dq13 delay fine-tuning. This field holds the number of delay units that are added to dq13 relative to dqs1. 000 No change in dq13 delay 001 Add dq13 delay of 1 delay unit 010 Add dq13 delay of 2 delay units. 011 Add dq13 delay of 3 delay units. 100 Add dq13 delay of 4 delay units. 101 Add dq13 delay of 5 delay units. 110 Add dq13 delay of 6 delay units. 111 Add dq13 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 rd_dq12_del	Read dqs1 to dq12 delay fine-tuning. This field holds the number of delay units that are added to dq12 relative to dqs1. 000 No change in dq12 delay 001 Add dq12 delay of 1 delay unit 010 Add dq12 delay of 2 delay units. 011 Add dq12 delay of 3 delay units. 100 Add dq12 delay of 4 delay units. 101 Add dq12 delay of 5 delay units. 110 Add dq12 delay of 6 delay units. 111 Add dq12 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq11_del	Read dqs1 to dq11 delay fine-tuning. This field holds the number of delay units that are added to dq11 relative to dqs1. 000 No change in dq11 delay 001 Add dq11 delay of 1 delay unit 010 Add dq11 delay of 2 delay units. 011 Add dq11 delay of 3 delay units. 100 Add dq11 delay of 4 delay units. 101 Add dq11 delay of 5 delay units. 110 Add dq11 delay of 6 delay units. 111 Add dq11 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq10_del	Read dqs1 to dq10 delay fine-tuning. This field holds the number of delay units that are added to dq10 relative to dqs1. 000 No change in dq10 delay 001 Add dq10 delay of 1 delay unit 010 Add dq10 delay of 2 delay units. 011 Add dq10 delay of 3 delay units.

Table continues on the next page...

MMDCx_MPRDDQBY1DL field descriptions (continued)

Field	Description
	100 Add dq10 delay of 4 delay units. 101 Add dq10 delay of 5 delay unit 110 Add dq10 delay of 6 delay units. 111 Add dq10 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq9_del	Read dqs1 to dq9 delay fine-tuning. This field holds the number of delay units that are added to dq9 relative to dqs1. 000 No change in dq9 delay 001 Add dq9 delay of 1 delay unit 010 Add dq9 delay of 2 delay units. 011 Add dq9 delay of 3 delay units. 100 Add dq9 delay of 4 delay units. 101 Add dq9 delay of 5 delay units. 110 Add dq9 delay of 6 delay units. 111 Add dq9 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
rd_dq8_del	Read dqs1 to dq8 delay fine-tuning. This field holds the number of delay units that are added to dq8 relative to dqs1. 000 No change in dq8 delay 001 Add dq8 delay of 1 delay unit 010 Add dq8 delay of 2 delay units. 011 Add dq8 delay of 3 delay units. 100 Add dq8 delay of 4 delay units. 101 Add dq8 delay of 5 delay units. 110 Add dq8 delay of 6 delay units. 111 Add dq8 delay of 7 delay units.

22.11.34 MMDC PHY Read DQ Byte2 Delay Register (MMDCx_MPRDDQBY2DL)

This register is used to add fine-tuning adjustment to every bit in the read DQ byte2 relative to the read DQS

Address: 40AB_0000h base + 824h offset = 40AB_0824h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	rd_dq23_del			0	rd_dq22_del			0	rd_dq21_del			0	rd_dq20_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	rd_dq19_del			0	rd_dq18_del			0	rd_dq17_del			0	rd_dq16_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPRDDQBY2DL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq23_del	Read dqs2 to dq23 delay fine-tuning. This field holds the number of delay units that are added to dq23 relative to dqs2. 000 No change in dq23 delay 001 Add dq23 delay of 1 delay unit 010 Add dq23 delay of 2 delay units. 011 Add dq23 delay of 3 delay units. 100 Add dq23 delay of 4 delay units. 101 Add dq23 delay of 5 delay units. 110 Add dq23 delay of 6 delay units. 111 Add dq23 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 rd_dq22_del	Read dqs2 to dq22 delay fine-tuning. This field holds the number of delay units that are added to dq22 relative to dqs2. 000 No change in dq22 delay 001 Add dq22 delay of 1 delay unit 010 Add dq22 delay of 2 delay units. 011 Add dq22 delay of 3 delay units. 100 Add dq22 delay of 4 delay units. 101 Add dq22 delay of 5 delay units. 110 Add dq22 delay of 6 delay units. 111 Add dq22 delay of 7 delay units.

Table continues on the next page...

MMDCx_MPRDDQBY2DL field descriptions (continued)

Field	Description
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq21_del	Read dqs2 to dq21 delay fine-tuning. This field holds the number of delay units that are added to dq21 relative to dqs2. 000 No change in dq21 delay 001 Add dq21 delay of 1 delay unit 010 Add dq21 delay of 2 delay units. 011 Add dq21 delay of 3 delay units. 100 Add dq21 delay of 4 delay units. 101 Add dq21 delay of 5 delay units. 110 Add dq21 delay of 6 delay units. 111 Add dq21 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 rd_dq20_del	Read dqs2 to dq20 delay fine-tuning. This field holds the number of delay units that are added to dq20 relative to dqs2. 000 No change in dq20 delay 001 Add dq20 delay of 1 delay unit 010 Add dq20 delay of 2 delay units. 011 Add dq20 delay of 3 delay units. 100 Add dq20 delay of 4 delay units. 101 Add dq20 delay of 5 delay units. 110 Add dq20 delay of 6 delay units. 111 Add dq20 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq19_del	Read dqs2 to dq19 delay fine-tuning. This field holds the number of delay units that are added to dq19 relative to dqs2. 000 No change in dq19 delay 001 Add dq19 delay of 1 delay unit 010 Add dq19 delay of 2 delay units. 011 Add dq19 delay of 3 delay units. 100 Add dq19 delay of 4 delay units. 101 Add dq19 delay of 5 delay units. 110 Add dq19 delay of 6 delay units. 111 Add dq19 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq18_del	Read dqs2 to dq18 delay fine-tuning. This field holds the number of delay units that are added to dq18 relative to dqs2. 000 No change in dq18 delay 001 Add dq18 delay of 1 delay unit 010 Add dq18 delay of 2 delay units. 011 Add dq18 delay of 3 delay units.

Table continues on the next page...

MMDCx_MPRDDQBY2DL field descriptions (continued)

Field	Description
	100 Add dq18 delay of 4 delay units. 101 Add dq18 delay of 5 delay units. 110 Add dq18 delay of 6 delay units. 111 Add dq18 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq17_del	Read dqs2 to dq17 delay fine-tuning. This field holds the number of delay units that are added to dq17 relative to dqs2. 000 No change in dq17 delay 001 Add dq17 delay of 1 delay unit 010 Add dq17 delay of 2 delay units. 011 Add dq17 delay of 3 delay units. 100 Add dq17 delay of 4 delay units. 101 Add dq17 delay of 5 delay units. 110 Add dq17 delay of 6 delay units. 111 Add dq17 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
rd_dq16_del	Read dqs2 to dq16 delay fine-tuning. This field holds the number of delay units that are added to dq16 relative to dqs2. 000 No change in dq16 delay 001 Add dq16 delay of 1 delay unit 010 Add dq16 delay of 2 delay units. 011 Add dq16 delay of 3 delay units. 100 Add dq16 delay of 4 delay units. 101 Add dq16 delay of 5 delay units. 110 Add dq16 delay of 6 delay units. 111 Add dq16 delay of 7 delay units.

22.11.35 MMDC PHY Read DQ Byte3 Delay Register (MMDCx_MPRDDQBY3DL)

This register is used to add fine-tuning adjustment to every bit in the read DQ byte3 relative to the read DQS.

The bit assignments and the bit field descriptions for the register are shown below.

Address: 40AB_0000h base + 828h offset = 40AB_0828h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	rd_dq31_del			0	rd_dq30_del			0	rd_dq29_del			0	rd_dq28_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	rd_dq27_del			0	rd_dq26_del			0	rd_dq25_del			0	rd_dq24_del		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPRDDQBY3DL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq31_del	Read dqs3 to dq31 delay fine-tuning. This field holds the number of delay units that are added to dq31 relative to dqs3. 000 No change in dq31 delay 001 Add dq31 delay of 1 delay unit 010 Add dq31 delay of 2 delay units. 011 Add dq31 delay of 3 delay units. 100 Add dq31 delay of 4 delay units. 101 Add dq31 delay of 5 delay units. 110 Add dq31 delay of 6 delay units. 111 Add dq31 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 rd_dq30_del	Read dqs3 to dq30 delay fine-tuning. This field holds the number of delay units that are added to dq30 relative to dqs3. 000 No change in dq30 delay 001 Add dq30 delay of 1 delay unit 010 Add dq30 delay of 2 delay units. 011 Add dq30 delay of 3 delay units. 100 Add dq30 delay of 4 delay units. 101 Add dq30 delay of 5 delay units. 110 Add dq30 delay of 6 delay units. 111 Add dq30 delay of 7 delay units.
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq29_del	Read dqs3 to dq29 delay fine-tuning. This field holds the number of delay units that are added to dq29 relative to dqs3. 000 No change in dq29 delay 001 Add dq29 delay of 1 delay unit 010 Add dq29 delay of 2 delay units. 011 Add dq29 delay of 3 delay units. 100 Add dq29 delay of 4 delay units. 101 Add dq29 delay of 5 delay units. 110 Add dq29 delay of 6 delay units. 111 Add dq29 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

MMDCx_MPRDDQBY3DL field descriptions (continued)

Field	Description
18–16 rd_dq28_del	Read dqs3 to dq28 delay fine-tuning. This field holds the number of delay units that are added to dq28 relative to dqs3. 000 No change in dq28 delay 001 Add dq28 delay of 1 delay unit 010 Add dq28 delay of 2 delay units. 011 Add dq28 delay of 3 delay units. 100 Add dq28 delay of 4 delay units. 101 Add dq28 delay of 5 delay units. 110 Add dq28 delay of 6 delay units. 111 Add dq28 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq27_del	Read dqs3 to dq27 delay fine-tuning. This field holds the number of delay units that are added to dq27 relative to dqs3. 000 No change in dq27 delay 001 Add dq27 delay of 1 delay unit 010 Add dq27 delay of 2 delay units. 011 Add dq27 delay of 3 delay units. 100 Add dq27 delay of 4 delay units. 101 Add dq27 delay of 5 delay units. 110 Add dq27 delay of 6 delay units. 111 Add dq27 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq26_del	Read dqs3 to dq26 delay fine-tuning. This field holds the number of delay units that are added to dq26 relative to dqs3. 000 No change in dq26 delay 001 Add dq26 delay of 1 delay unit 010 Add dq26 delay of 2 delay units. 011 Add dq26 delay of 3 delay units. 100 Add dq26 delay of 4 delay units. 101 Add dq26 delay of 5 delay units. 110 Add dq26 delay of 6 delay units. 111 Add dq26 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq25_del	Read dqs3 to dq25 delay fine-tuning. This field holds the number of delay units that are added to dq25 relative to dqs3. 000 No change in dq25 delay 001 Add dq25 delay of 1 delay unit 010 Add dq25 delay of 2 delay units. 011 Add dq25 delay of 3 delay units. 100 Add dq25 delay of 4 delay units. 101 Add dq25 delay of 5 delay units.

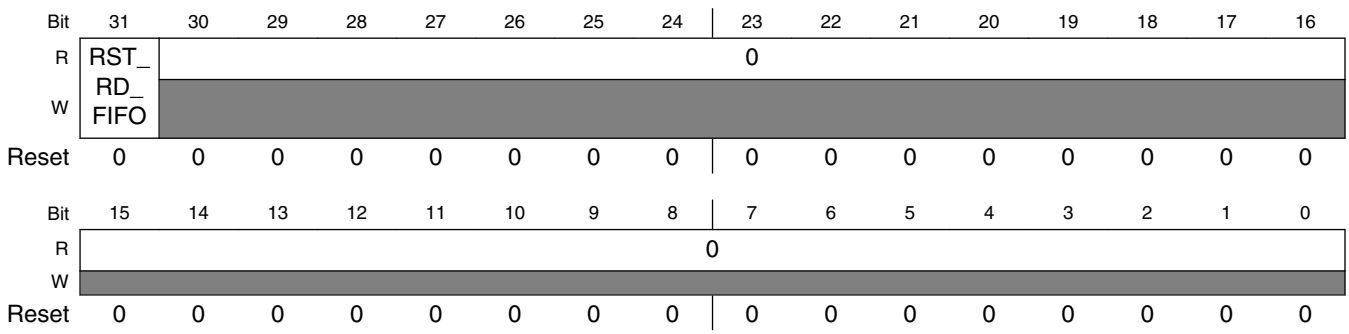
Table continues on the next page...

MMDCx_MPRDDQBY3DL field descriptions (continued)

Field	Description
	110 Add dq25 delay of 6 delay units. 111 Add dq25 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
rd_dq24_del	Read dqs3 to dq24 delay fine-tuning. This field holds the number of delay units that are added to dq24 relative to dqs3. 000 No change in dq24 delay 001 Add dq24 delay of 1 delay unit 010 Add dq24 delay of 2 delay units. 011 Add dq24 delay of 3 delay units. 100 Add dq24 delay of 4 delay units. 101 Add dq24 delay of 5 delay units. 110 Add dq24 delay of 6 delay units. 111 Add dq24 delay of 7 delay units.

22.11.36 MMDC PHY Read DQS Gating Control Register 0 (MMDCx_MPDGCTRL0)

Address: 40AB_0000h base + 83Ch offset = 40AB_083Ch



MMDCx_MPDGCTRL0 field descriptions

Field	Description
31 RST_RD_FIFO	Reset Read Data FIFO and associated pointers. If this bit is asserted then the MMDC resets the read data FIFO and the associated pointers. This bit is self cleared after the FIFO reset is done.
Reserved	This read-only field is reserved and always has the value 0.

22.11.37 MMDC PHY Read delay-lines Configuration Register (MMDCx_MPRDDLCTL)

This register controls read delay-lines functionality; it determines DQS delay relative to the associated DQ read access. The delay-line compensates for process variations and produces a constant delay regardless of the process, temperature and voltage.

Address: 40AB_0000h base + 848h offset = 40AB_0848h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RD_DL_ABS_OFFSET3							0	RD_DL_ABS_OFFSET2						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	RD_DL_ABS_OFFSET1							0	RD_DL_ABS_OFFSET0						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

MMDCx_MPRDDLCTL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 RD_DL_ABS_OFFSET3	Absolute read delay offset for Byte3. This field indicates the absolute delay between read DQS strobe and the read data of Byte3 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (RD_DL_ABS_OFFSET3 / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of (HW_RD_DL_LOW3 + HW_RD_DL_UP3) /2 **NOTE:** Not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 RD_DL_ABS_OFFSET2	Absolute read delay offset for Byte2. This field indicates the absolute delay between read DQS strobe and the read data of Byte2 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (RD_DL_ABS_OFFSET2 / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of (HW_RD_DL_LOW2 + HW_RD_DL_UP2) /2 **NOTE:** Not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.
15 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

MMDCx_MPRDDLCTL field descriptions (continued)

Field	Description
14–8 RD_DL_ABS_OFFSET1	Absolute read delay offset for Byte1. This field indicates the absolute delay between read DQS strobe and the read data of Byte1 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (RD_DL_ABS_OFFSET1 / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of (HW_RD_DL_LOW1 + HW_RD_DL_UP1) /2 **NOTE:** Not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.
7 Reserved	This read-only field is reserved and always has the value 0.
RD_DL_ABS_OFFSET0	Absolute read delay offset for Byte0. This field indicates the absolute delay between read DQS strobe and the read data of Byte0 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (RD_DL_ABS_OFFSET0 / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of (HW_RD_DL_LOW0 + HW_RD_DL_UP0) /2 **NOTE:** Not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.

22.11.38 MMDC PHY Read delay-lines Status Register (MMDCx_MPRDDLST)

This register holds the status of the 4 read delay-lines.

Address: 40AB_0000h base + 84Ch offset = 40AB_084Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RD_DL_UNIT_NUM3							0	RD_DL_UNIT_NUM2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	RD_DL_UNIT_NUM1							0	RD_DL_UNIT_NUM0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPRDDLST field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

MMDCx_MPRDDLST field descriptions (continued)

Field	Description
30–24 RD_DL_UNIT_NUM3	This field reflects the number of delay units that are actually used by read delay-line 3.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 RD_DL_UNIT_NUM2	This field reflects the number of delay units that are actually used by read delay-line 2.
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 RD_DL_UNIT_NUM1	This field reflects the number of delay units that are actually used by read delay-line 1.
7 Reserved	This read-only field is reserved and always has the value 0.
RD_DL_UNIT_NUM0	This field reflects the number of delay units that are actually used by read delay-line 0.

22.11.39 MMDC PHY Write delay-lines Configuration Register (MMDCx_MPWRDLCTL)

This register controls write delay-lines functionality, it determines DQ/DM delay relative to the associated DQS in write access. The delay-line compensates for process variations, and produces a constant delay regardless of the process, temperature and voltage.

Address: 40AB_0000h base + 850h offset = 40AB_0850h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	WR_DL_ABS_OFFSET3							0	WR_DL_ABS_OFFSET2						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WR_DL_ABS_OFFSET1							0	WR_DL_ABS_OFFSET0						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

MMDCx_MPWRDLCTL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

MMDCx_MPWRDLCTL field descriptions (continued)

Field	Description
30–24 WR_DL_ABS_ OFFSET3	Absolute write delay offset for Byte3. This field indicates the absolute delay between write DQS strobe and the write data of Byte3 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (WR_DL_ABS_OFFSET3 / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of (HW_WR_DL_LOW3 + HW_WR_DL_UP3) /2 **NOTE:** Not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 WR_DL_ABS_ OFFSET2	Absolute write delay offset for Byte2. This field indicates the absolute delay between write DQS strobe and the write data of Byte2 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (WR_DL_ABS_OFFSET2/ 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of (HW_WR_DL_LOW2 + HW_WR_DL_UP2) /2 **NOTE:** Not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 WR_DL_ABS_ OFFSET1	Absolute write delay offset for Byte1. This field indicates the absolute delay between write DQS strobe and the write data of Byte1 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (WR_DL_ABS_OFFSET1 / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of (HW_WR_DL_LOW1 + HW_WR_DL_UP1) /2 **NOTE:** Not all changes of this value will affect the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.
7 Reserved	This read-only field is reserved and always has the value 0.
WR_DL_ABS_ OFFSET0	Absolute write delay offset for Byte0. This field indicates the absolute delay between write DQS strobe and the write data of Byte3 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be (WR_DL_ABS_OFFSET0 / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay. This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of (HW_WR_DL_LOW0 + HW_WR_DL_UP0) /2 **NOTE:** Not all changes of this value will affect the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.

22.11.40 MMDC PHY Write delay-lines Status Register (MMDCx_MPWRDLST)

This register holds the status of the 4 write delay-line.

Address: 40AB_0000h base + 854h offset = 40AB_0854h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	WR_DL_UNIT_NUM3							0	WR_DL_UNIT_NUM2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WR_DL_UNIT_NUM1							0	WR_DL_UNIT_NUM0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPWRDLST field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 WR_DL_UNIT_NUM3	This field reflects the number of delay units that are actually used by write delay-line 3.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 WR_DL_UNIT_NUM2	This field reflects the number of delay units that are actually used by write delay-line 2.
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 WR_DL_UNIT_NUM1	This field reflects the number of delay units that are actually used by write delay-line 1.
7 Reserved	This read-only field is reserved and always has the value 0.
WR_DL_UNIT_NUM0	This field reflects the number of delay units that are actually used by write delay-line 0.

22.11.41 MMDC ZQ LPDDR2 HW Control Register (MMDCx_MPZQLP2CTL)

This register controls the idle time that takes the LPDDR2/3 device to perform ZQ calibration.

Address: 40AB_0000h base + 85Ch offset = 40AB_085Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	ZQ_LP2_HW_ZQCS							ZQ_LP2_HW_ZQCL							
W																
Reset	0	0	0	0	1	1	0	1	1	0	1	0	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								ZQ_LP2_HW_ZQINIT							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1

MMDCx_MPZQLP2CTL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 ZQ_LP2_HW_ZQCS	This register defines the period in cycles that it takes the memory device to perform a short ZQ calibration. This is the period of time that the MMDC has to wait after sending a short ZQ calibration and before sending other commands. This delay will also be used if ZQ reset is sent. 0x0-0x1A Reserved 0x1B 112 cycles (default) 0x1C 116 cycles 0x7E 508 cycles 0x7F 512 cycles
23–16 ZQ_LP2_HW_ZQCL	This register defines the period in cycles that it takes the memory device to perform a long ZQ calibration. This is the period of time that the MMDC has to wait after sending a long ZQ calibration and before sending other commands. 0x0-0x36 Reserved 0x37 112 cycles 0x38 114 cycles 0x5F 192 cycles (Default. This may need to be adjusted to achieve the correct timing depending on the DDR clock frequency.) 0xFE 510 cycles 0xFF 512 cycles
15–9 Reserved	This read-only field is reserved and always has the value 0.

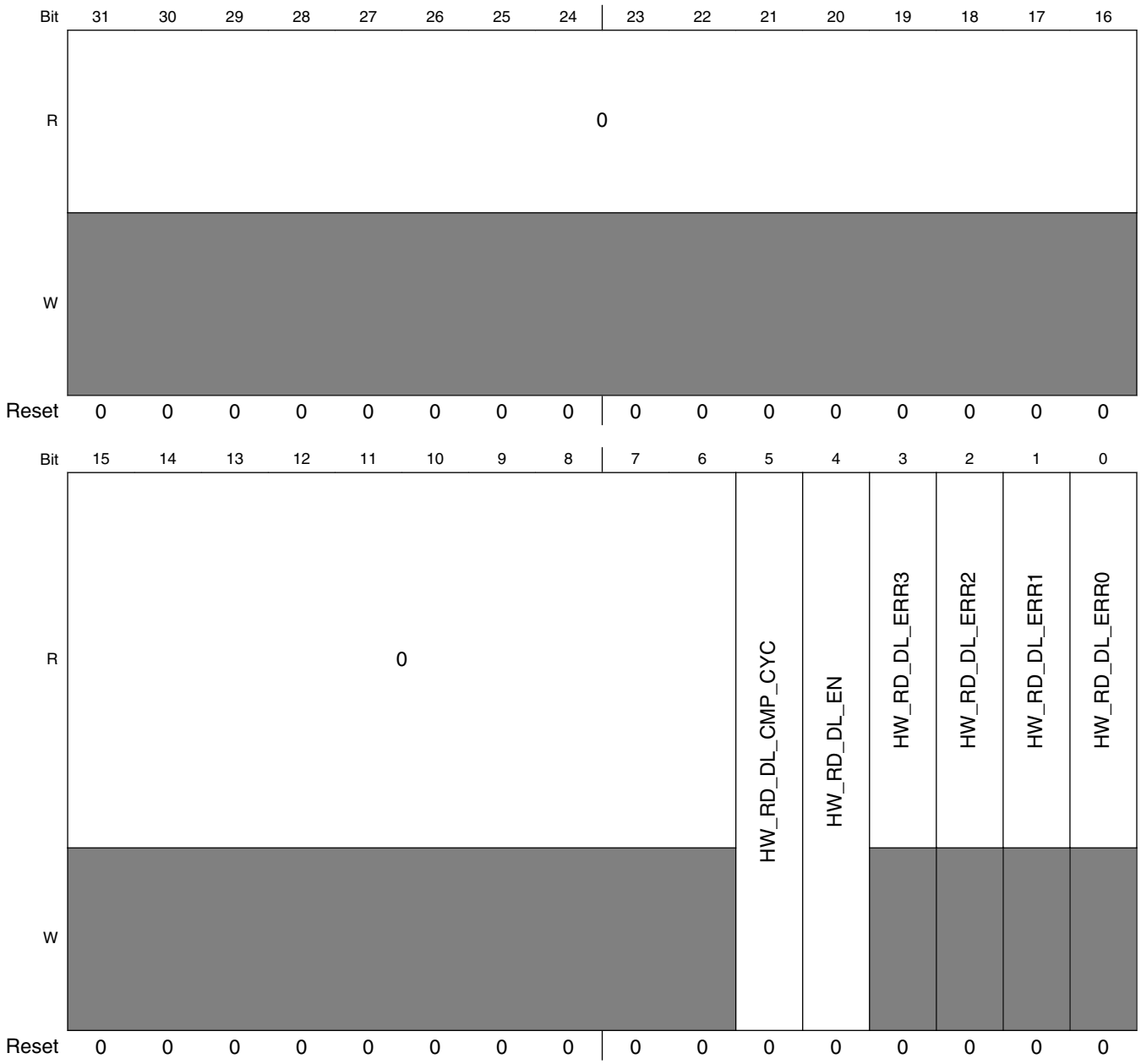
Table continues on the next page...

MMDCx_MPZQLP2CTL field descriptions (continued)

Field	Description
ZQ_LP2_HW_ZQINIT	This register defines the period in cycles that it takes the memory device to perform a Init ZQ calibration. This is the period of time that the MMDC has to wait after sending a init ZQ calibration and before sending other commands. 0x0-0x36 Reserved 0x37 112 cycles 0x38 114 cycles 0x109 532 cycles (Default. This may need to be adjusted to achieve the correct timing depending on the DDR clock frequency.) 0x1FE 1022 cycles 0x1FF 1024 cycles

22.11.42 MMDC PHY Read Delay HW Calibration Control Register (MMDCx_MPRDDLHWCTL)

Address: 40AB_0000h base + 860h offset = 40AB_0860h



MMDCx_MPRDDLHWCTL field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.

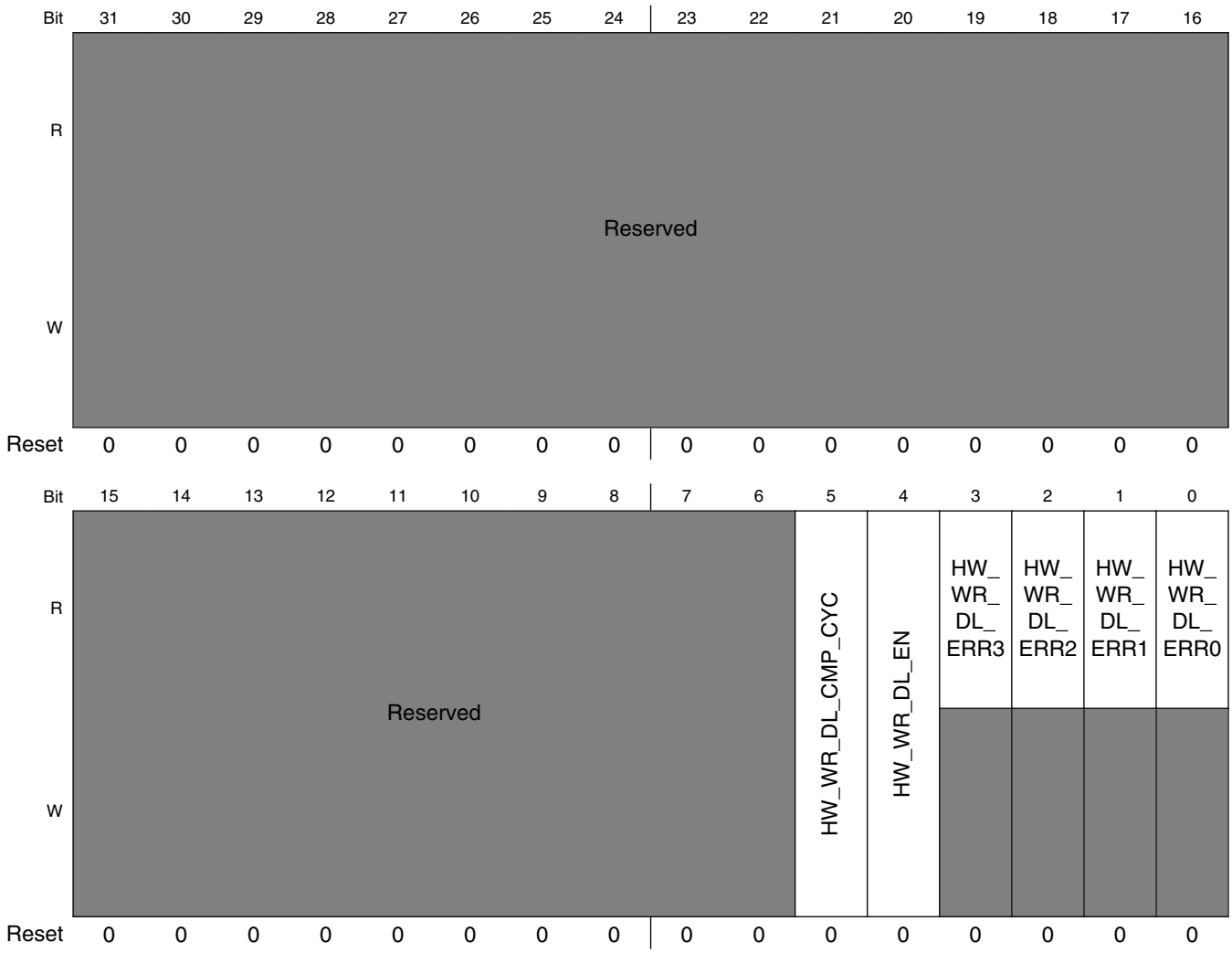
Table continues on the next page...

MMDCx_MPRDDLHWCTL field descriptions (continued)

Field	Description
5 HW_RD_DL_ CMP_CYC	Automatic (HW) read sample cycle. If this bit is asserted then the MMDC will compare the read data 32 cycles after the MMDC sent the read command enable pulse else it compares the data after 16 cycles.
4 HW_RD_DL_EN	Enable automatic (HW) read calibration. If this bit is asserted then the MMDC will perform an automatic read calibration. HW should negate this bit upon completion of the calibration. Negation of this bit also points that the read calibration results are valid **NOTE:** Before issuing the first read command MMDC counts 12 cycles.
3 HW_RD_DL_ ERR3	Automatic (HW) read calibration error of Byte3. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 3. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST1 register. This bit is valid only after HW_RD_DL_EN is de-asserted. 0 No error was found in read delay-line 3 during the automatic (HW) read calibration process of read delay-line 3. 1 An error was found in read delay-line 3 during the automatic (HW) read calibration process of read delay-line 3.
2 HW_RD_DL_ ERR2	Automatic (HW) read calibration error of Byte2. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 2. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST1 register. This bit is valid only after HW_RD_DL_EN is de-asserted. 0 No error was found in read delay-line 2 during the automatic (HW) read calibration process of read delay-line 2. 1 An error was found in read delay-line 2 during the automatic (HW) read calibration process of read delay-line 2.
1 HW_RD_DL_ ERR1	Automatic (HW) read calibration error of Byte1. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 1. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST0 register. This bit is valid only after HW_RD_DL_EN is de-asserted. 0 No error was found in read delay-line 1 during the automatic (HW) read calibration process of read delay-line 1. 1 An error was found in read delay-line 1 during the automatic (HW) read calibration process of read delay-line 1.
0 HW_RD_DL_ ERR0	Automatic (HW) read calibration error of Byte0. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 0. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST0 register. This bit is valid only after HW_RD_DL_EN is de-asserted. 0 No error was found in read delay-line 0 during the automatic (HW) read calibration process of read delay-line 0. 1 An error was found in read delay-line 0 during the automatic (HW) read calibration process of read delay-line 0.

22.11.43 MMDC PHY Write Delay HW Calibration Control Register (MMDCx_MPWRDLHWCTL)

Address: 40AB_0000h base + 864h offset = 40AB_0864h



MMDCx_MPWRDLHWCTL field descriptions

Field	Description
31–6 -	This field is reserved.
5 HW_WR_DL_CMP_CYC	Write sample cycle. If this bit is asserted then the MMDC will compare the data 32 cycles after the MMDC sent the read command enable pulse else it compares the data after 16 cycles.
4 HW_WR_DL_EN	Enable automatic (HW) write calibration. If this bit is asserted then the MMDC will perform an automatic write calibration. HW should negate this bit upon completion of the calibration. Negation of this bit also indicates that the write calibration results are valid **NOTE:** Before issuing the first read command MMDC counts 12 cycles.

Table continues on the next page...

MMDCx_MPWRDLHWCTL field descriptions (continued)

Field	Description
3 HW_WR_DL_ ERR3	Automatic (HW) write calibration error of Byte3. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 3. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST1 register. This bit is valid only after HW_WR_DL_EN is de-asserted. 0 No error was found during the automatic (HW) write calibration process of write delay-line 3. 1 An error was found during the automatic (HW) write calibration process of write delay-line 3.
2 HW_WR_DL_ ERR2	Automatic (HW) write calibration error of Byte2. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 2. T In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST1 register. This bit is valid only after HW_WR_DL_EN is de-asserted. 0 No error was found during the automatic (HW) write calibration process of write delay-line 2. 1 An error was found during the automatic (HW) write calibration process of write delay-line 2.
1 HW_WR_DL_ ERR1	Automatic (HW) write calibration error of Byte1. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 1. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST0 register. This bit is valid only after HW_WR_DL_EN is de-asserted. 0 No error was found during the automatic (HW) write calibration process of write delay-line 1. 1 An error was found during the automatic (HW) write calibration process of write delay-line 1.
0 HW_WR_DL_ ERR0	Automatic (HW) write calibration error of Byte0. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 0. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST0 register. This bit is valid only after HW_WR_DL_EN is de-asserted. 0 No error was found during the automatic (HW) write calibration process of write delay-line 0. 1 An error was found during the automatic (HW) write calibration process of write delay-line 0.

22.11.44 MMDC PHY Read Delay HW Calibration Status Register 0 (MMDCx_MPRDDLHWST0)

Address: 40AB_0000h base + 868h offset = 40AB_0868h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_RD_DL_UP1							0	HW_RD_DL_LOW1						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_RD_DL_UP0							0	HW_RD_DL_LOW0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPRDDLHWST0 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_RD_DL_ UP1	Automatic (HW) read calibration result of the upper boundary of Byte1. This field holds the automatic (HW) read calibration result of the upper boundary of Byte1
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_RD_DL_ LOW1	Automatic (HW) read calibration result of the lower boundary of Byte1. This field holds the automatic (HW) read calibration result of the lower boundary of Byte1
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_RD_DL_ UP0	Automatic (HW) read calibration result of the upper boundary of Byte0. This field holds the automatic (HW) read calibration result of the upper boundary of Byte0.
7 Reserved	This read-only field is reserved and always has the value 0.
HW_RD_DL_ LOW0	Automatic (HW) read calibration result of the lower boundary of Byte0. This field holds the automatic (HW) read calibration result of the lower boundary of Byte0.

22.11.45 MMDC PHY Read Delay HW Calibration Status Register 1 (MMDCx_MPRDDLHWST1)

Address: 40AB_0000h base + 86Ch offset = 40AB_086Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_RD_DL_UP3							0	HW_RD_DL_LOW3						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_RD_DL_UP2							0	HW_RD_DL_LOW2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPRDDLHWST1 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_RD_DL_ UP3	Automatic (HW) read calibration result of the upper boundary of Byte3. This field holds the automatic (HW) read calibration result of the upper boundary of Byte3

Table continues on the next page...

MMDCx_MPRDDLHWST1 field descriptions (continued)

Field	Description
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_RD_DL_ LOW3	Automatic (HW) read calibration result of the lower boundary of Byte3. This field holds the automatic (HW) read calibration result of the lower boundary of Byte3
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_RD_DL_ UP2	Automatic (HW) read calibration result of the upper boundary of Byte2. This field holds the automatic (HW) read calibration result of the upper boundary of Byte2.
7 Reserved	This read-only field is reserved and always has the value 0.
HW_RD_DL_ LOW2	Automatic (HW) read calibration result of the lower boundary of Byte2. This field holds the automatic (HW) read calibration result of the lower boundary of Byte2.

22.11.46 MMDC PHY Write Delay HW Calibration Status Register 0 (MMDCx_MPWRDLHWST0)

Address: 40AB_0000h base + 870h offset = 40AB_0870h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_WR_DL_UP1							0	HW_WR_DL_LOW1						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_WR_DL_UP0							0	HW_WR_DL_LOW0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPWRDLHWST0 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_WR_DL_ UP1	Automatic (HW) write automatic (HW) write calibration result of the upper boundary of Byte1. This field holds the automatic (HW) write calibration result of the upper boundary of Byte1.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_WR_DL_ LOW1	Automatic (HW) write calibration result of the lower boundary of Byte1. This field holds the automatic (HW) write calibration result of the lower boundary of Byte1.

Table continues on the next page...

MMDCx_MPWRDLHWST0 field descriptions (continued)

Field	Description
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_WR_DL_ UP0	Automatic (HW) write calibration result of the upper boundary of Byte0. This field holds the automatic (HW) write calibration result of the upper boundary of Byte0.
7 Reserved	This read-only field is reserved and always has the value 0.
HW_WR_DL_ LOW0	Automatic (HW) write calibration result of the lower boundary of Byte0. This field holds the automatic (HW) write calibration result of the lower boundary of Byte0.

22.11.47 MMDC PHY Write Delay HW Calibration Status Register 1 (MMDCx_MPWRDLHWST1)

Address: 40AB_0000h base + 874h offset = 40AB_0874h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_WR_DL_UP3							0	HW_WR_DL_LOW3						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_WR_DL_UP2							0	HW_WR_DL_LOW2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPWRDLHWST1 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_WR_DL_ UP3	Automatic (HW) write calibration result of the upper boundary of Byte3. This field holds the automatic (HW) write calibration result of the upper boundary of Byte3.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_WR_DL_ LOW3	Automatic (HW) write calibration result of the lower boundary of Byte3. This field holds the automatic (HW) write calibration result of the lower boundary of Byte3.
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_WR_DL_ UP2	Automatic (HW) write calibration result of the upper boundary of Byte2. This field holds the automatic (HW) write calibration result of the upper boundary of Byte2.

Table continues on the next page...

MMDCx_MPWRDLHWST1 field descriptions (continued)

Field	Description
7 Reserved	This read-only field is reserved and always has the value 0.
HW_WR_DL_ LOW2	Automatic (HW) write calibration result of the lower boundary of Byte2. This field holds the automatic (HW) write calibration result of the lower boundary of Byte2.

22.11.48 MMDC PHY Pre-defined Compare Register 1 (MMDCx_MPPDCMPR1)

This register holds the MMDC pre-defined compare value that will be used during automatic read, read DQS gating and write calibration process. The compare value can be the MPR value (as defined in the JEDEC) or can be programmed by the PDV1 and PDV2 fields..

Address: 40AB_0000h base + 88Ch offset = 40AB_088Ch

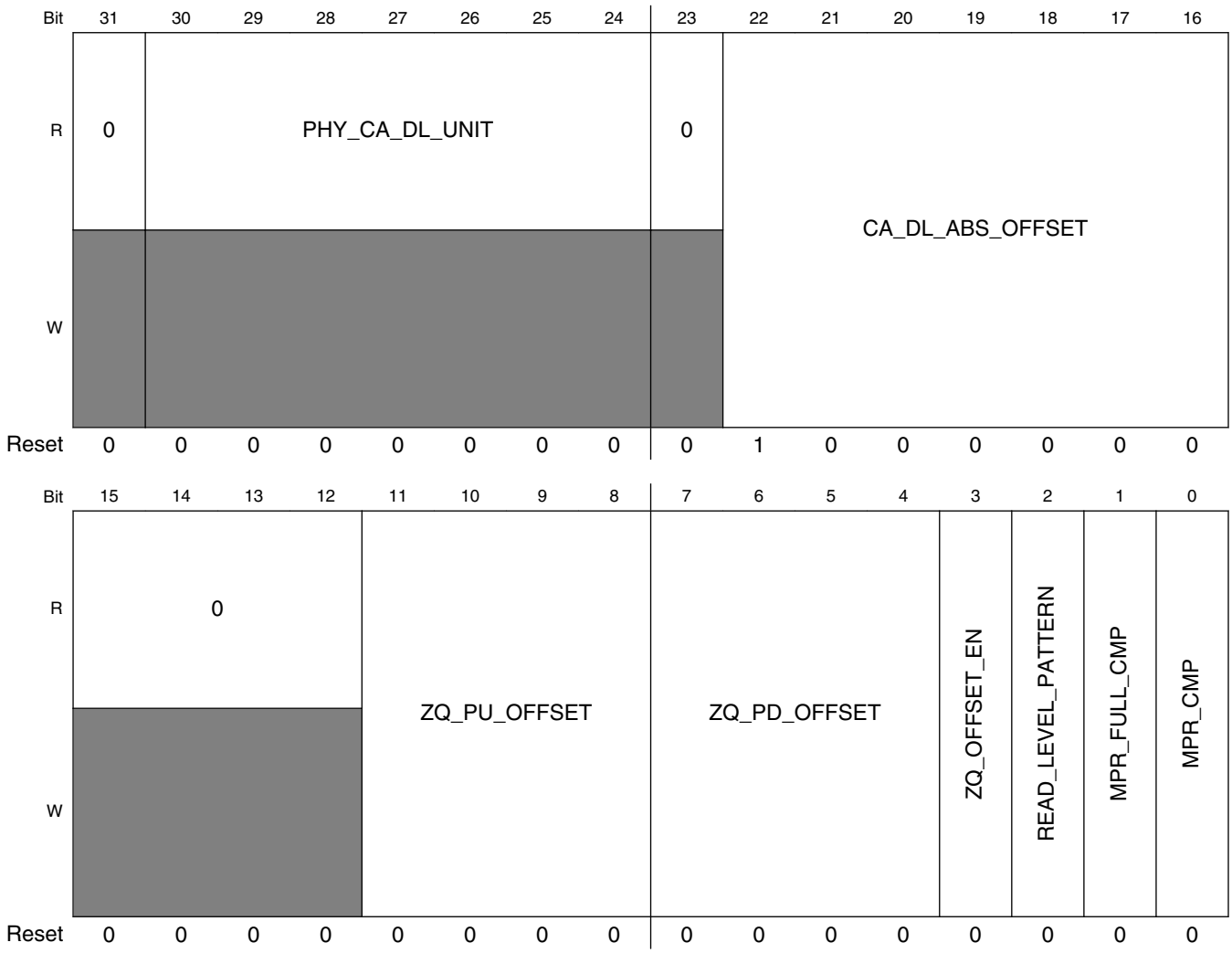
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDV2																PDV1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MMDCx_MPPDCMPR1 field descriptions

Field	Description
31–16 PDV2	MMDC Pre defined compare value2. This field holds the 2 MSB of the data that will be driven to the DDR device during automatic read, read DQS gating and write calibrations in case DQ calibration (LPDDR2/ LPDDR3) mode are disabled (MPR_CMP is disabled). Upon read access during the calibration the MMDC will compare the read data with the data that is stored in this field. **NOTE:** Before issue the read access the MMDC will invert the value of this field and drive it to the associate entry in the read comparison FIFO. For further information, see Calibration with pre-defined value and Hardware (automatic) Calibration with DQ Calibration
PDV1	MMDC Pre defined compare value2. This field holds the 2 LSB of the data that will be driven to the DDR device during automatic read, read DQS gating and write calibrations in case DQ calibration (LPDDR2/ LPDDR3) mode are disabled (MPR_CMP is disabled). Upon read access during the calibration the MMDC will compare the read data with the data that is stored in this field. **NOTE:** Before issuing the read access, the MMDC will invert the value of this field and drive it to the associated entry in the read comparison FIFO.

22.11.49 MMDC PHY Pre-defined Compare and CA delay-line Configuration Register (MMDCx_MPPDCMPR2)

Address: 40AB_0000h base + 890h offset = 40AB_0890h



MMDCx_MPPDCMPR2 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 PHY_CA_DL_UNIT	This field reflects the number of delay units that are actually used by CA(Command/Address of LPDDR2) delay-line
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 CA_DL_ABS_OFFSET	Absolute CA (Command/Address of LPDDRR2) offset. This field indicates the absolute delay between CA (Command/Address) bus and the DDR clock (CK) with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be

Table continues on the next page...

MMDCx_MPPDCMPR2 field descriptions (continued)

Field	Description
	(CA_DL_ABS_OFFSET / 256) * MMDC AXI clock (fast clock). So for the default value of 64 we get a quarter cycle delay.
15–12 Reserved	This read-only field is reserved and always has the value 0.
11–8 ZQ_PU_OFFSET	Programmable offset from -7 to 7 added to the MMDC_MPZQHWCTRL[ZQ_HW_PU_RES] field when ZQ_OFFSET_EN is enabled. Bit[11] determines direction: 'b0 = Offset is added to ZQ_HW_PU_RES field 'b1 = Offset is subtracted from ZQ_HW_PU_RES field Bits[10:8] = Amount of change to apply. 0000 **0** — Offset is added to ZQ_HW_PU_RES field 0001 **1** — Offset is added to ZQ_HW_PU_RES field 0010 **2** — Offset is added to ZQ_HW_PU_RES field 0011 **3** — Offset is added to ZQ_HW_PU_RES field 0100 **4** — Offset is added to ZQ_HW_PU_RES field 0101 **5** — Offset is added to ZQ_HW_PU_RES field 0110 **6** — Offset is added to ZQ_HW_PU_RES field 0111 **7** — Offset is added to ZQ_HW_PU_RES field 1000 **0** — Offset is subtracted from ZQ_HW_PU_RES field 1001 **1** — Offset is subtracted from ZQ_HW_PU_RES field 1010 **2** — Offset is subtracted from ZQ_HW_PU_RES field 1011 **3** — Offset is subtracted from ZQ_HW_PU_RES field 1100 **4** — Offset is subtracted from ZQ_HW_PU_RES field 1101 **5** — Offset is subtracted from ZQ_HW_PU_RES field 1110 **6** — Offset is subtracted from ZQ_HW_PU_RES field 1111 **7** — Offset is subtracted from ZQ_HW_PU_RES field
7–4 ZQ_PD_OFFSET	Programmable offset from -7 to 7 added to the MMDC_MPZQHWCTRL[ZQ_HW_PD_RES] field when ZQ_OFFSET_EN is enabled. Bit[7] determines direction: 'b0 = Offset is added to ZQ_HW_PD_RES field 'b1 = Offset is subtracted from ZQ_HW_PD_RES field Bits[6:4] = Amount of change to apply. 0000 **0** — Offset is added to ZQ_HW_PD_RES field 0001 **1** — Offset is added to ZQ_HW_PD_RES field 0010 **2** — Offset is added to ZQ_HW_PD_RES field 0011 **3** — Offset is added to ZQ_HW_PD_RES field 0100 **4** — Offset is added to ZQ_HW_PD_RES field 0101 **5** — Offset is added to ZQ_HW_PD_RES field 0110 **6** — Offset is added to ZQ_HW_PD_RES field 0111 **7** — Offset is added to ZQ_HW_PD_RES field 1000 **0** — Offset is subtracted from ZQ_HW_PD_RES field 1001 **1** — Offset is subtracted from ZQ_HW_PD_RES field 1010 **2** — Offset is subtracted from ZQ_HW_PD_RES field 1011 **3** — Offset is subtracted from ZQ_HW_PD_RES field

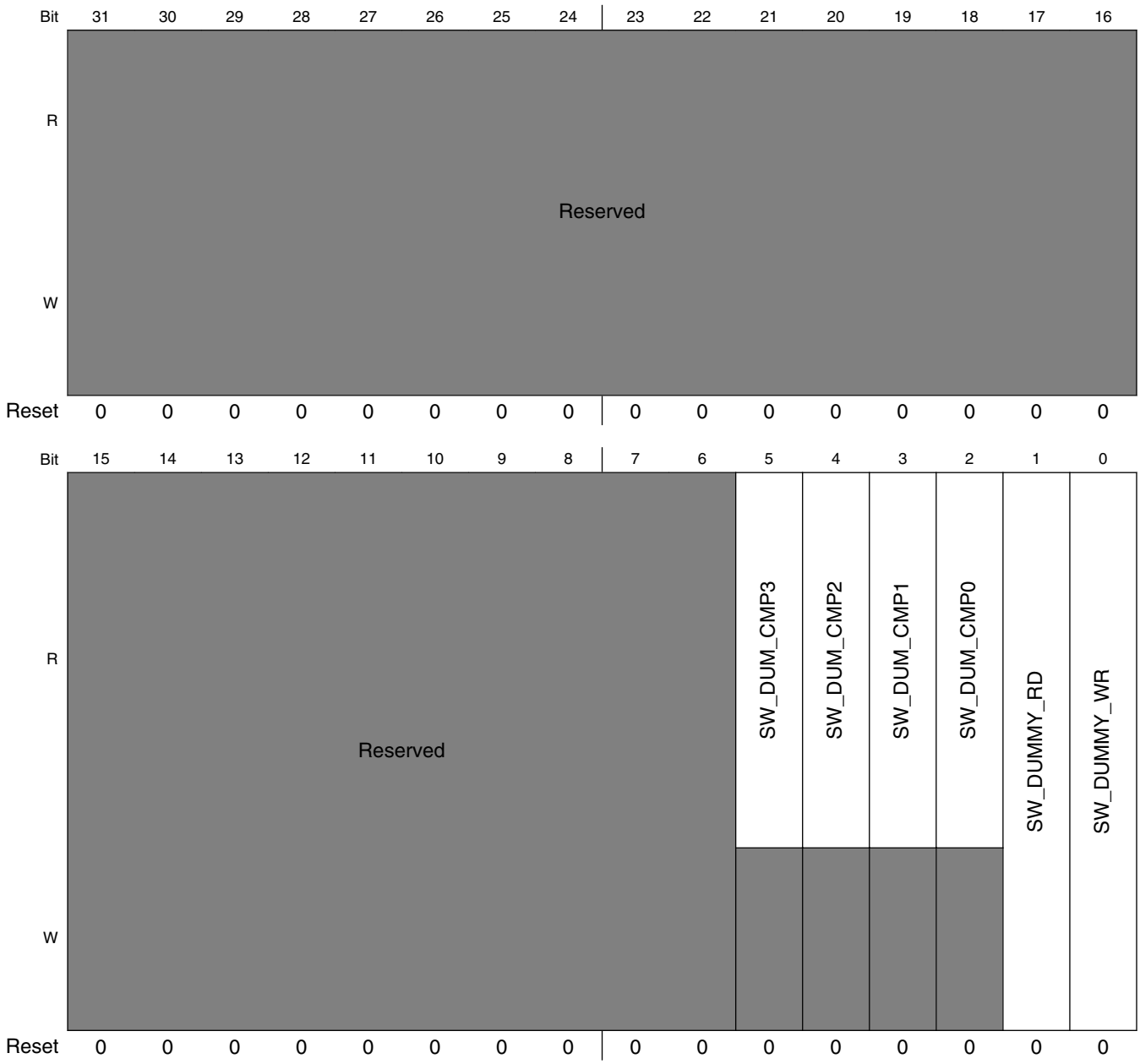
Table continues on the next page...

MMDCx_MPPDCMPR2 field descriptions (continued)

Field	Description
	1100 **4** — Offset is subtracted from ZQ_HW_PD_RES field 1101 **5** — Offset is subtracted from ZQ_HW_PD_RES field 1110 **6** — Offset is subtracted from ZQ_HW_PD_RES field 1111 **7** — Offset is subtracted from ZQ_HW_PD_RES field
3 ZQ_OFFSET_EN	ZQ Offset Enable 0 Hardware ZQ offset disabled 1 Hardware ZQ offset enabled
2 READ_LEVEL_ PATTERN	DQ calibration (LPDDR2/LPDDR3) read compare pattern. In case DQ calibration (LPDDR2/LPDDR3) modes are used during the calibration process (MPR_CMP is asserted) then this field indicates the read pattern for the comparison. 0 Compare with read pattern 1010 1 Compare with read pattern 0011 (Used only in LPDDR2/LPDDR3 mode)
1 MPR_FULL_ CMP	DQ calibration (LPDDR2/LPDDR3) full compare enable. In case DQ calibration (LPDDR2/LPDDR3) modes are used during the calibration process (MPR_CMP is asserted) then this field indicates whether the MMDC will compare all the bits of the data that is read from the DDR device to the MPR pre-defined pattern. When this bit is de-asserted only LSB of each byte is compared.
0 MPR_CMP	DQ calibration (LPDDR2/LPDDR3) compare enable. This bit indicates whether the MMDC will compare the read data during automatic read and read DQS calibration processes to the pre-defined patterns that are driven by the DDR device (READ_LEVEL_PATTERN as defined by JEDEC) or general pre-defined value that are stored in PDV1 and PDV2. When this bit is disabled data is compared to the data of the pre defined compare value field For further information see Read Calibration .

22.11.50 MMDC PHY SW Dummy Access Register (MMDCx_MPSWDAR0)

Address: 40AB_0000h base + 894h offset = 40AB_0894h



MMDCx_MPSWDAR0 field descriptions

Field	Description
31–6 -	This field is reserved.

Table continues on the next page...

MMDCx_MPSWDAR0 field descriptions (continued)

Field	Description
5 SW_DUM_CMP3	SW dummy read byte3 compare results. This bit indicates the result of the read data comparison of Byte3 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted. 0 Dummy read fail 1 Dummy read pass
4 SW_DUM_CMP2	SW dummy read byte2 compare results. This bit indicates the result of the read data comparison of Byte2 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted. 0 Dummy read fail 1 Dummy read pass
3 SW_DUM_CMP1	SW dummy read byte1 compare results. This bit indicates the result of the read data comparison of Byte1 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted. 0 Dummy read fail 1 Dummy read pass
2 SW_DUM_CMP0	SW dummy read byte0 compare results. This bit indicates the result of the read data comparison of Byte0 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted. 0 Dummy read fail 1 Dummy read pass
1 SW_DUMMY_ RD	SW dummy read. When this bit is asserted the MMDC will generate internally read access without intervention of the system toward bank 0, row 0, column 0. If MPR_CMP = 1then the read data will be compared to MPPDCMPR2[READ_LEVEL_PATTERN] . If MPR_CMP =0 then the read data will be compared to MPPDCMPR1[PDV1], MPPDCMPR1[PDV2]. Upon completion of the access this bit is de-asserted automatically and the read data and comparison results are valid at MPSWDAR0[SW_DUM_CMP#] and MPSWDRDR0-MPSWDRDR7 respectively.
0 SW_DUMMY_ WR	SW dummy write. When this bit is asserted the MMDC will generate internally write access without intervention of the system toward bank 0, row 0, column 0, while the data is driven from MPPDCMPR1[PDV1] and MPPDCMPR1[PDV2]. The bit is de-asserted automatically upon completion of the access.

22.11.51 MMDC PHY SW Dummy Read Data Register 0 (MMDCx_MPSWDRDR0)

Address: 40AB_0000h base + 898h offset = 40AB_0898h

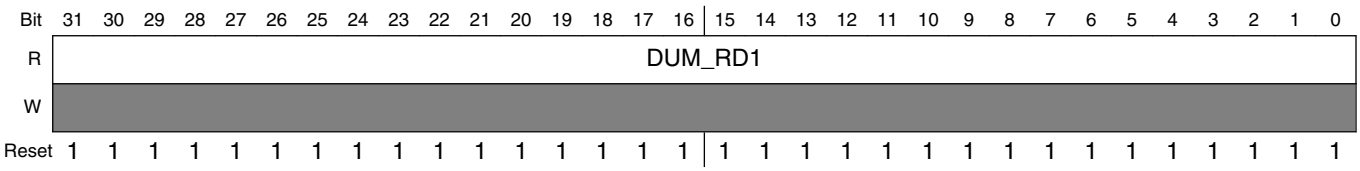
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD0																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

MMDCx_MPSWDRDR0 field descriptions

Field	Description
DUM_RD0	Dummy read data0. This field holds the first data that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted

22.11.52 MMDC PHY SW Dummy Read Data Register 1 (MMDCx_MPSWDRDR1)

Address: 40AB_0000h base + 89Ch offset = 40AB_089Ch

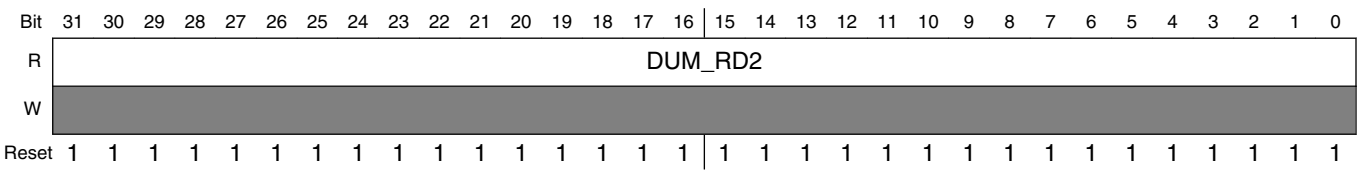


MMDCx_MPSWDRDR1 field descriptions

Field	Description
DUM_RD1	Dummy read data1. This field holds the second data that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted

22.11.53 MMDC PHY SW Dummy Read Data Register 2 (MMDCx_MPSWDRDR2)

Address: 40AB_0000h base + 8A0h offset = 40AB_08A0h

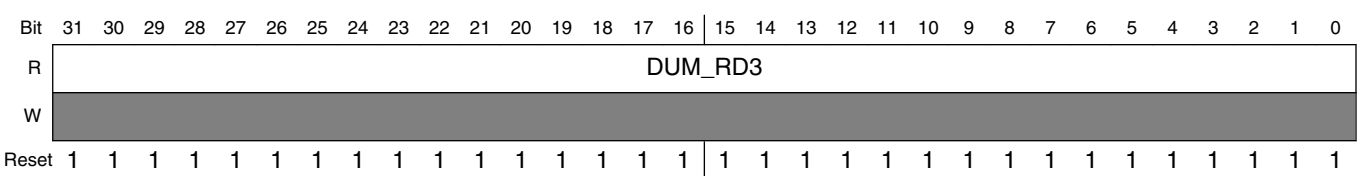


MMDCx_MPSWDRDR2 field descriptions

Field	Description
DUM_RD2	Dummy read data2. This field holds the third data that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

22.11.54 MMDC PHY SW Dummy Read Data Register 3 (MMDCx_MPSWDRDR3)

Address: 40AB_0000h base + 8A4h offset = 40AB_08A4h

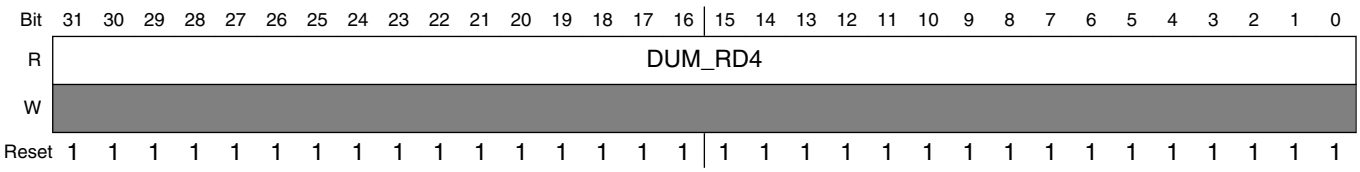


MMDCx_MPSWDRDR3 field descriptions

Field	Description
DUM_RD3	Dummy read data3. This field holds the forth data that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

22.11.55 MMDC PHY SW Dummy Read Data Register 4 (MMDCx_MPSWDRDR4)

Address: 40AB_0000h base + 8A8h offset = 40AB_08A8h

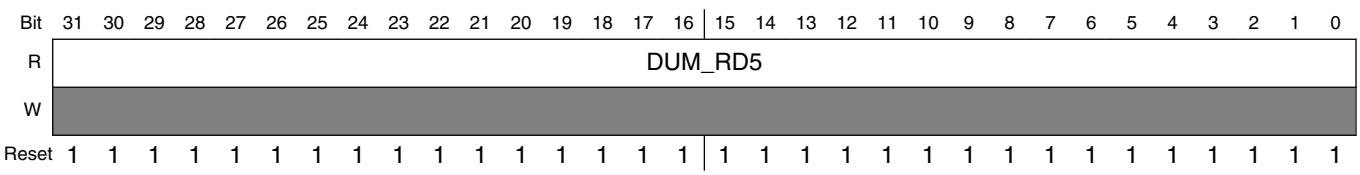


MMDCx_MPSWDRDR4 field descriptions

Field	Description
DUM_RD4	Dummy read data4. This field holds the fifth data (only in case of burst length 8 (BL =1)) that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

22.11.56 MMDC PHY SW Dummy Read Data Register 5 (MMDCx_MPSWDRDR5)

Address: 40AB_0000h base + 8ACh offset = 40AB_08ACh

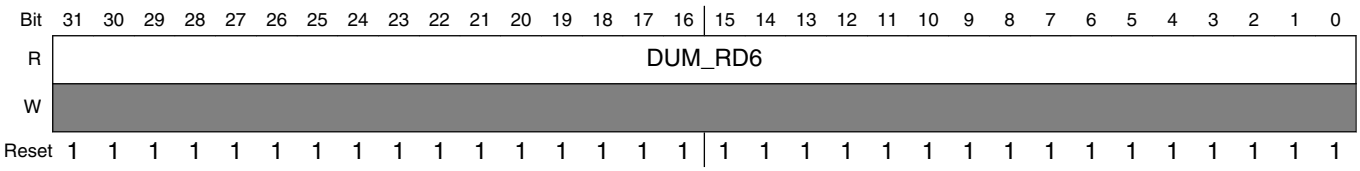


MMDCx_MPSWDRDR5 field descriptions

Field	Description
DUM_RD5	Dummy read data5. This field holds the sixth data (only in case of burst length 8 (BL =1)) that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

22.11.57 MMDC PHY SW Dummy Read Data Register 6 (MMDCx_MPSWDRDR6)

Address: 40AB_0000h base + 8B0h offset = 40AB_08B0h

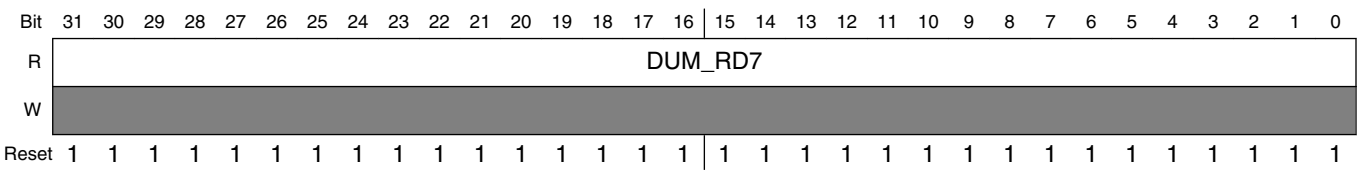


MMDCx_MPSWDRDR6 field descriptions

Field	Description
DUM_RD6	Dummy read data6. This field holds the seventh data (only in case of burst length 8 (BL =1)) that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

22.11.58 MMDC PHY SW Dummy Read Data Register 7 (MMDCx_MPSWDRDR7)

Address: 40AB_0000h base + 8B4h offset = 40AB_08B4h

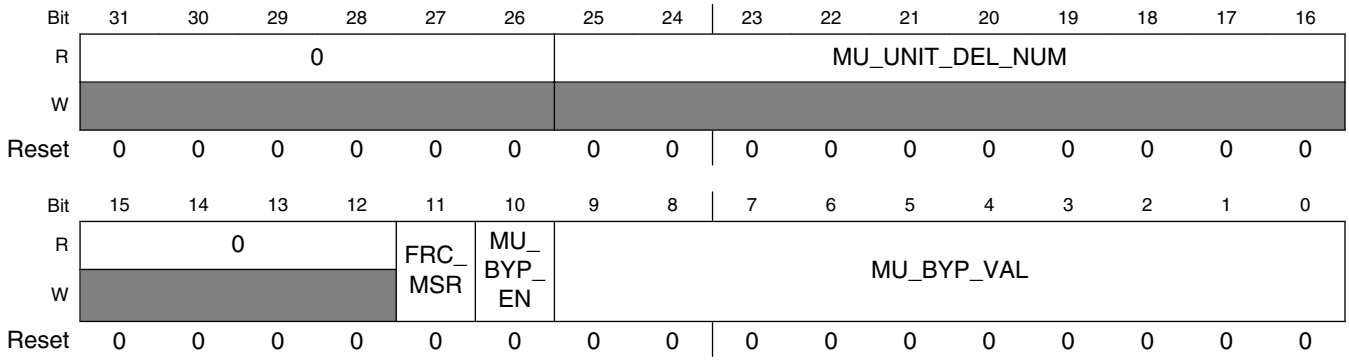


MMDCx_MPSWDRDR7 field descriptions

Field	Description
DUM_RD7	Dummy read data7. This field holds the eight data (only in case of burst length 8 (BL =1)) that is read from the DDR during SW dummy read access (i.e. when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

22.11.59 MMDC PHY Measure Unit Register (MMDCx_MPMUR0)

Address: 40AB_0000h base + 8B8h offset = 40AB_08B8h



MMDCx_MPMUR0 field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value 0.
25–16 MU_UNIT_DEL_NUM	Number of delay units measured per cycle. This field is used in debug mode and holds the number of delay units that were measured by the measure unit per DDR clock cycle. The delay-lines that are used in every calibration process use that number for generating the desired delay.
15–12 Reserved	This read-only field is reserved and always has the value 0.
11 FRC_MSR	Force measurement on delay-lines. When this bit is asserted then a measurement process will be performed, where at the completion of the process the delay-lines will issue the desired delay. Upon completion of the measurement process the measure unit and the delay-lines will return to functional mode. This bit is self cleared. **NOTE:** This bit should be used only during manual (SW) calibration and not while the DDR is functional (being accessed). After initial calibration is done the hardware performs periodic measurements to track any operating conditions changes. Hence, force measurements (FRC_MSR) should not be used. See Calibration Process for more information. **NOTE:** User should make sure that there is no active accesses to/from DDR before asserting this bit. 0 No measurement is performed 1 Perform measurement process
10 MU_BYP_EN	Measure unit bypass enable. This field is used in debug mode and when it is asserted then the delay-lines will use the number of delay units that are indicated at MU_BYP_VAL, otherwise the delay-lines will use the number of delay units that was measured by the measurement unit and are indicated at MU_UNIT_DEL_NUM 0 The delay-lines use delay units as indicated at MU_UNIT_DEL_NUM. 1 The delay-lines use delay units as indicated at MU_BYPASS_VAL.
MU_BYP_VAL	Number of delay units for measurement bypass. This field is used in debug mode and holds the number of delay units that will be used by the delay-lines when MU_BYP_EN is asserted.

22.11.60 MMDC Duty Cycle Control Register (MMDCx_MPDCCR)

This register is used to control the duty cycle of the DQS and the primary clock (CK0). Programming of this register is permitted by entering the DDR device into self-refresh mode through LPMD/DVFS mechanism.

MMDC1_MPDCCR only affects SDCLK0.

NOTE

If the duty cycle is modified after DDR initialization, the DDR will have to be placed in self-refresh mode.

NOTE

The duty cycle may be changed during initial DDR initialization without having to be placed in self-refresh mode.

Address: 40AB_0000h base + 8C0h offset = 40AB_08C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	RD_DQS3_FT_DCC			RD_DQS2_FT_DCC				RD_DQS1_FT_DCC			RD_DQS0_FT_DCC		CK_FT1_DCC		
Reset	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	CK_FT0_DCC			WR_DQS3_FT_DCC				WR_DQS2_FT_DCC			WR_DQS1_FT_DCC			WR_DQS0_FT_DCC	
Reset	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0

MMDCx_MPDCCR field descriptions

Field	Description
31 -	This field is reserved. reserved

Table continues on the next page...

MMDCx_MPDCCR field descriptions (continued)

Field	Description
30–28 RD_DQS3_FT_DCC	Read DQS duty cycle fine tuning control of Byte3. This field controls the duty cycle of read DQS of Byte3 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high
27–25 RD_DQS2_FT_DCC	Read DQS duty cycle fine tuning control of Byte2. This field controls the duty cycle of read DQS of Byte2 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high
24–22 RD_DQS1_FT_DCC	Read DQS duty cycle fine tuning control of Byte1. This field controls the duty cycle of read DQS of Byte1 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high
21–19 RD_DQS0_FT_DCC	Read DQS duty cycle fine tuning control of Byte0. This field controls the duty cycle of read DQS of Byte0 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high
18–16 CK_FT1_DCC	Secondary duty cycle fine tuning control of DDR clock. This field controls the duty cycle of the DDR clock and is cascaded to CK_FT0_DCC **NOTE:** All the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
15 -	This field is reserved. Reserved
14–12 CK_FT0_DCC	Primary duty cycle fine tuning control of DDR clock. This field controls the duty cycle of the DDR clock **NOTE:** All the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
11–9 WR_DQS3_FT_DCC	Write DQS duty cycle fine tuning control of Byte0. This field controls the duty cycle of write DQS of Byte0 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high

Table continues on the next page...

MMDCx_MPDCCR field descriptions (continued)

Field	Description
8–6 WR_DQS2_FT_DCC	Write DQS duty cycle fine tuning control of Byte1. This field controls the duty cycle of write DQS of Byte1 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high
5–3 WR_DQS1_FT_DCC	Write DQS duty cycle fine tuning control of Byte1. This field controls the duty cycle of write DQS of Byte1 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high
WR_DQS0_FT_DCC	Write DQS duty cycle fine tuning control of Byte0. This field controls the duty cycle of write DQS of Byte0 **NOTE:** All the other options are not allowed 001 51.5% low 48.5% high 010 50% duty cycle (default) 100 48.5% low 51.5% high

Chapter 23
Ultra Secured Digital Host Controller (uSDHC)

23.1 Chip-specific uSDHC information

Table 23-1. Reference links to related information

Topic	Related module	Reference
Full description	uSDHC	uSDHC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

23.1.1 ultra Secured Digital Host Controller (uSDHC)

The ultra Secured Digital Host Controller (uSDHC) provides the interface between the host system and SD, SDIO or eMMC cards. The uSDHC acts as a bridge, passing host bus transactions to the cards by sending commands and performing data accesses to/from the cards or devices. It handles SD, SDIO and eMMC protocol at the transmission level. See the i.MX 7ULP Data Sheet for bus clock frequency as per the bus mode being used on 1-bit, 4-bit, and 8-bit data bus width.

Table 23-2. uSDHC configuration

Parameter	Description
Name	ultra Secured Digital Host Controller (uSDHC)
Instances	2
Configurable features	NA
Interface speed	See the i.MX 7ULP Data Sheet for maximum operating frequency of different modes.
External I/O pins	• uSDHC0: CLK, CMD, D0-D7, DQS, RESET_b • uSDHC1: CD, CLK, CMD, D0-D7, RESET_b

Table continues on the next page...

Table 23-2. uSDHC configuration (continued)

Parameter	Description
	See the attached IOMUXC spreadsheet for pin details on this device.
Interrupts	1

23.2 Introduction

uSDHC provides the interface between the host system and the SD/SDIO/MMC cards, as depicted in Figure 23-1.

uSDHC acts as a bridge, passing host bus transactions to the SD/SDIO/MMC cards by sending commands and performing data accesses to/from the cards.

It handles the SD/SDIO/MMC protocols at the transmission level.

The following are brief descriptions of the cards supported by uSDHC:

The Multi Media Card (MMC) is a universal low-cost data storage and communication media designed to cover a wide array of applications including mobile video and gaming. Previous MMC cards were based on a 7-pin serial bus with a single data pin, while the new high speed MMC communication is based on an advanced 11-pin serial bus designed to operate in the low-voltage range.

The Secure Digital Card (SD) is an evolution of the old MMC technology. It is specifically designed to meet the security, capacity, performance, and environmental requirements inherent in newly emerging audio and video consumer electronic devices. The physical form factor, pin assignment, and data transfer protocol are forward compatible with the old MMC (with some additions).

Under the SD protocol, it can be categorized into memory card, I/O card, and combo card, which have both memory and I/O functions. The memory card invokes a copyright protection mechanism that complies with the security of the SDMI standard. The I/O card, which is also known as SDIO card, provides high-speed data I/O with low-power consumption for mobile electronic devices. For the sake of simplicity, the next figure does not show cards with reduced size or mini cards.

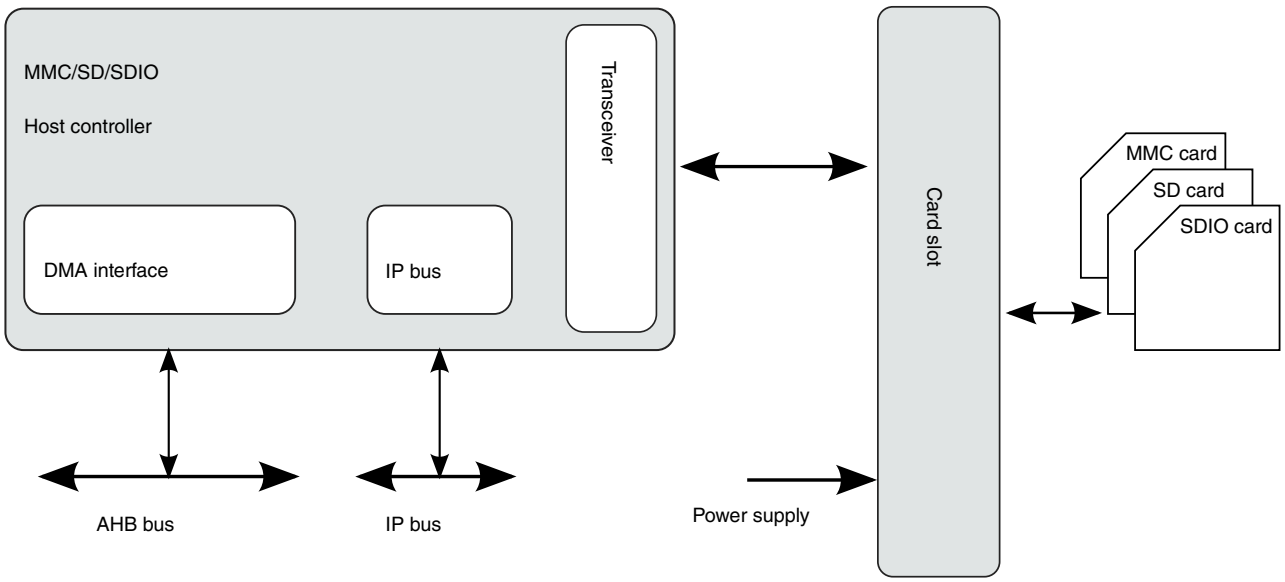


Figure 23-1. System connection of uSDHC

The following figure illustrates the block diagram of uSDHC.

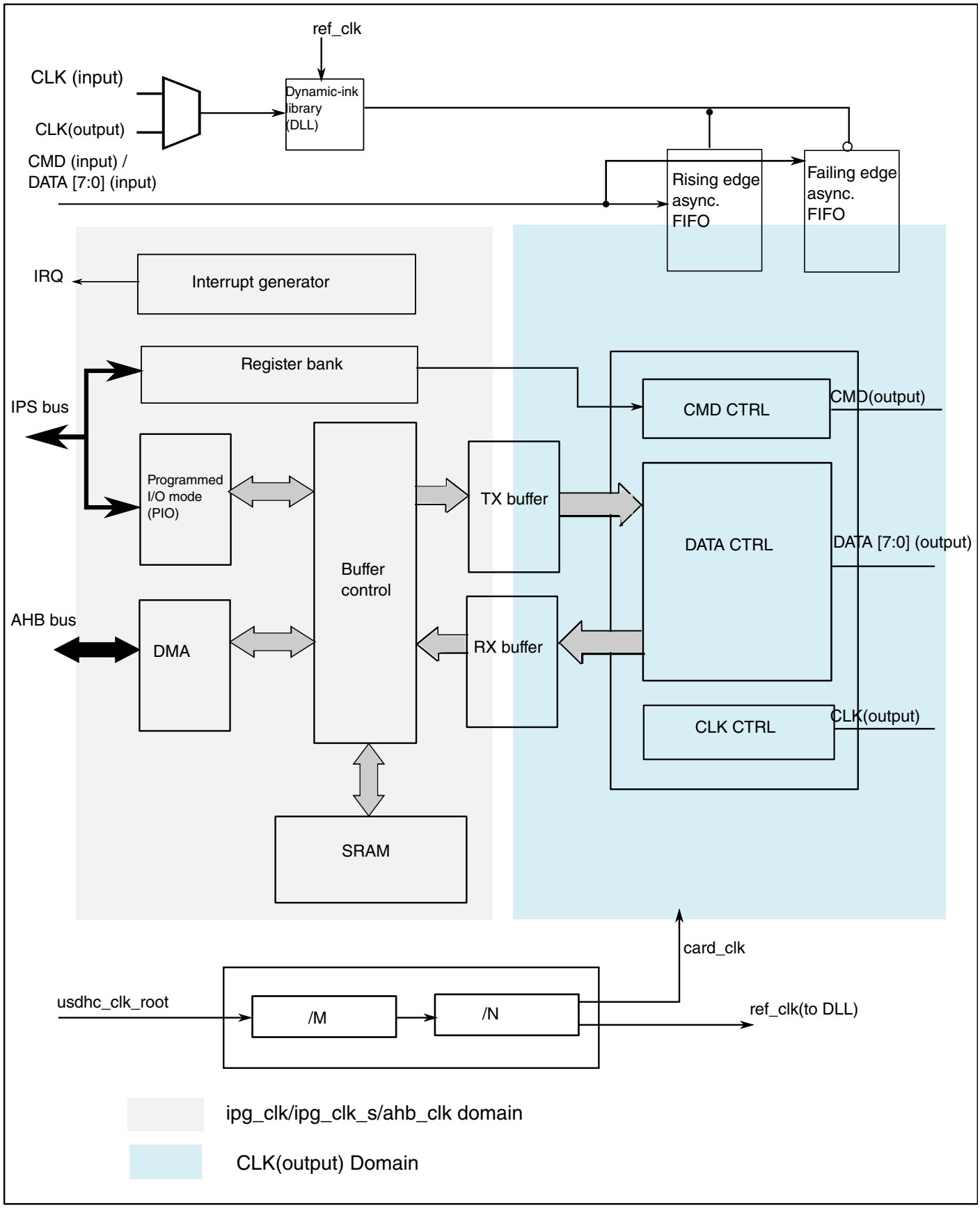


Figure 23-2. uSDHC block diagram

23.2.1 Features

The features of the uSDHC module include the following:

- Conforms to the SD Host Controller Standard Specification version 2.0/3.0
- Compatible with the MMC System Specification version 4.2/4.3/4.4/4.41/4.5/5.0
- Compatible with the SD Memory Card Specification version 3.0 and supports the Extended Capacity SD Memory Card
- Compatible with the SDIO Card Specification version 2.0/3.0
- Designed to work with SD Memory, miniSD Memory, SDIO, miniSDIO, SD Combo, MMC, MMC plus, and MMC RS cards
- Card bus clock frequency up to 208 MHz
- Supports 1-bit/4-bit SD and SDIO modes, and 1-bit/4-bit/8-bit MMC modes
  - Up to 832 Mbps of data transfer for SDIO cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDIO card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 832 Mbps of data transfer for SDXC cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDXC card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 1600 Mbps of data transfer for MMC cards using eight parallel data lines in the Single Data Rate (SDR) mode
  - Up to 3200 Mbps of data transfer for MMC cards using eight parallel data lines in the Dual Data Rate (DDR) mode
- Supports a single block/multi-block read and write
- Supports block sizes of 1 ~ 4096 bytes
- Supports the write protection switch for write operations
- Supports both synchronous and asynchronous abort
- Supports pause during the data transfer at block gap
- Supports SDIO Read Wait and Suspend Resume operations
- Supports Auto CMD12 for multi-block transfer
- Host can initiate non-data transfer command while data transfer is in progress
- Allows cards to interrupt the host in 1-bit and 4-bit SDIO modes; also supports interrupt period
- Embodies a fully configurable 256x32-bit FIFO for read/write data
- Supports internal DMA capabilities
- Support voltage selection by configuring vendor-specific register bit
- Supports Advanced DMA to perform linked memory access

NOTE

This block can support all the above listed speed mode and maximum data throughput. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

23.2.2 Modes and operations

23.2.2.1 Data transfer modes

The uSDHC module can select the following modes for data transfer:

- SD 1-bit
- SD 4-bit
- MMC 1-bit
- MMC 4-bit
- MMC 8-bit
- Identification mode (up to 400 kHz)
- MMC full-speed mode (up to 26 MHz)
- MMC high-speed mode (up to 52 MHz)
- MMC HS200 mode(up to 200 MHz)
- MMC HS400 mode (200 MHz both edges)
- MMC DDR mode (52 MHz both edges)
- SD/SDIO full-speed mode (up to 25 MHz)
- SD/SDIO high-speed mode (up to 50 MHz)
- SD/SDIO UHS-I mode(up to 208 MHz in SDR mode, up to 50 MHz in the DDR mode)

NOTE

This block can support all the above listed speed mode and maximum clock frequency. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

23.3 External signals

The following table describes the external signals of uSDHC:

Table 23-3. uSDHC external signals

Signal	Description	Direction
CLK	Clock for MMC/SD/SDIO card	O
CMD	CMD line connect to card	I/O
DATA7	DAT7 line in the 8-bit mode — Not used in other modes	I/O
DATA6	DAT6 line in the 8-bit mode — Not used in other modes	I/O
DATA5	DAT5 line in the 8-bit mode — Not used in other modes	I/O
DATA4	DAT4 line in the 8-bit mode — Not used in other modes	I/O
DATA3	DAT3 line in the 4/8-bit mode or configured as card detection pin. The bit may be configured as card detection pin in the 1-bit mode.	I/O
DATA2	DAT2 line or Read Wait in the 4-bit mode Read Wait in 1-bit mode	I/O
DATA1	DAT1 line in the 4/8-bit mode Also, used to detect interrupt in 1/4-bit mode	I/O
DATA0	DAT0 line in all the modes Also, used to detect busy state	I/O
RESET_B	Card hardware reset signal, active low	O
VSELECT	IO power voltage selection signal	O
STROBE	Input clock for eMMC HS400 mode	I

23.3.1 Signals overview

uSDHC has 15 associated I/O signals.

- The CLK is an internally generated clock used to drive the MMC, SD, and SDIO cards.
- The CMD I/O is used to send commands and receive responses to and from the card. Eight data lines (DAT7~DAT0) are used to perform data transfers between the uSDHC module and the card.
- RST is an output signal used to reset the MMC card.
- VSELECT is an output signal used to change the voltage of the external power supplier.

CD, WP, LCTL, RST, and VSELECT are all optional for system implementation. If uSDHC needs to support a 4-bit data transfer, DAT7~DAT4 can also be optional and tied to high. If the uSDHC does not support the HS400 mode, STROBE can also be optional and tied to low.

23.4 Clocks

The table found here describes the clock sources for uSDHC. For more information on clocking, see the Clocking chapter.

Table 23-4. uSDHC clocks

Clock name	Clock root	Description
hclk	ahb_clk_root	AHB bus clock
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_perclk	usdhc_clk_root	Base clock
ipg_clk_s	ipg_clk_root	Peripheral access clock for register accesses

23.5 uSDHC memory map/register definition

23.5.1 uSDHC register descriptions

This section includes the module memory map and detailed descriptions of all registers.

See the table below for the register memory map of uSDHC. All these registers only support 32-bit accesses.

NOTE

The uSDHC registers are 32-bit wide and only support 32-bit access.

23.5.1.1 uSDHC Memory map

uSDHC0 base address: 4037_0000h

uSDHC1 base address: 4038_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DMA System Address (DS_ADDR)	32	RW	0000_0000h
4h	Block Attributes (BLK_ATT)	32	RW	0001_0000h
8h	Command Argument (CMD_ARG)	32	RW	0000_0000h
Ch	Command Transfer Type (CMD_XFR_TYP)	32	RW	0000_0000h
10h	Command Response0 (CMD_RSP0)	32	RO	0000_0000h
14h	Command Response1 (CMD_RSP1)	32	RO	0000_0000h
18h	Command Response2 (CMD_RSP2)	32	RO	0000_0000h
1Ch	Command Response3 (CMD_RSP3)	32	RO	0000_0000h
20h	Data Buffer Access Port (DATA_BUFF_ACC_PORT)	32	RW	0000_0000h
24h	Present State (PRES_STATE)	32	RO	See description.
28h	Protocol Control (PROT_CTRL)	32	RW	0880_0020h
2Ch	System Control (SYS_CTRL)	32	RW	0080_800Fh
30h	Interrupt Status (INT_STATUS)	32	W1C	0000_0000h
34h	Interrupt Status Enable (INT_STATUS_EN)	32	RW	0000_0000h
38h	Interrupt Signal Enable (INT_SIGNAL_EN)	32	RW	0000_0000h
3Ch	Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)	32	RW	0000_0000h
40h	Host Controller Capabilities (HOST_CTRL_CAP)	32	RW	07F3_B407h
44h	Watermark Level (WTMK_LVL)	32	RW	0810_0810h
48h	Mixer Control (MIX_CTRL)	32	RW	8000_0000h
50h	Force Event (FORCE_EVENT)	32	WORZ	0000_0000h
54h	ADMA Error Status (ADMA_ERR_STATUS)	32	RO	0000_0000h
58h	ADMA System Address (ADMA_SYS_ADDR)	32	RW	0000_0000h
60h	DLL (Delay Line) Control (DLL_CTRL)	32	RW	0000_0000h
64h	DLL Status (DLL_STATUS)	32	RO	0000_0200h
68h	CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)	32	RW	0000_0000h
70h	Strobe DLL control (STROBE_DLL_CTRL)	32	RW	0000_0000h
74h	Strobe DLL status (STROBE_DLL_STATUS)	32	RO	0000_0200h
C0h	Vendor Specific Register (VEND_SPEC)	32	RW	2000_7809h
C4h	MMC Boot (MMC_BOOT)	32	RW	0000_0000h
C8h	Vendor Specific 2 Register (VEND_SPEC2)	32	RW	0000_1006h
CCh	Tuning Control (TUNING_CTRL)	32	RW	0021_2800h

23.5.1.2 DMA System Address (DS_ADDR)

23.5.1.2.1 Offset

Register	Offset
DS_ADDR	0h

23.5.1.2.2 Function

This register contains the physical system memory address used for DMA transfers.

23.5.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DS_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DS_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.2.4 Fields

Field	Function
31-0 DS_ADDR	System address **DMA system address / argument 2** When ACMD23_ARGU2_EN is set to 0, SDMA uses this register as system address and supports only 32-bit addressing mode. Auto CMD23 cannot be used with SDMA. When ACMD23_ARGU2_EN is set to 1, SDMA uses ADMA System Address register (05Fh – 058h) instead of this register to support both 32-bit and 64-bit addressing. This register is used only for Argument2 and SDMA may use Auto CMD23. 1. **SDMA system address** Because the address must be word (4 bytes) aligned, the least 2 bits are reserved, always 0. When uSDHC stops a DMA transfer, this register points out the system address of the next contiguous data position. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operation during transfers may return an invalid value. The host driver initializes this register before starting a DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this register. This register is protected during a data transfer. When data lines are active, write to this register is ignored. The host driver waits until the DLA bit in the Present State register is cleared, before writing to this register. The uSDHC internal DMA does not support a virtual memory system. It only supports continuous physical memory access. And due to AHB burst limitations, if the burst must cross the 1 KB boundary, uSDHC automatically changes SEQ burst type to NSEQ.

Field	Function
	Because this register supports dynamic address reflecting, when TC bit is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC bit is set. Such restriction is also listed in Software restrictions. 2. **Argument 2** This register is used with the Auto CMD23 to set a 32-bit block count value to the argument of the CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without ADMA, the available block count value is limited by the Block Count register. 65535 blocks is the maximum value in this case.

23.5.1.3 Block Attributes (BLK_ATT)

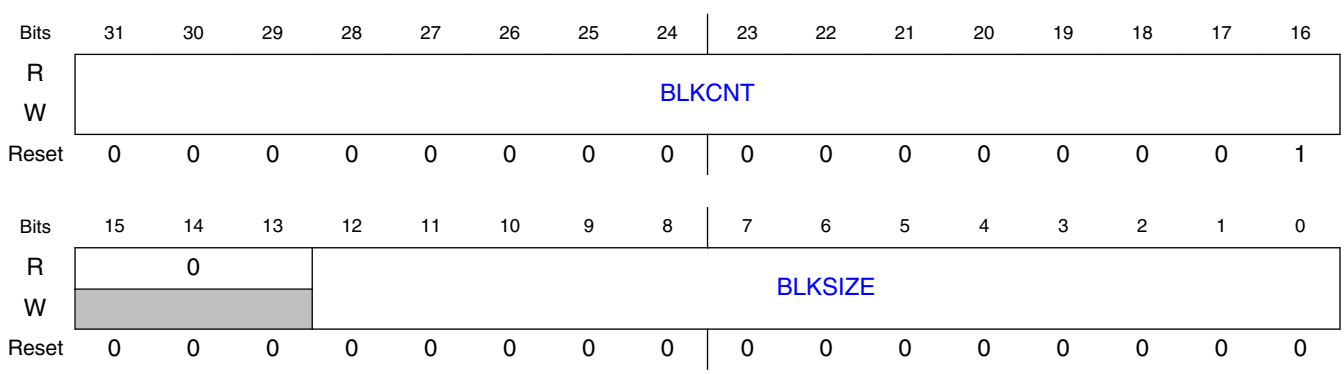
23.5.1.3.1 Offset

Register	Offset
BLK_ATT	4h

23.5.1.3.2 Function

This register is used to configure the number of data blocks and the number of bytes in each block.

23.5.1.3.3 Diagram



23.5.1.3.4 Fields

Field	Function
31-16 BLKCNT	Blocks count for current transfer This register is enabled when the Block Count Enable field in the Transfer Mode register is set to 1 and is valid only for multiple block transfers. For single block transfer, this register always reads as 1. The host driver sets this register to a value between 1 and the maximum block count. The uSDHC module decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to zero results in no data blocks being transferred. This register should be accessed only when no transaction is executing (that is, after transactions are stopped). During data transfer, read operations on this register may return an invalid value and write operations are ignored. When saving transfer content because of a Suspend command, the number of blocks yet to be transferred can be determined by reading this register. The reading of this register should be applied after transfer is paused by stop at block gap operation and before sending the command marked as suspend. This is because when the Suspend command is sent out, uSDHC treats the current transfer as aborted and change the BLKCNT register back to its original value instead of keeping the dynamical indicator of the remaining block count. When restoring transfer content prior to issuing a Resume command, the host driver restores the previously saved block count. **NOTE:** Although the BLKCNT field is 0 after reset, the read of reset value is 0x1. This is because when MSBSEL field is indicating a single block transfer, the read value of BLKCNT is always 1. 0000000000000000b - Stop count 0000000000000001b - 1 block 0000000000000010b - 2 blocks 1111111111111111b - 65535 blocks
15-13 —	Reserved
12-0 BLKSIZE	Transfer block size This register specifies the block size for block data transfers. Values ranging from 1 byte up to the maximum buffer size can be set. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored. 0000000000000b - No data transfer 0000000000001b - 1 byte 0000000000010b - 2 bytes 0000000000011b - 3 bytes 0000000000100b - 4 bytes 0000111111111b - 511 bytes 0001000000000b - 512 bytes 0100000000000b - 2048 bytes 1000000000000b - 4096 bytes

23.5.1.4 Command Argument (CMD_ARG)

23.5.1.4.1 Offset

Register	Offset
CMD_ARG	8h

23.5.1.4.2 Function

This register contains the SD/MMC command argument.

23.5.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CMDARG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CMDARG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.4.4 Fields

Field	Function
31-0 CMDARG	Command argument The SD/MMC command argument is specified as bits 39-8 of the command format in the SD or MMC specification. This register is write protected when the Command Inhibit (CMD) field in the Present State register is set.

23.5.1.5 Command Transfer Type (CMD_XFR_TYP)

23.5.1.5.1 Offset

Register	Offset
CMD_XFR_TYP	Ch

23.5.1.5.2 Function

This register is used to control the operation of data transfers. The host driver sets this register before issuing a command followed by a data transfer or before issuing a Resume command. To prevent data loss, uSDHC prevents writing to the bits, which are involved in the data transfer of this register, when data transfer is active. These bits are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

The host driver checks the Command Inhibit DAT field (PRES_STATE[CDIHB]) and the Command Inhibit CMD field (PRES_STATE[CIHB]) in the Present State register before writing to this register. When the CDIHB field in the Present State register is set, any attempt to send a command with data by writing to this register is ignored; when the CIHB field is set, any write to this register is ignored.

On sending commands with data transfer involved, it is mandatory that the block size is non-zero. Block count must also be non-zero, or indicated as a single block transfer (bit 5 of this register is '0' when written), or block count is disabled (bit 1 of this register is '0' when written), otherwise uSDHC ignores the sending of this command and do nothing. For write command, with all above restrictions, it is also mandatory that the write protect switch is not active (PRES_STATE[WPSPL] field of Present State register is '1'); otherwise, uSDHC also ignores the command.

If the commands with data transfer do not receive the response in 64 clock cycles, that is, if response time-out happens, uSDHC treats the external device, does not accept the command, and aborts the data transfer. In this scenario, the driver should issue the command again to retry the transfer. It is also possible that for some reason the card responds to the command but uSDHC does not receive the response, and if it is internal DMA (either simple DMA or ADMA) read operation, the external system memory is over-written by the internal DMA with data sent back from the card.

The table below shows the summary of how register settings determine the type of data transfer.

Table 23-5. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

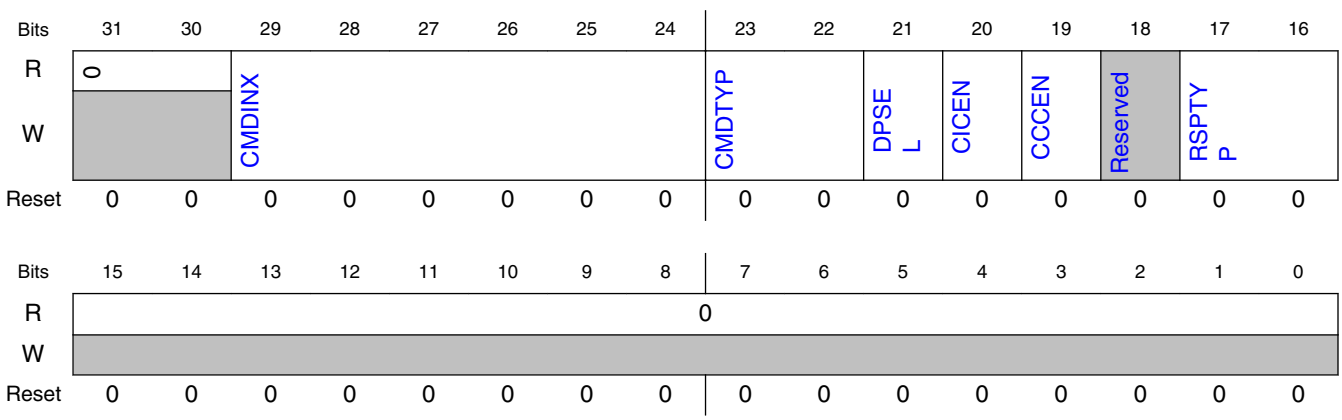
The table below shows the relationship between the Command Index Check Enable and the Command CRC Check Enable, regarding the Response Type bits as well as the name of the response type.

Table 23-6. Relationship between parameters and the name of the response type

Response type	Index check enable	CRC check enable	Name of response type
00	0	0	No response
01	0	1	R2
10	0	0	R3, R4
10	1	1	R1, R5, R6
11	1	1	R1b,R5b

- In the SDIO specification, response type notation for R5b is not defined. R5 includes R5b in the SDIO specification, but R5b is defined in this specification to specify that uSDHC checks the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command is used with R5b.
- The CRC fields for R3 and R4 are expected to be all 1 bits. The CRC check is disabled for these response types.

23.5.1.5.3 Diagram



23.5.1.5.4 Fields

Field	Function
31-30 —	Reserved
29-24 CMDINX	Command index These bits are set to the command number that is specified in bits 45-40 of the command-format in the SD Memory Card Physical Layer Specification and SDIO Card Specification.
23-22 CMDTYP	Command type There are three types of special commands: Suspend, Resume, and Abort. These bits are set to 00b for all other commands.

Table continues on the next page...

Field	Function
	• Suspend command: If the Suspend command succeeds, uSDHC assumes that the card bus has been released and that it is possible to issue the next command that uses the DATA line. Because uSDHC does not monitor the content of command response, it does not know if the Suspend command succeeded or not. It is the host driver's responsibility to check the status of the Suspend command and send another command marked as Suspend to inform uSDHC that a Suspend command was successfully issued. See Suspend Resume for more details. After the end bit of command is sent, uSDHC deasserts Read Wait for read transactions and stops checking busy for write transactions. In a 4-bit mode, the interrupt cycle starts. If the Suspend command fails, uSDHC maintains its current state, and the host driver restarts the transfer by setting the Continue Request field in the Protocol Control register. • Resume command: The host driver re-starts the data transfer by restoring the registers saved before sending the Suspend command and then sends the Resume command. The uSDHC module checks for a pending busy state before starting write transfers. • Abort command: If this command is set when executing a read transfer, uSDHC stops reads to the buffer. If this command is set when executing a write transfer, uSDHC stops driving the DATA line. After issuing the Abort command, the host driver should issue a software reset (Abort Transaction). 00b - Normal other commands 01b - Suspend CMD52 for writing bus suspend in CCCR 10b - Resume CMD52 for writing function select in CCCR 11b - Abort CMD12, CMD52 for writing I/O Abort in CCCR
21 DPSEL	Data present select This field is set to 1 to indicate that data is present and is transferred using the DATA line. It is set to 0 for the following: • Commands using only the CMD line (for example, CMD52) • Commands with no data transfer, but using the busy signal on DATA0 line (R1b or R5b (for example, CMD38)) **NOTE:** In resume command, this field is set, and other bits in this register is set the same as when the transfer was initially launched. When the write protect switch is on, (that is, the WPSPL field is active as '0'), any command with a write operation ignored. When this field is set, while the DTDSEL field is 0, writes to the register Transfer Type are ignored. 0b - No data present 1b - Data present
20 CICEN	Command index check enable If this field is set to 1, uSDHC checks the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this field is set to 0, the Index field is not checked. 0b - Disable command index check 1b - Enables command index check
19 CCCEN	Command CRC check enable If this field is set to 1, uSDHC checks the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this field is set to 0, the CRC field is not checked. The number of bits checked by the CRC field value changes according to the length of the response. See RSPTYP[1:0] and Command Transfer Type (CMD_XFR_TYP). 0b - Disables command CRC check 1b - Enables command CRC check
18 —	Reserved
17-16 RSPTYP	Response type select 00b - No response

Table continues on the next page...

Field	Function
	01b - Response length 136 10b - Response length 48 11b - Response length 48, check busy after response
15-0 —	Reserved

23.5.1.6 Command Response0 (CMD_RSP0)

23.5.1.6.1 Offset

Register	Offset
CMD_RSP0	10h

23.5.1.6.2 Function

This register is used to store part 0 of the response bits from the card.

23.5.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.6.4 Fields

Field	Function
31-0 CMDRSP0	Command response 0 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

23.5.1.7 Command Response1 (CMD_RSP1)

23.5.1.7.1 Offset

Register	Offset
CMD_RSP1	14h

23.5.1.7.2 Function

This register is used to store part 1 of the response bits from the card.

23.5.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.7.4 Fields

Field	Function
31-0 CMDRSP1	Command response 1 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

23.5.1.8 Command Response2 (CMD_RSP2)

23.5.1.8.1 Offset

Register	Offset
CMD_RSP2	18h

23.5.1.8.2 Function

This register is used to store part 2 of the response bits from the card.

23.5.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.8.4 Fields

Field	Function
31-0 CMDRSP2	Command response 2 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

23.5.1.9 Command Response3 (CMD_RSP3)

23.5.1.9.1 Offset

Register	Offset
CMD_RSP3	1Ch

23.5.1.9.2 Function

This register is used to store part 3 of the response bits from the card.

The table below describes the mapping of command responses from the SD bus to Command Response registers for each response type. In this table, R[ ] refers to a bit range within the response data as transmitted on the SD bus.

Table 23-7. Response bit definition for each response type

Response type	Meaning of response	Response field	Response register
R1,R1b (normal response)	Card status	R[39:8]	CMDRSP0
R1b (auto CMD12 response)	Card status for auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[23:0], CMDRSP2, CMDRSP1, CMDRSP0}
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O etc.	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (publish RCA)	New published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

This table shows that most responses with a length of 48 (R[47:0]) have 32-bits of the response data (R[39:8]) stored in the CMDRSP0 register. Responses of type R1b (Auto CMD12 responses) have response data bits (R[39:8]) stored in the CMDRSP3 register. Responses with length 136 (R[135:0]) have 120-bits of the response data (R[127:8]) stored in the CMDRSP0, 1, 2, and 3 registers.

To be able to read the response status efficiently, uSDHC only stores part of the response data in the Command Response registers. This enables the host driver to efficiently read 32-bits of response data in one read cycle on a 32-bit bus system. Parts of the response, the Index field and the CRC, are checked by uSDHC (as specified by the Command Index Check Enable and the Command CRC Check Enable bits in the Transfer Type register) and generate an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, uSDHC checks R[47:1], and if the response length is 136 the uSDHC checks R[119:1].

Because uSDHC may have a multiple block data transfer executing concurrently with a CMD_wo_DAT command, uSDHC stores the Auto CMD12 response in the CMDRSP3 register. The CMD_wo_DAT response is stored in CMDRSP0. This allows uSDHC to avoid overwriting the Auto CMD12 response with the CMD_wo_DAT and vice versa. When uSDHC modifies part of the Command Response registers, as shown in the table above, it preserves the unmodified bits.

23.5.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.9.4 Fields

Field	Function
31-0 CMDRSP3	Command response 3 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this register for each response type.

23.5.1.10 Data Buffer Access Port (DATA_BUFF_ACC_PORT)

23.5.1.10.1 Offset

Register	Offset
DATA_BUFF_ACC_PORT	20h

23.5.1.10.2 Function

This is a 32-bit data port register used to access the internal buffer.

23.5.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATCONT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATCONT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.10.4 Fields

Field	Function
31-0 DATCONT	Data content The Buffer Data Port register is for 32-bit data access by the Arm platform. When the internal DMA is enabled, any write to this register is ignored, and any read from this register always yields 0s.

23.5.1.11 Present State (PRES_STATE)

23.5.1.11.1 Offset

Register	Offset
PRES_STATE	24h

23.5.1.11.2 Function

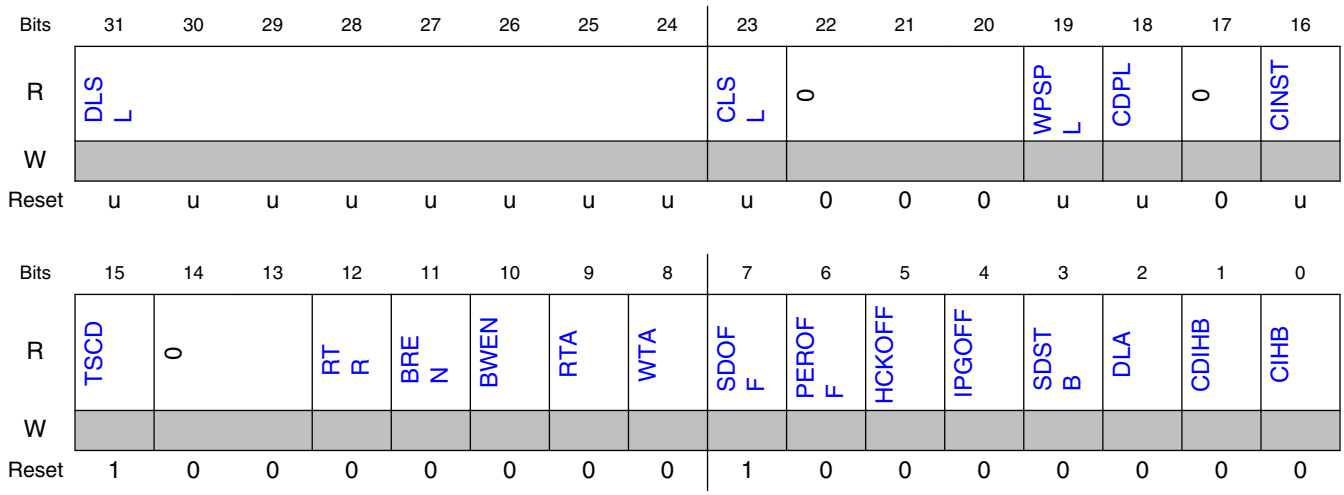
The host driver can get status of uSDHC from this 32-bit read only register.

The host driver can issue CMD0, CMD12, CMD13 (for memory) and CMD52 (for SDIO) when the DATA lines are busy during a data transfer. These commands can be issued when Command Inhibit (CMD) is set to zero. Other commands are issued when Command Inhibit (DATA) is set to zero. Possible changes to the SD Physical Specification may add other commands to this list in the future.

NOTE

The reset value of Present State register depends on board connectivity.

23.5.1.11.3 Diagram



23.5.1.11.4 Fields

Field	Function
31-24 DLSL	DATA[7:0] line signal level This status is used to check the DATA line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DATA0. The reset value is affected by the external pull-up / pull-down resistors. By default, the read value of this field field after reset is 8'b11110111, when DATA3 is pulled down and the other lines are pulled up. 00000000b - Data 0 line signal level 00000001b - Data 1 line signal level 00000010b - Data 2 line signal level 00000011b - Data 3 line signal level 00000100b - Data 4 line signal level 00000101b - Data 5 line signal level 00000110b - Data 6 line signal level 00000111b - Data 7 line signal level
23 CLSL	CMD line signal level This status is used to check the CMD line level to recover from errors, and for debugging. The reset value is affected by the external pull-up / pull-down resistor, by default, the read value of this field after reset is 1'b1, when the command line is pulled up.
22-20 —	Reserved
19 WPSPL	Write protect switch pin level The Write Protect switch is supported for memory and combo cards. This field reflects the inverted value of the WP pin of the card socket. A software reset does not affect this field. The reset value is affected by the external write protect switch. If the WP pin is not used, it should be tied low, so that the reset value of this field is high and write is enabled. 0b - Write protected (WP = 1) 1b - Write enabled (WP = 0)

Table continues on the next page...

Field	Function
18 CDPL	Card detect pin level This field reflects the inverse value of the CD_B pin for the card socket. Debouncing is not performed on this field. This field may be valid, but is not guaranteed, because of propagation delay. Use of this field is limited to testing because it must be debounced by software. A software reset does not affect this field. A write to the Force Event Register does not affect this field. The reset value is affected by the external card detection pin. This field shows the value on the CD_B pin (that is, when a card is inserted in the socket, it is 0 on the CD_B input, and consequently, the CDPL reads 1. 0b - No card present (CD_B = 1) 1b - Card present (CD_B = 0)
17 —	Reserved
16 CINST	Card inserted This field indicates whether a card has been inserted. The uSDHC module debounces this signal so that the host driver does not need to wait for it to stabilize. Changing from a 0 to 1 generates a Card Insertion interrupt in the Interrupt Status register. Changing from a 1 to 0 generates a Card Removal interrupt in the Interrupt Status register. A write to the Force Event Register does not affect this field. The Software Reset For All in the System Control register does not affect this field. A software reset does not affect this field. 0b - Power on reset or no card 1b - Card inserted
15 TSCD	Tape select change done This field indicates the delay setting is effective after write CLK_TUNE_CTRL_STATUS register. 0b - Delay cell select change is not finished. 1b - Delay cell select change is finished.
14-13 —	Reserved
12 RTR	Re-Tuning Request (only for SD3.0 SDR104 mode and EMMC HS200 mode) Host controller may request host driver to execute re-tuning sequence by setting this field when the data window is shifted by temperature drift and a tuned sampling point does not have a good margin to receive correct data. This field is cleared when a command is issued with setting Execute Tuning field in MIXER_CTRL register. Changing of this field from 0 to 1 generates Re-Tuning Event. See Interrupt status registers for more detail. This field isn't set to 1 if Sampling Clock Select in the MIXER_CTRL register is set to 0 (using fixed sampling clock). 0b - Fixed or well tuned sampling clock 1b - Sampling clock needs re-tuning
11 BREN	Buffer read enable This status field is used for non-DMA read transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this field is high, valid data greater than the watermark level exist in the buffer. A change of this field from 1 to 0 occurs when some reads from the buffer (read DATPORT (Base + 0x20)) are made and the buffer hasn't valid data greater than the watermark level. A change of this field from 0 to1 occurs when there is enough valid data ready in the buffer and the Buffer Read Ready interrupt has been generated and enabled. 0b - Read disable

Table continues on the next page...

Field	Function
	1b - Read enable
10 BWEN	Buffer write enable This status field is used for non-DMA write transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates if space is available for write data. If this field is 1, valid data greater than the watermark level can be written to the buffer. A change of this field from 1 to 0 occurs when some writes to the buffer (write DATPORT(Base + 0x20)) are made and the buffer hasn't valid space greater than the watermark level. A change of this field from 0 to 1 occurs when the buffer can hold valid data greater than the write watermark level and the Buffer Write Ready interrupt is generated and enabled. 0b - Write disable 1b - Write enable
9 RTA	Read transfer active This status field is used for detecting completion of a read transfer. This field is set for either of the following conditions: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer A transfer complete interrupt is generated when this field changes to 0. This field is cleared for either of the following conditions: • When the last data block as specified by block length is transferred to the System, that is, all data are read away from uSDHC internal buffer. • When all valid data blocks have been transferred from uSDHC internal buffer to the System and no current block transfers are being sent because of the Stop At Block Gap Request being set to 1. 0b - No valid data 1b - Transferring data
8 WTA	Write transfer active This status field indicates a write transfer is active. If this field is 0, it means no valid write data exists in uSDHC. This field is set in either of the following cases: • After the end field of the write command • When writing 1 to the Continue Request field in the Protocol Control register to restart a write transfer This field is cleared in either of the following cases: • After getting the CRC status of the last data block as specified by the transfer count (Single and Multiple) • After getting the CRC status of any block where data transmission is about to be stopped by a Stop At Block Gap Request During a write transaction, a Block Gap Event interrupt is generated when this field is changed to 0, as result of the Stop At Block Gap Request being set. This status is useful for the host driver in determining when to issue commands during Write Busy state. 0b - No valid data 1b - Transferring data
7 SDOFF	SD clock gated off internally This status field indicates that the SD clock is internally gated off, because of buffer over / under-run or read pause without read wait assertion, or the driver set FRC_SDCLK_ON field is 0 to stop the SD clock in idle status. This field is for the host driver to debug data transaction on the SD bus.

Table continues on the next page...

Field	Function
	0b - SD clock is active. 1b - SD clock is gated off.
6 PEROFF	IPG_PERCLK gated off internally This status field indicates that the IPG_PERCLK is internally gated off. This field is for the host driver to debug transaction on the SD bus. When IPG_CLK_SOFT_EN is cleared, IPG_PERCLK is gated off, otherwise IPG_PERCLK is always active. 0b - IPG_PERCLK is active. 1b - IPG_PERCLK is gated off.
5 HCKOFF	HCLK gated off internally This status field indicates that the HCLK is internally gated off. This field is for the host driver to debug during a data transfer. 0b - HCLK is active. 1b - HCLK is gated off.
4 IPGOFF	Peripheral clock gated off internally This status field indicates that the peripheral clock is internally gated off. This field is for the host driver to debug. 0b - Peripheral clock is active. 1b - Peripheral clock is gated off.
3 SDSTB	SD clock stable This status field indicates that the internal card clock is stable. This field is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear FRC_SDCLK_ON field in System Control register to remove glitches on the card clock when the frequency is changing. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. 0b - Clock is changing frequency and not stable. 1b - Clock is stable.
2 DLA	Data line active This status field indicates whether one of the DATA lines on the SD bus is in use. In the case of read transactions: This status indicates if a read transfer is executing on the SD bus. Changes in this value from 1 to 0, between data blocks, generates a Block Gap Event interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer This field is cleared in either of the following cases: • When the end field of the last data block is sent from the SD bus to uSDHC. • When the Read Wait state is stopped by a Suspend command and the DATA2 line is released. The uSDHC module waits at the next block gap by driving Read Wait at the start of the interrupt cycle. If the Read Wait signal is already driven (data buffer cannot receive data), uSDHC can wait for a current block gap by continuing to drive the Read Wait signal. It is necessary to support Read Wait to use the suspend / resume function. This field remains 1 during Read Wait. In the case of write transactions: This status indicates that a write transfer is executing on the SD bus. Changes in this value from 1 to 0 generate a Transfer Complete interrupt in the Interrupt Status register.

Table continues on the next page...

Field	Function
	This field is set in either of the following cases: • After the end field of the write command • When writing to 1 to the Continue Request field in the Protocol Control register to continue a write transfer This field is cleared in either of the following cases: • When the SD card releases Write Busy of the last data block, uSDHC also detects if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, uSDHC assumes the card drive "Not Busy". • When the SD card releases write busy, prior to waiting for write transfer, and because of a Stop At Block Gap Request. In the case of command with busy pending: This status indicates that a busy state follows the command and the data line is in use. This field is cleared when the DATA0 line is released. 0b - DATA line inactive 1b - DATA line active
1 CDIHB	Command inhibit (DATA) This status field is generated if either the DAT Line Active or the Read Transfer Active is set to 1. If this field is 0, it indicates that uSDHC can issue the next SD / MMC Command. Commands with a busy signal belong to Command Inhibit (DATA) (for example. R1b, R5b type). Changing from 1 to 0 generates a Transfer Complete interrupt in the Interrupt Status register. **NOTE:** The SD host driver can save registers for a suspend transaction after this field has changed from 1 to 0. 0b - Can issue command that uses the DATA line 1b - Cannot issue command that uses the DATA line
0 CIHB	Command inhibit (CMD) If this status bit is 0, it indicates that the CMD line is not in use and uSDHC can issue a SD / MMC command using the CMD line. This field is set also immediately after the Transfer Type register is written. This field is cleared when the command response is received. Even if the Command Inhibit (DATA) is set to 1, commands using only the CMD line can be issued if this field is 0. Changing from 1 to 0 generates a Command Complete interrupt in the Interrupt Status register. If uSDHC cannot issue the command because of a command conflict error (see Command CRC Error) or because of a Command Not Issued By Auto CMD12 Error, this field remains 1 and the Command Complete is not set. The Status of issuing an auto CMD12 does not show on this field. 0b - Can issue command using only CMD line 1b - Cannot issue command

23.5.1.12 Protocol Control (PROT_CTRL)

23.5.1.12.1 Offset

Register	Offset
PROT_CTRL	28h

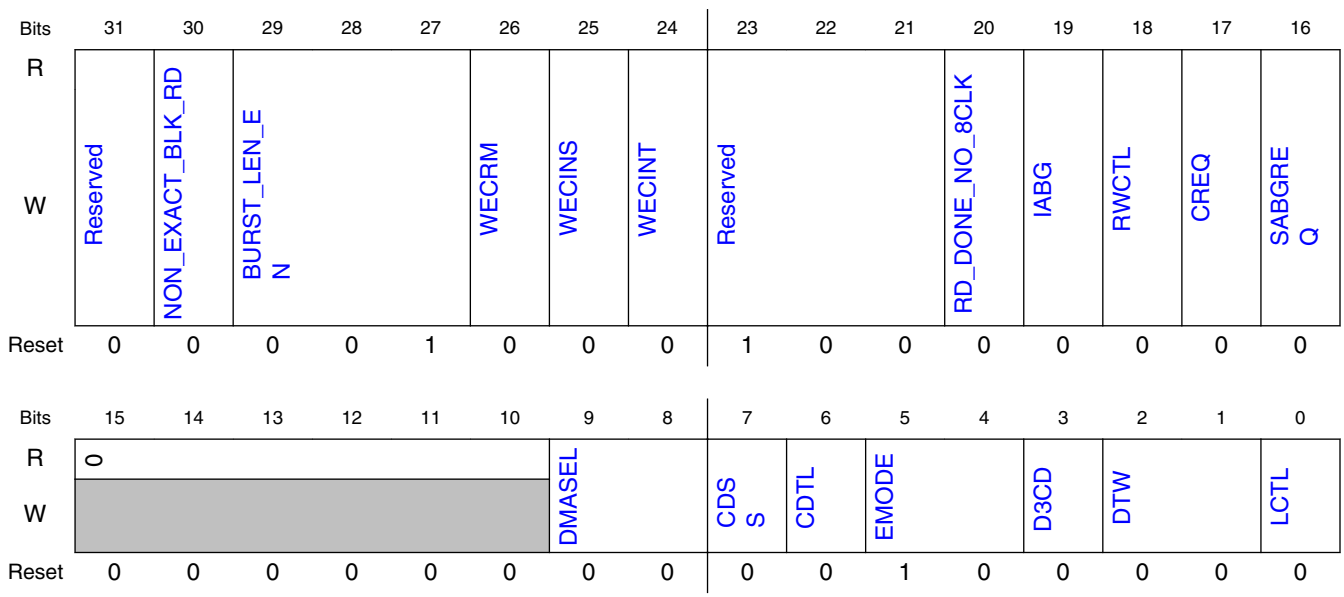
23.5.1.12.2 Function

There are three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether uSDHC issues a Suspend command or the SD card accepts the Suspend command.

- If the host driver does not issue a Suspend command, the Continue request is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card accepts it, a Resume command is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card does not accept it, the Continue request is used to restart the transfer.

Any time stop at block gap request stops the data transfer, the host driver waits for a Transfer Complete (in the Interrupt Status register), before attempting to restart the transfer. When restarting the data transfer by Continue Request, the host driver clears the Stop At Block Gap Request before or simultaneously.

23.5.1.12.3 Diagram



23.5.1.12.4 Fields

Field	Function
31	Reserved
—	Always write as 0

Table continues on the next page...

Field	Function
30 NON_EXACT_BLK_RD	Non-exact block read Current block read is non-exact block read. It is only used for SDIO. 0b - The block read is exact block read. Host driver does not need to issue abort command to terminate this multi-block read. 1b - The block read is non-exact block read. Host driver needs to issue abort command to terminate this multi-block read.
29-27 BURST_LEN_EN	BURST length enable for INCR, INCR4 / INCR8 / INCR16, INCR4-WRAP / INCR8-WRAP / INCR16-WRAP This is used to enable / disable the burst length for the external AHB2AXI bridge. It is useful especially for INCR transfer because without burst length indicator, the AHB2AXI bridge does not know the burst length in advance. Without burst length indicator, AHB INCR transfers can only be converted to SINGLEs on the AXI side. 1xxb - Burst length is enabled for INCR4-WRAP / INCR8-WRAP / INCR16-WRAP. x1xb - Burst length is enabled for INCR4 / INCR8 / INCR16. xx1b - Burst length is enabled for INCR.
26 WECRM	Wakeup event enable on SD card removal This field enables a wakeup event, via a card removal, in the Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not affect this field. When this field is set, the Card Removal Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Removal Status and uSDHC interrupt. 0b - Disables wakeup event enable on SD card removal 1b - Enables wakeup event enable on SD card removal
25 WECINS	Wakeup event enable on SD card insertion This field enables a wakeup event, via a card insertion, in the Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not affect this field. When this field is set, the Card Insertion Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Insertion Status and uSDHC interrupt. 0b - Disable wakeup event enable on SD card insertion 1b - Enable wakeup event enable on SD card insertion
24 WECINT	Wakeup event enable on card interrupt This field enables a wakeup event, via a card interrupt, in the Interrupt Status register. This field can be set to 1 if FN_WUS (Wake Up Support) in CIS is set to 1. When this field is set, the Card Interrupt Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Interrupt Status and uSDHC interrupt. 0b - Disables wakeup event enable on card interrupt 1b - Enables wakeup event enable on card interrupt
23-21 —	Reserved Always write as 3'b100
20 RD_DONE_NO_8CLK	Read performed number 8 clock According to the SD/MMC spec, for read data transaction, 8 clocks are needed after the end field of the last data block. So, by default(RD_DONE_NO_8CLK=0), 8 clocks are active after the end field of the last read data transaction. However, these 8 clocks should not be active if user wants to use stop at block gap (include the auto stop at block gap in boot mode) feature for read and the RWCTL field (bit18) is not enabled. In this case, software should set RD_DONE_NO_8CLK to avoid these 8 clocks. Otherwise, the device might send extra data to uSDHC while uSDHC ignores these data. In a summary, this field should be set only if the use case needs to use stop at block gap feature while the device can't support the read wait feature.

Table continues on the next page...

Field	Function
19 IABG	Interrupt at block gap This field is valid only in 4-bit mode, of the SDIO card, and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If the SDIO card cannot signal an interrupt during a multiple block transfer, this field should be set to 0 to avoid an inadvertent interrupt. When the host driver detects an SDIO card insertion, it sets this field according to the CCCR of the card. 0b - Disables interrupt at block gap 1b - Enables interrupt at block gap
18 RWCTL	Read wait control The read wait function is optional for SDIO cards. If the card supports read wait, set this field to enable use of the read wait protocol to stop read data using the DATA2 line. Otherwise, uSDHC has to stop the SD clock to hold read data, which restricts commands generation. When the host driver detects an SDIO card insertion, it sets this field according to the CCCR of the card. If the card does not support read wait, this field should never be set to 1; otherwise, DATA line conflicts might occur. If this field is set to 0, stop at block gap during read operation is also supported, but uSDHC stops the SD clock to pause reading operation. 0b - Disables read wait control and stop SD clock at block gap when SABGREQ field is set 1b - Enables read wait control and assert read wait without stopping SD clock at block gap when SABGREQ field is set
17 CREQ	Continue request This field is used to restart a transaction which was stopped using the stop at block gap request. When a suspend operation is not accepted by the card, it is also by setting this field to restart the paused transfer. To cancel stop at the block gap, set stop at block gap request to 0 and set this field to 1 to restart the transfer. The uSDHC module automatically clears this field, therefore it is not necessary for the host driver to set this field to 0. If both stop at block gap request and this field are 1, the continue request is ignored. 0b - No effect 1b - Restart
16 SABGREQ	Stop at block gap request This field is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the transfer complete is set to 1, indicating a transfer completion, the host driver leaves this field set to 1. Clearing both the stop at block gap request and continue request does not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The uSDHC module supports the stop at block gap request for write transfers, but for read transfers it requires that the SDIO card support read wait. Therefore, the host driver does not set this field during read transfers unless the SDIO card supports Read Wait and has set the read wait control to 1; otherwise, uSDHC stops the SD bus clock to pause the read operation during block gap. In the case of write transfers in which the host driver writes data to the Data Port register, the host driver sets this field after all block data is written. If this field is set to 1, the host driver does not write data to the Data Port register after a block is sent. Once this field is set, the host driver does not clear this field before the Transfer Complete field in Interrupt Status register is set, otherwise uSDHC's behavior is undefined. This field effects read transfer active, write transfer active, DATA Line Active and Command Inhibit (DATA) in the Present State register. 0b - Transfer 1b - Stop
15-10 —	Reserved
9-8 DMASEL	DMA select This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation.

Table continues on the next page...

Field	Function
	00b - No DMA or simple DMA is selected. 01b - ADMA1 is selected. 10b - ADMA2 is selected. 11b - Reserved
7 CDSS	Card detect signal selection This field selects the source for the card detection. 0b - Card detection level is selected (for normal purpose). 1b - Card detection test level is selected (for test purpose).
6 CDTL	Card detect test level This bit is enabled while the card detection signal selection is set to 1 and it indicates card insertion. 0b - Card detect test level is 0, no card inserted 1b - Card detect test level is 1, card inserted
5-4 EMODE	Endian mode The uSDHC module supports all three endian modes in data transfer. See Data buffer for more details. 00b - Big endian mode 01b - Half word big endian mode 10b - Little endian mode 11b - Reserved
3 D3CD	DATA3 as card detection pin If this field is set, DATA3 should be pulled down to act as a card detection pin. Be cautious when using this feature, because DATA3 is also a chip-select for the SPI mode. A pull-down on this pin and CMD0 might set the card into the SPI mode, which uSDHC does not support. 0b - DATA3 does not monitor card insertion 1b - DATA3 as card detection pin
2-1 DTW	Data transfer width This field selects the data width of the SD bus for a data transfer. The host driver sets it to match the data width of the card. Possible data transfer width is 1-bit, 4-bits or 8-bits. 00b - 1-bit mode 01b - 4-bit mode 10b - 8-bit mode 11b - Reserved
0 LCTL	LED control This field, fully controlled by the host driver, is used to caution the user not to remove the card while the card is being accessed. If the software is going to issue multiple SD commands, this field can be set during all these transactions. It is not necessary to change for each transaction. When the software issues multiple SD commands, setting the field once before the first command is sufficient: it is not necessary to reset the bit between commands. 0b - LED off 1b - LED on

23.5.1.13 System Control (SYS_CTRL)

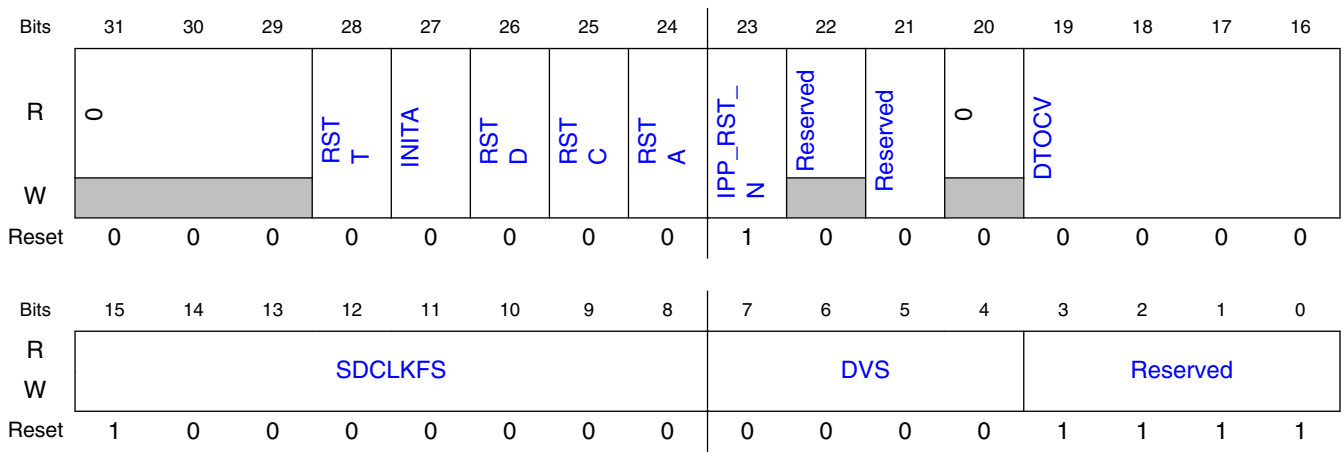
23.5.1.13.1 Offset

Register	Offset
SYS_CTRL	2Ch

23.5.1.13.2 Function

This register provides control of the system. See detail in the field description.

23.5.1.13.3 Diagram



23.5.1.13.4 Fields

Field	Function
31-29 —	Reserved
28 RSTT	Reset tuning When set this field to 1, it resets tuning circuit. After tuning circuits are reset, bit value is 0. Clearing execute_tuning field in AUTOCMD12_ERR_STATUS also sets this field to 1 to reset tuning circuit
27 INITA	Initialization active When this field is set, 80 SD-clocks are sent to the card. After the 80 clocks are sent, this field is self cleared. This field is very useful during the card power-up period when 74 SD-clocks are needed and the clock auto gating feature is enabled. Writing 1 to this bit when this field is already 1 has no effect. Writing 0 to this field at any time has no effect. When either of the CIHB and CDIHB fields in the Present State register are set, writing 1 to this field is ignored (that is, when command line or data lines are active, write to this field is not allowed). On the other-hand, when this field is set, that is, during initialization active period, it is allowed to issue command, and the command bit stream appears on the CMD pad after all 80 clock cycles are done. So, when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs to send 80 cycles to the card and does not want to wait till this field is self cleared.
26	Software reset for data line

Table continues on the next page...

Field	Function
RSTD	Only part of the data circuit is reset. DMA circuit is also reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Data Port register • Buffer is cleared and initialized • Present State register • Buffer read enable • Buffer write enable • Read transfer active • Write transfer active • DATA line active • Command Inhibit (DATA) Protocol Control register • Continue request • Stop At Block Gap Request Interrupt Status register • Buffer read ready • Buffer write ready • DMA interrupt • Block gap event • Transfer complete **NOTE:** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
25 RSTC	Software reset for CMD line Only part of the command circuit is reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Present State Register Command Inhibit (CMD) • Interrupt Status register Command Complete 0b - No reset 1b - Reset
24 RSTA	Software reset for all This reset effects the entire host controller except for the card detection circuit. Register bits of type ROC, RW, RW1C, RWAC are cleared. During its initialization, the host driver is set this field to 1 to reset uSDHC. The uSDHC module resets this field to 0 when the capabilities registers are valid and the host driver can read them. Additional use of Software Reset For All does not affect the value of the capabilities registers. After this field is set, it is recommended that the host driver reset the external card and re-initialize it. After this field is set, the software should wait for self-clear. In tuning process, after every CMD19 is finished, this field is set to retest uSDHC. **NOTE:** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
23 IPP_RST_N	Hardware reset This register's value is output to card through pad directly to hardware reset pin of the card if the card supports this feature.
22 —	Reserved
21	Reserved

Table continues on the next page...

Field	Function
—	
20 —	Reserved
19-16 DTOCV	Data timeout counter value This value determines the interval by which DAT line timeouts are detected. See the Data Timeout Error field in the Interrupt Status register for information on factors that dictate time-out generation. Time-out clock frequency is generated by dividing the base clock SDCLK value by this value. The host driver can clear the Data Timeout Error Status Enable (in the Interrupt Status Enable register) to prevent inadvertent time-out events. 0000b - SDCLK x 2^{14} 0001b - SDCLK x 2^{15} 0010b - SDCLK x 2^{16} 0011b - SDCLK x 2^{17} 0100b - SDCLK x 2^{18} 0101b - SDCLK x 2^{19} 0110b - SDCLK x 2^{20} 0111b - SDCLK x 2^{21} 1000b - SDCLK x 2^{22} 1001b - SDCLK x 2^{23} 1010b - SDCLK x 2^{24} 1011b - SDCLK x 2^{25} 1100b - SDCLK x 2^{26} 1101b - SDCLK x 2^{27} 1110b - SDCLK x 2^{28} 1111b - SDCLK x 2^{29}
15-8 SDCLKFS	SDCLK frequency select This register is used to select the frequency of the SDCLK pin. The frequency is not programmed directly, rather this register holds the prescaler (this register) and divisor (next register) of the Base Clock Frequency register. In Single Data Rate mode (DDR_EN field of MIXERCTRL is '0') Only the following settings are allowed: 80h) Base clock divided by 256 40h) Base clock divided by 128 20h) Base clock divided by 64 10h) Base clock divided by 32 08h) Base clock divided by 16 04h) Base clock divided by 8 02h) Base clock divided by 4 01h) Base clock divided by 2 00h) Base clock divided by 1 While in Dual Data Rate mode (DDR_EN field of MIXERCTRL is '1') Only the following settings are allowed: 80h) Base clock divided by 512 40h) Base clock divided by 256 20h) Base clock divided by 128

Table continues on the next page...

Field	Function
	10h) Base clock divided by 64 08h) Base clock divided by 32 04h) Base clock divided by 16 02h) Base clock divided by 8 01h) Base clock divided by 4 00h) Base clock divided by 2 When the software changes the DDR_EN field, SDCLKFS might need to be changed also. In Single Data Rate mode, setting 00h bypasses the frequency prescaler of the SD clock. Multiple bits must not be set, or the behavior of this prescaler is undefined. The two default divider values can be calculated by the frequency of ipg_perclk and the following Divisor bits. The frequency of SDCLK is set by the following formula: Clock Frequency = (Base Clock) / (prescaler x divisor) For example, in Single Data Rate mode, if the Base Clock Frequency is 96 MHz, and the target frequency is 25 MHz, then choosing the prescaler value of 01h and divisor value of 1h yields 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to approach a clock value of 400 kHz, the prescaler value of 08h and divisor value of eh yields the exact clock value of 400 kHz. The reset value of this field is 80h, so if the input Base Clock (ipg_perclk) is about 96 MHz, the default SD clock after reset is 375 kHz. According to the SD Physical Specification Version 1.1 and the SDIO Card Specification Version 1.2, the maximum SD clock frequency is 50 MHz and is never exceed this limit. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. If setting SDCLKFS and DVS can generate the same clock frequency,(for example, in SDR mode, SDCLKFS = 01h is same as DVS = 01h.) SDCLKFS is highly recommended.
7-4 DVS	Divisor This register is used to provide a more exact divisor to generate the desired SD clock frequency. Note the divider can even support odd divisors without deterioration of duty cycle. Before changing clock divisor value (SDCLKFS or DVS), Host driver should make sure the SDSTB field is high. The settings are as follows: 0000b - Divide-by-1 0001b - Divide-by-2 1110b - Divide-by-15 1111b - Divide-by-16
3-0 —	Reserved Always write as 1.

23.5.1.14 Interrupt Status (INT_STATUS)

23.5.1.14.1 Offset

Register	Offset
INT_STATUS	30h

23.5.1.14.2 Function

An interrupt is generated when the normal interrupt signal enable is enabled and at least one of the status fields is set to 1. For all fields, writing 1 to a bit clears it; writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. For card interrupt, before writing 1 to clear, it is required that the card stops asserting the interrupt, meaning that when the card driver services the interrupt condition; otherwise, the CINT field is asserted again.

The table below shows the relationship between the command timeout error and the command complete.

Table 23-8. uSDHC status for command timeout error/command complete bit combinations

Command complete	Command timeout error	Meaning of the status
0	0	X
X	1	Response not received within 64 SDCLK cycles
1	0	Response received

The table below shows the relationship between the transfer complete and the data timeout error.

Table 23-9. uSDHC status for data timeout error/transfer complete bit combinations

Transfer complete	Data timeout error	Meaning of the status
0	0	X
0	1	Timeout occurred during transfer
1	X	Data transfer complete

The table below shows the relationship between the command CRC error and command timeout error.

Table 23-10. uSDHC status for command CRC error/command timeout error bit combinations

Command complete	Command timeout error	Meaning of the status
0	0	No error
0	1	Response timeout error

Table continues on the next page...

Table 23-10. uSDHC status for command CRC error/command timeout error bit combinations (continued)

Command complete	Command timeout error	Meaning of the status
1	0	Response CRC error
1	1	CMD line conflict

23.5.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAE	0	TNE	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W				W1C		W1C		W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TP	0	RTE	0			CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W		W1C		W1C				W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

23.5.1.14.4 Fields

Field	Function
31-29 —	Reserved
28 DMAE	DMA error Occurs when an Internal DMA transfer has failed. This field is set to 1, when some error occurs in the data transfer. This error can be caused by either simple DMA or ADMA, depending on which DMA is in use. The value in DMA System Address register is the next fetch address where the error occurs. Because any error corrupts the whole data block, the host driver restarts the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DS_ADDR value or from the remaining number of blocks and the block size. 0b - No error 1b - Error
27 —	Reserved
26 TNE	Tuning error: (only for SD3.0 SDR104 mode and EMMC HS200 mode) This field is set when an unrecoverable error is detected in a tuning circuit. By detecting Tuning Error, host driver needs to abort a command executing and perform tuning.

Table continues on the next page...

Field	Function
25 —	Reserved
24 AC12E	Auto CMD12 error Occurs when detecting that one of the fields in the Auto CMD12 Error Status register has changed from 0 to 1. This field is set to 1, not only when the errors in Auto CMD12 occur, but also, when the Auto CMD12 is not executed due to the previous command error. 0b - No error 1b - Error
23 —	Reserved
22 DEBE	Data end bit error Occurs either when detecting 0 at the end field position of read data that uses the DATA line, or at the end field position of the CRC. This field is not asserted in tuning process. 0b - No error 1b - Error
21 DCE	Data CRC error Occurs when detecting a CRC error when transferring read data that uses the DATA line, or when detecting the Write CRC status having a value other than 010. This field is not asserted in tuning process. 0b - No error 1b - Error
20 DTOE	Data timeout error Occurs when detecting one of following time-out conditions. • Busy time-out for R1b, R5b type • Busy time-out after Write CRC status • Read Data time-out. This field is not asserted in tuning process. 0b - No error 1b - Time out
19 CIE	Command index error Occurs if a command index error occurs in the command response. This field is not asserted in tuning process. 0b - No error 1b - Error
18 CEBE	Command end bit error Occurs when detecting that the end field of a command response is 0. This field is not asserted in tuning process. 0b - No error 1b - End bit error generated
17 CCE	Command CRC error Command CRC Error is generated in two cases.

Table continues on the next page...

Field	Function
	• If a response is returned and the Command Timeout Error is set to 0 (indicating no time-out), this field is set when detecting a CRC error in the command response. • The uSDHC module detects a CMD line conflict by monitoring the CMD line when a command is issued. If uSDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then uSDHC aborts the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error should also be set to 1 to distinguish CMD line conflict. This field is not asserted in tuning process. 0b - No error 1b - CRC error generated
16 CTOE	Command timeout error Occurs only if no response is returned within 64 SDCLK cycles from the end field of the command. If uSDHC detects a CMD line conflict, in which case a Command CRC Error is also be set (as shown in Interrupt Status (INT_STATUS)), this field is set without waiting for 64 SDCLK cycles. This is because the command is aborted by uSDHC. This field is not asserted in tuning process. 0b - No error 1b - Time out
15 —	Reserved
14 TP	Tuning pass:(only for SD3.0 SDR104 mode and EMMC HS200 mode) Current CMD19 transfer is done successfully. That is, current sampling point is correct.
13 —	Reserved
12 RTE	Re-tuning event: (only for SD3.0 SDR104 mode and EMMC HS200 mode) This status is set if Re-Tuning Request in the Present State register changes from 0 to 1. host controller requests host driver to perform re-tuning for next data transfer. Current data transfer (not large block count) can be completed without re-tuning. 0b - Re-tuning is not required. 1b - Re-tuning should be performed.
11-9 —	Reserved
8 CINT	Card interrupt This status field is set when an interrupt signal is detected from the external card. In 1-bit mode, uSDHC detects the Card Interrupt without the SD clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so the interrupt from card can only be sampled during interrupt cycle, introducing some delay between the interrupt signal from the SDIO card and the interrupt to the host system. Writing this field to 1 can clear this field, but as the interrupt source from the SDIO card does not clear, this field is set again. to clear this field, it is required to reset the interrupt source from the external card followed by a writing 1 to this field. When this status has been set, and the host driver needs to service this interrupt, the Card Interrupt Signal Enable in the Interrupt Signal Enable register should be 0 to stop driving the interrupt signal to the host system. After completion of the card interrupt service (It should reset the interrupt sources in the SDIO card and the interrupt signal might not be asserted), write 1 to clear this field, set the Card Interrupt Signal Enable to 1, and start sampling the interrupt signal again. 0b - No card interrupt 1b - Generate card interrupt
7	Card removal

Table continues on the next page...

Field	Function
CRM	This status field is set if the Card Inserted field in the Present State register changes from 1 to 0. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if no card is inserted. to leave it cleared, clear the Card Removal Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or inserted 1b - Card removed
6 CINS	Card insertion This status field is set if the Card Inserted field in the Present State register changes from 0 to 1. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if a card is inserted. to leave it cleared, clear the Card Inserted Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or removed 1b - Card inserted
5 BRR	Buffer read ready This status field is set if the Buffer Read Enable field, in the Present State register, changes from 0 to 1. See the Buffer Read Enable field in the Present State register for additional information. This field indicates that cmd19 is finished in tuning process. 0b - Not ready to read buffer 1b - Ready to read buffer
4 BWR	Buffer write ready This status field is set if the Buffer Write Enable field, in the Present State register, changes from 0 to 1. See the Buffer Write Enable field in the Present State register for additional information. 0b - Not ready to write buffer 1b - Ready to write buffer
3 DINT	DMA interrupt Occurs only when the internal DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this field does not be set. Instead, the DMAE field is set. Either Simple DMA or ADMA finishes data transferring, this field is set. 0b - No DMA interrupt 1b - DMA interrupt is generated.
2 BGE	Block gap event If the Stop At Block Gap Request field in the Protocol Control register is set, this field is set when a read or write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this field is not set to 1. In the case of a Read Transaction: This field is set at the falling edge of the DATA Line Active Status (When the transaction is stopped at SD bus timing). The Read Wait must be supported to use this function. In the case of Write Transaction: This field is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). 0b - No block gap event 1b - Transaction stopped at block gap
1 TC	Transfer complete This field is set when a read or write transfer is completed.

Table continues on the next page...

Field	Function
	In the case of a Read Transaction: This field is set at the falling edge of the Read Transfer Active Status. There are two cases in which this interrupt is generated. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the host system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request field in the Protocol Control register (after valid data has been read to the host system). In the case of a Write Transaction: This field is set at the falling edge of the DATA Line Active Status. There are two cases in which this interrupt is generated. The first is when the last data is written to the SD card as specified by the data length and the busy signal is released. The second is when data transfers are stopped at the block gap, by setting the Stop At Block Gap Request field in the Protocol Control register, and the data transfers are completed. (after valid data is written to the SD card and the busy signal released). In the case of a command with busy, this field is set when busy is deasserted. This field is not asserted in tuning process. 0b - Transfer does not complete 1b - Transfer complete
0 CC	Command complete This field is set when you receive the end field of the command response (except auto CMD12). See the Command Inhibit (CMD) in the Present State register. This field is not asserted in tuning process. 0b - Command not complete 1b - Command complete

23.5.1.15 Interrupt Status Enable (INT_STATUS_EN)

23.5.1.15.1 Offset

Register	Offset
INT_STATUS_EN	34h

23.5.1.15.2 Function

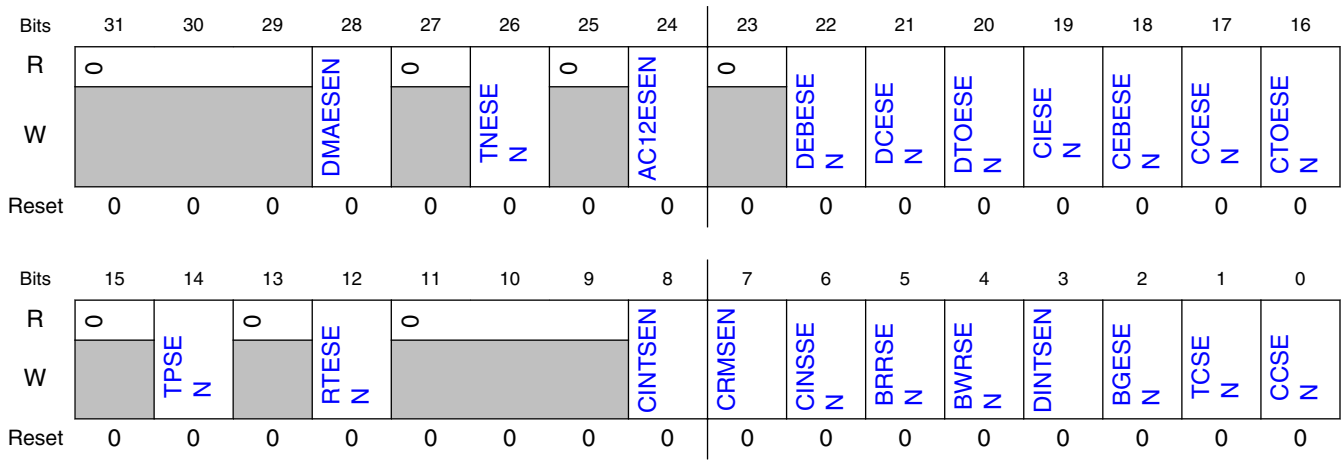
Setting the fields in this register to 1 enables the corresponding Interrupt Status to be set by the specified event. If any field is cleared, the corresponding Interrupt Status field is also cleared (that is, when the field in this register is cleared, the corresponding field in Interrupt Status register is always 0).

- Depending on IABG field setting, uSDHC might be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There

are some delays on the Card Interrupt, asserted from the card, to the time the host system is informed.

- To detect a CMD line conflict, the host driver must set both Command Timeout Error Status Enable and Command CRC Error Status Enable to 1.

23.5.1.15.3 Diagram



23.5.1.15.4 Fields

Field	Function
31-29 —	Reserved
28 DMAESEN	DMA error status enable 0b - Masked 1b - Enabled
27 —	Reserved
26 TNESEN	Tuning error status enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12ESEN	Auto CMD12 error status enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBESEN	Data end bit error status enable 0b - Masked 1b - Enabled

Table continues on the next page...

Field	Function
21 DCESEN	Data CRC error status enable 0b - Masked 1b - Enabled
20 DTOESEN	Data timeout error status enable 0b - Masked 1b - Enabled
19 CIESEN	Command index error status enable 0b - Masked 1b - Enabled
18 CEBESEN	Command end bit error status enable 0b - Masked 1b - Enabled
17 CCESEN	Command CRC error status enable 0b - Masked 1b - Enabled
16 CTOESEN	Command timeout error status enable 0b - Masked 1b - Enabled
15 —	Reserved
14 TPSEN	Tuning pass status enable 0b - Masked 1b - Enabled
13 —	Reserved
12 RTESEN	Re-tuning event status enable 0b - Masked 1b - Enabled
11-9 —	Reserved
8 CINTSEN	Card interrupt status enable If this field is set to 0, uSDHC clears the interrupt request to the system. The Card Interrupt detection is stopped when this field is cleared and restarted when this field is set to 1. The host driver should clear the Card Interrupt Status Enable before servicing the Card Interrupt and should set this field again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0b - Masked 1b - Enabled
7 CRMSEN	Card removal status enable 0b - Masked 1b - Enabled
6 CINSSEN	Card insertion status enable 0b - Masked 1b - Enabled
5 BRRSEN	Buffer read ready status enable 0b - Masked 1b - Enabled
4 BWRSEN	Buffer write ready status enable 0b - Masked

Table continues on the next page...

Field	Function
	1b - Enabled
3 DINTSEN	DMA interrupt status enable 0b - Masked 1b - Enabled
2 BGESEN	Block gap event status enable 0b - Masked 1b - Enabled
1 TCSEN	Transfer complete status enable 0b - Masked 1b - Enabled
0 CCSEN	Command complete status enable 0b - Masked 1b - Enabled

23.5.1.16 Interrupt Signal Enable (INT_SIGNAL_EN)

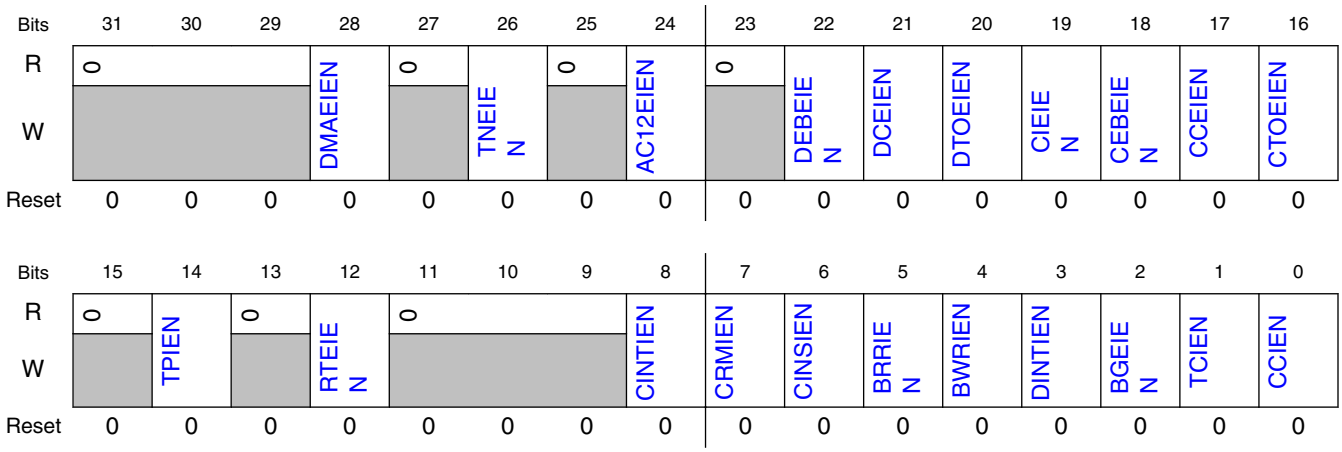
23.5.1.16.1 Offset

Register	Offset
INT_SIGNAL_EN	38h

23.5.1.16.2 Function

This register is used to select which interrupt status is indicated to the host system as the interrupt. These status fields all share the same interrupt line. Setting any of these fields to 1 enables interrupt generation. The corresponding Status register field generates an interrupt when the corresponding interrupt signal enable field is set.

23.5.1.16.3 Diagram



23.5.1.16.4 Fields

Field	Function
31-29 —	Reserved
28 DMAEIEN	DMA error interrupt enable 0b - Masked 1b - Enable
27 —	Reserved
26 TNEIEN	Tuning error interrupt enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12EIEN	Auto CMD12 error interrupt enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBEIEN	Data end bit error interrupt enable 0b - Masked 1b - Enabled
21 DCEIEN	Data CRC error interrupt enable 0b - Masked 1b - Enabled
20 DTOEIEN	Data timeout error interrupt enable 0b - Masked 1b - Enabled
19	Command index error interrupt enable 0b - Masked

Table continues on the next page...

Field	Function
CIEIEN	1b - Enabled
18 CEBEIEN	Command end bit error interrupt enable 0b - Masked 1b - Enabled
17 CCEIEN	Command CRC error interrupt enable 0b - Masked 1b - Enabled
16 CTOEIEN	Command timeout error interrupt enable 0b - Masked 1b - Enabled
15 —	Reserved
14 TPIEN	Tuning Pass interrupt enable 0b - Masked 1b - Enabled
13 —	Reserved
12 RTEIEN	Re-tuning event interrupt enable 0b - Masked 1b - Enabled
11-9 —	Reserved
8 CINTIEN	Card interrupt enable 0b - Masked 1b - Enabled
7 CRMIEN	Card removal interrupt enable 0b - Masked 1b - Enabled
6 CINSIEN	Card insertion interrupt enable 0b - Masked 1b - Enabled
5 BRRIEN	Buffer read ready interrupt enable 0b - Masked 1b - Enabled
4 BWRIEN	Buffer write ready interrupt enable 0b - Masked 1b - Enabled
3 DINTIEN	DMA interrupt enable 0b - Masked 1b - Enabled
2 BGEIEN	Block gap event interrupt enable 0b - Masked 1b - Enabled
1 TCIEN	Transfer complete interrupt enable 0b - Masked 1b - Enabled
0 CCIEN	Command complete interrupt enable 0b - Masked

Field	Function
	1b - Enabled

23.5.1.17 Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)

23.5.1.17.1 Offset

Register	Offset
AUTOCMD12_ERR_STATUS	3Ch

23.5.1.17.2 Function

When the Auto CMD12 Error Status field in the Status register is set, the host driver checks this register to identify what kind of error the Auto CMD12 / CMD 23 indicated. Auto CMD23 errors are indicated in field 04-01. This register is valid only when the Auto CMD12 Error status field is set.

The table below shows the relationship between the Auto CMGD12 CRC Error and the Auto CMD12 Command Timeout Error.

Table 23-11. Relationship between command CRC error and command timeout error for auto CMD12

Auto CMD12 CRC error	Auto CMD12 timeout error	Type of error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Changes in Auto CMD12 Error Status register can be classified in three scenarios:

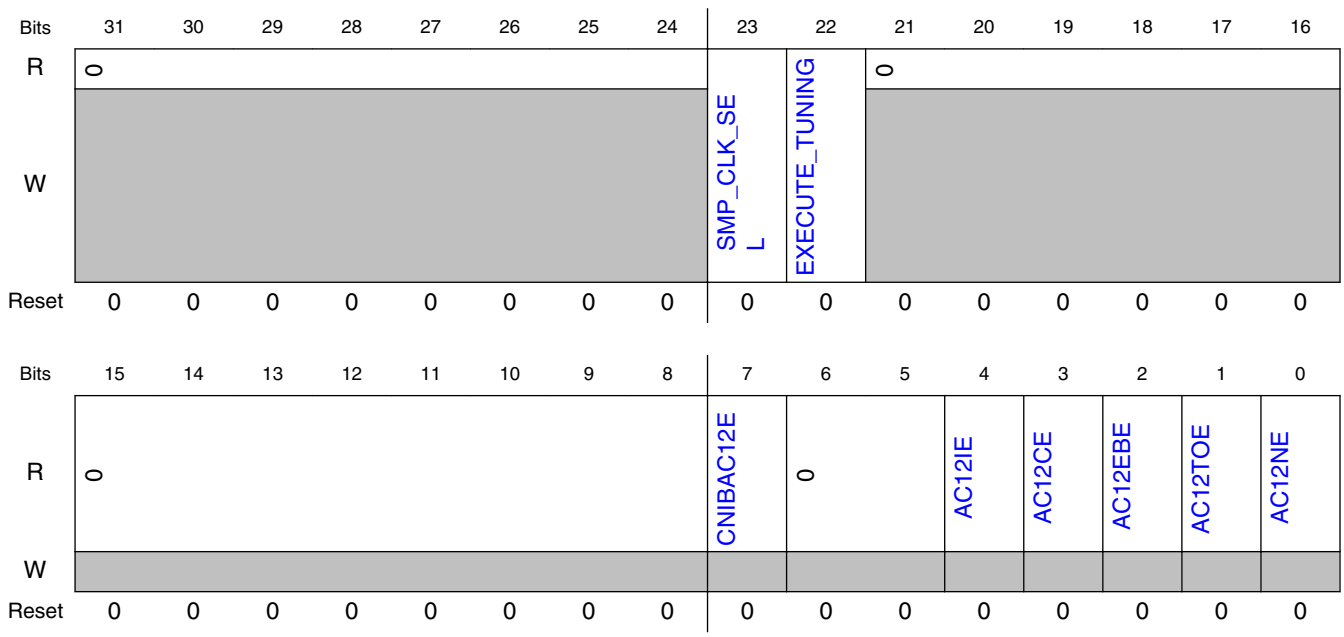
- When uSDHC is going to issue an Auto CMD12
  - Set field 0 to 1 if the Auto CMD12 cannot be issued due to an error in the previous command
  - Set field 0 to 0 if the Auto CMD12 is issued
- At the end field of an Auto CMD12 response
  - Check errors correspond to fields 1-4

- Set fields 1-4 corresponding to detected errors
- Clear fields 1-4 corresponding to detected errors

- Before reading the Auto CMD12 Error Status field 7
  - Set field 7 to 1 if there is a command that can't be issued
  - Clear field 7 if there is no command to issue

The timing for generating the Auto CMD12 Error and writing to the Command register are asynchronous. After that, field 7 is sampled when the driver is not writing to the Command register. So, it is suggested to read this register only when the AC12E field in Interrupt Status register is set. An Auto CMD12 Error Interrupt is generated when one of the error fields (0-4) is set to 1. The Command Not Issued By Auto CMD12 Error does not generate an interrupt.

23.5.1.17.3 Diagram



23.5.1.17.4 Fields

Field	Function
31-24 —	Reserved
23 SMP_CLK_SEL	Sample clock select When std_tuning_en field is set, this field is used to select sampling clock to receive CMD and DATA. Otherwise, this field is reserved. This field is set by ty tuning procedure and valid after the completion of tuning(When Execute Tuning is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning is failed. Writing 1 to this field is meaningless and ignored. A tuning circuit is

Table continues on the next page...

Field	Function
	reset by writing to 0. This field can be cleared with setting Execute Tuning. Once the tuning circuit is reset, it takes time to complete tuning sequence. Therefore, host driver should keep this field to 1 to perform re-tuning sequence to complete re-tuning sequence in a short time. Change of this field is not allowed while the Host controller us receiving response or a read data block. 0b - Fixed clock is used to sample data 1b - Tuned clock is used to sample data
22 EXECUTE_TUNING	Execute tuning When std_tuning_en field is set, this field is used to start tuning procedure. Otherwise, this field is reserved. This field is set to start tuning procedure and automatically cleared when runing procedure is completed. The result of tuning is indicated to sam_clk_sel field. Tuning procedure is aborted by writing 0.
21-8 —	Reserved
7 CNIBAC12E	Command not issued by Auto CMD12 error Setting this field to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (D04-D01) in this register. 0b - No error 1b - Not issued
6-5 —	Reserved
4 AC12IE	Auto CMD12 / 23 index error Occurs if the command index error occurs in response to a command. 0b - No error 1b - Error, the CMD index in response is not CMD12/23
3 AC12CE	Auto CMD12 / 23 CRC error Occurs when detecting a CRC error in the command response. 0b - No CRC error 1b - CRC error met in Auto CMD12/23 response
2 AC12EBE	Auto CMD12 / 23 end bit error Occurs when detecting that the end field of command response is 0 which should be 1. 0b - No error 1b - End bit error generated
1 AC12TOE	Auto CMD12 / 23 timeout error Occurs if no response is returned within 64 SDCLK cycles from the end field of the command. If this field is set to1, the other error status fields (2-4) have no meaning. 0b - No error 1b - Time out
0 AC12NE	Auto CMD12 not executed If memory multiple block data transfer is not started, due to a command error, this field is not set because it is not necessary to issue an Auto CMD12. Setting this field to 1 means uSDHC cannot issue the Auto CMD12 to stop a memory multiple block data transfer due to some error. If this field is set to 1, other error status fields (1-4) have no meaning. 0b - Executed 1b - Not executed

23.5.1.18 Host Controller Capabilities (HOST_CTRL_CAP)

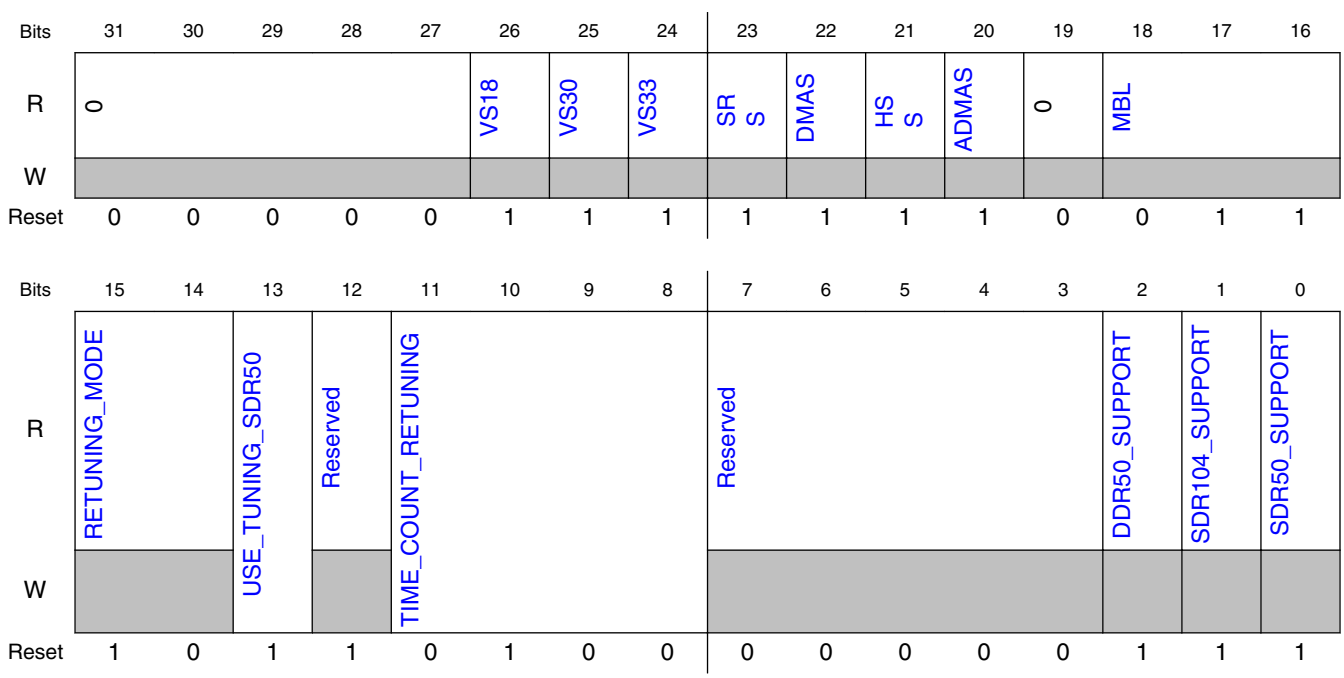
23.5.1.18.1 Offset

Register	Offset
HOST_CTRL_CAP	40h

23.5.1.18.2 Function

This register provides the host driver with information specific to uSDHC implementation. The value in this register is the power-on-reset value and does not change with a software reset.

23.5.1.18.3 Diagram



23.5.1.18.4 Fields

Field	Function
31-27	Reserved

Table continues on the next page...

Field	Function
—	
26 VS18	Voltage support 1.8 V This field depends on the host system ability. 0b - 1.8 V not supported 1b - 1.8 V supported
25 VS30	Voltage support 3.0 V This field depends on the host system ability. 0b - 3.0 V not supported 1b - 3.0 V supported
24 VS33	Voltage support 3.3 V This field depends on the host system ability. 0b - 3.3 V not supported 1b - 3.3 V supported
23 SRS	Suspend / resume support This field indicates whether uSDHC supports Suspend / Resume functionality. If this field is 0, the Suspend and Resume mechanism, as well as the read wait, are not supported, and the host driver does not issue either Suspend or Resume commands. 0b - Not supported 1b - Supported
22 DMAS	DMA support This field indicates whether uSDHC is capable of using the internal DMA to transfer data between system memory and the data buffer directly. 0b - DMA not supported 1b - DMA supported
21 HSS	High speed support This field indicates whether uSDHC supports High Speed mode and the host system can supply a SD clock frequency from 25 MHz to 50 MHz. 0b - High speed not supported 1b - High speed supported
20 ADMAS	ADMA support This field indicates whether uSDHC supports the ADMA feature. 0b - Advanced DMA not supported 1b - Advanced DMA supported
19 —	Reserved
18-16 MBL	Max block length This value indicates the maximum block size that the host driver can read and write to the buffer in uSDHC. The buffer transfers block size without wait cycles. 000b - 512 bytes 001b - 1024 bytes 010b - 2048 bytes 011b - 4096 bytes
15-14	Retuning Mode This field selects retuning method.

Table continues on the next page...

Field	Function
RETUNING_MODE	00b - Mode 1 01b - Mode 2 10b - Mode 3 11b - Reserved
13 USE_TUNING_SDR50	Use Tuning for SDR50 This field is set to 1. Host controller requires tuning to operate SDR50. 0b - SDR does not require tuning. 1b - SDR50 requires tuning.
12 —	Reserved
11-8 TIME_COUNT_RETUNING	Time counter for retuning This field indicates an initial value of the Retuning Timer for Re-Tuning Mode1 and 3.Setting to 0 disables Retuning Timer.
7-3 —	Reserved
2 DDR50_SUPPORT	DDR50 support This field indicates support of DDR50 mode.
1 SDR104_SUPPORT	SDR104 support This field indicates support of SDR104 mode.
0 SDR50_SUPPORT	SDR50 support This field indicates support of SDR50 mode.

23.5.1.19 Watermark Level (WTMK_LVL)

23.5.1.19.1 Offset

Register	Offset
WTMK_LVL	44h

23.5.1.19.2 Function

Both write and read watermark levels (FIFO threshold) are configurable. Their value can range from 1 to 128 words. Both write and read burst lengths are also Configurable. There value can range from 1 to 31 words.

23.5.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WR_BRST_LEN					WR_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			RD_BRST_LEN					RD_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

23.5.1.19.4 Fields

Field	Function
31-29 —	Reserved
28-24 WR_BRST_LEN	Write burst length due to system restriction, the actual burst length might not exceed 16 The number of words uSDHC writes in a single burst. The write burst length must be less than or equal to the write watermark level, and all bursts within a watermark level transfer is in back-to-back mode. On reset, this field is 8. Writing 0 to this field results in '01000' (that is, it is not able to clear this field).
23-16 WR_WML	Write watermark level The number of words used as the watermark level (FIFO threshold) in a DMA write operation. Also, the number of words as a sequence of write bursts in back-to-back mode. The maximum legal value for the write watermark level is 128.
15-13 —	Reserved
12-8 RD_BRST_LEN	Read burst length due to system restriction, the actual burst length might not exceed 16 The number of words uSDHC reads in a single burst. The read burst length must be less than or equal to the read watermark level, and all bursts within a watermark level transfer is in back-to-back mode. On reset, this field is 8. Writing 0 to this field results in '01000' (that is, it is not able to clear this field).
7-0 RD_WML	Read watermark level The number of words used as the watermark level (FIFO threshold) in a DMA read operation. Also, the number of words as a sequence of read bursts in back-to-back mode. The maximum legal value for the read water mark level is 128.

23.5.1.20 Mixer Control (MIX_CTRL)

23.5.1.20.1 Offset

Register	Offset
MIX_CTRL	48h

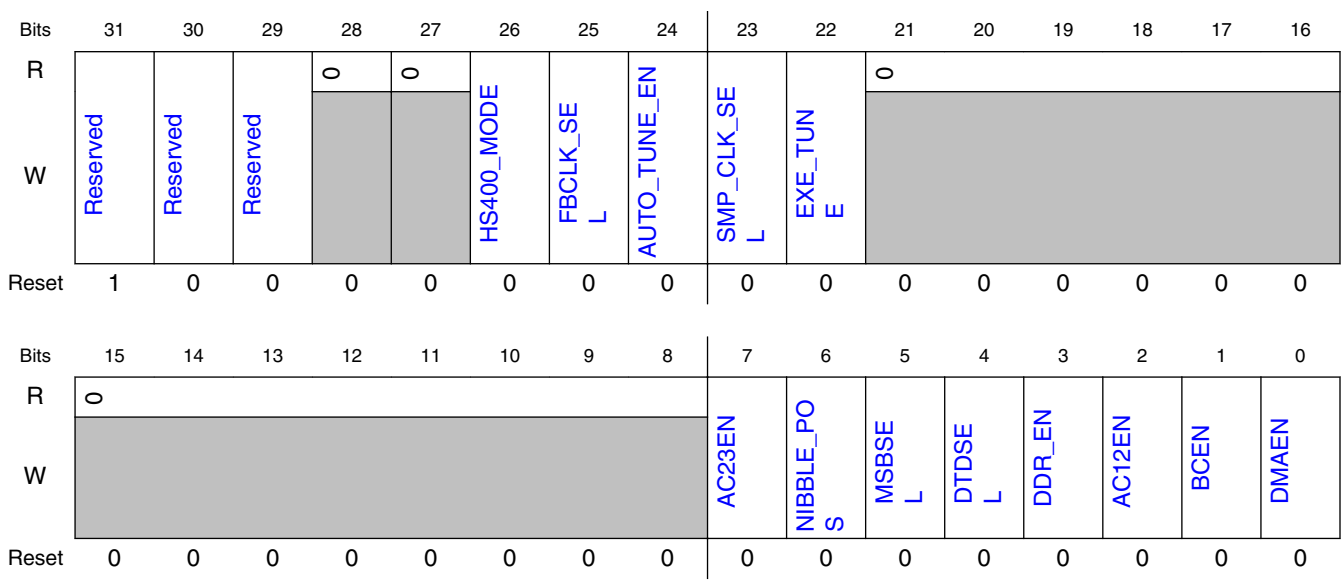
23.5.1.20.2 Function

This register is used to DMA and data transfer. To prevent data loss, The software should check if data transfer is active before writing this register. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

Table 23-12. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

23.5.1.20.3 Diagram



23.5.1.20.4 Fields

Field	Function
31	Reserved

Table continues on the next page...

Field	Function
—	Always write as 1.
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 0.
28 —	Reserved
27 —	Reserved
26 HS400_MODE	Enable HS400 mode HS400 Enable
25 FBCLK_SEL	Feedback clock source selection (Only used for SD3.0, SDR104 mode and EMMC HS200 mode) 0b - Feedback clock comes from the loopback CLK 1b - Feedback clock comes from the ipp_card_clk_out
24 AUTO_TUNE_EN	Auto tuning enable (Only used for SD3.0, SDR104 mode and and EMMC HS200 mode) 0b - Disable auto tuning 1b - Enable auto tuning
23 SMP_CLK_SEL	Clock selection When STD_TUNING_EN is 0, this field is used to select Tuned clock or Fixed clock to sample data / cmd (Only used for SD3.0, SDR104 mode and EMMC HS200 mode) 0b - Fixed clock is used to sample data / cmd 1b - Tuned clock is used to sample data / cmd
22 EXE_TUNE	Execute tuning: (Only used for SD3.0, SDR104 mode and EMMC HS200 mode) When STD_TUNING_EN is 0, this field is set to 1 to indicate the host driver is starting tuning procedure. Tuning procedure is aborted by writing 0. 0b - Not tuned or tuning completed 1b - Execute tuning
21-8 —	Reserved
7 AC23EN	Auto CMD23 enable When this field is set to 1, the host controller issues a CMD23 automatically before issuing a command specified in the Command Register.
6 NIBBLE_POS	Nibble position indication In DDR 4-bit mode nibble position indication. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'.
5 MSBSEL	Multi / Single block select This field enables multiple block DATA line data transfers. For any other commands, this field can be set to 0. If this field is 0, it is not necessary to set the Block Count register. See Command Transfer Type (CMD_XFR_TYP). 0b - Single block 1b - Multiple blocks
4 DTDSEL	Data transfer direction select

Table continues on the next page...

Field	Function
	This field defines the direction of DATA line data transfers. The field is set to 1 by the host driver to transfer data from the SD card to uSDHC and is set to 0 for all other commands. 0b - Write (Host to card) 1b - Read (Card to host)
3 DDR_EN	Dual data rate mode selection
2 AC12EN	Auto CMD12 enable Multiple block transfers for memory require a CMD12 to stop the transaction. When this field is set to 1, uSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver is not set this field to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, uSDHC ignores this field no matter it is set or not. 0b - Disable 1b - Enable
1 BCEN	Block count enable This field is used to enable the Block Count register, which is only relevant for multiple block transfers. When this field is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer. 0b - Disable 1b - Enable
0 DMAEN	DMA enable This field enables DMA functionality. If this field is set to 1, a DMA operation begins when the host driver sets the DPSEL field of this register. Whether the simple DMA or the advanced DMA is active depends on the DMA Select field of the Protocol Control register. 0b - Disable 1b - Enable

23.5.1.21 Force Event (FORCE_EVENT)

23.5.1.21.1 Offset

Register	Offset
FORCE_EVENT	50h

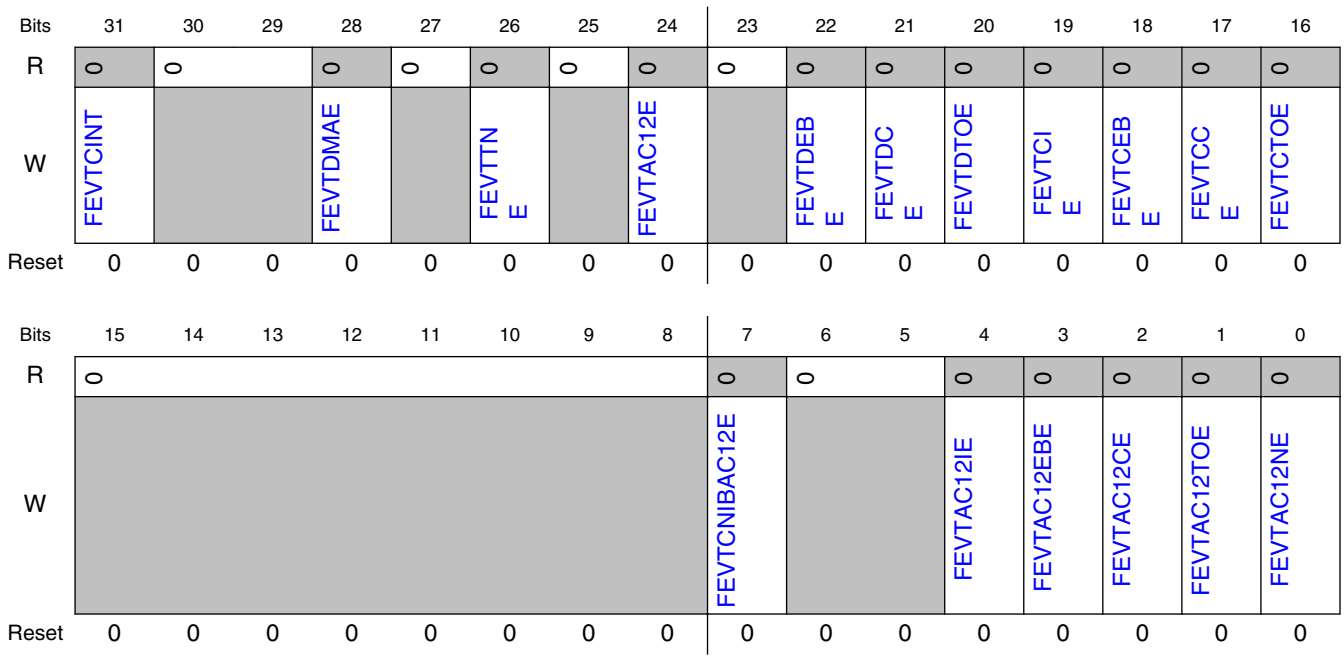
23.5.1.21.2 Function

The Force Event Register is not a physically implemented register. Rather, it is an address at which the Interrupt Status register can be written if the corresponding field of the Interrupt Status Enable Register is set. This register is a write only register and writing 0 to it has no effect. Writing 1 to this register actually sets the corresponding field

of Interrupt Status register. A read from this register always results in 0's. to change the corresponding status fields in the Interrupt Status register, make sure to set IPGEN field in System Control Register so that peripheral clock is always active.

Forcing a card interrupt generates a short pulse on the DATA1 line, and the driver might treat this interrupt as a normal interrupt. The interrupt service routine might skip polling the card interrupt factor as the interrupt is self cleared.

23.5.1.21.3 Diagram



23.5.1.21.4 Fields

Field	Function
31 FEVTCINT	Force event card interrupt Writing 1 to this field generates a short low-level pulse on the internal DATA1 line, as if a self clearing interrupt was received from the external card. If enabled, the CINT field is set and the interrupt service routine might treat this interrupt as a normal interrupt from the external card.
30-29 —	Reserved
28 FEVTDMAE	Force event DMA error Forces the DMAE field of Interrupt Status register to be set.
27 —	Reserved
26	Force tuning error

Table continues on the next page...

Field	Function
FEVTTNE	Forces the TNE field of Interrupt Status register to be set.
25 —	Reserved
24 FEVTAC12E	Force event Auto Command 12 error Forces the AC12E field of Interrupt Status register to be set.
23 —	Reserved
22 FEVTDEBE	Force event data end bit error Forces the DEBE field of Interrupt Status register to be set.
21 FEVTDCE	Force event data CRC error Forces the DCE field of Interrupt Status register to be set.
20 FEVTDTOE	Force event data time out error Force the DTOE field of Interrupt Status register to be set.
19 FEVTCIE	Force event command index error Forces the CCE field of Interrupt Status register to be set.
18 FEVTCEBE	Force event command end bit error Forces the CEBE field of Interrupt Status register to be set.
17 FEVTCCE	Force event command CRC error Forces the CCE field of Interrupt Status register to be set.
16 FEVTCTOE	Force event command time out error Forces the CTOE field of Interrupt Status register to be set.
15-8 —	Reserved
7 FEVTCNIBAC12E	Force event command not executed by Auto Command 12 error Forces the CNIBAC12E field in the Auto Command12 Error Status register to be set.
6-5 —	Reserved
4 FEVTAC12IE	Force event Auto Command 12 index error Forces the AC12IE field in the Auto Command12 Error Status register to be set.
3 FEVTAC12EBE	Force event Auto Command 12 end bit error Forces the AC12EBE field in the Auto Command12 Error Status register to be set.
2 FEVTAC12CE	Force event auto command 12 CRC error Forces the AC12CE field in the Auto Command12 Error Status register to be set.
1 FEVTAC12TOE	Force event auto command 12 time out error Forces the AC12TOE field in the Auto Command12 Error Status register to be set.
0 FEVTAC12NE	Force event auto command 12 not executed Forces the AC12NE field in the Auto Command12 Error Status register to be set.

23.5.1.22 ADMA Error Status (ADMA_ERR_STATUS)

23.5.1.22.1 Offset

Register	Offset
ADMA_ERR_STATUS	54h

23.5.1.22.2 Function

When an ADMA Error Interrupt has occurred, the ADMA Error States field in this register holds the ADMA state and the ADMA System Address register holds the address around the error descriptor.

For recovering from this error, the host driver requires the ADMA state to identify the error descriptor address as follows:

- ST_STOP: Previous location set in the ADMA System Address register is the error descriptor address.
- ST_FDS: Current location set in the ADMA System Address register is the error descriptor address.
- ST_CADR: This state is never set because it only increments the descriptor pointer and does not generate an ADMA error.
- ST_TFR: Previous location set in the ADMA System Address register is the error descriptor address.

In case of a write operation, the host driver should use the ACMD22 to get the number of the written block, rather than using this information, because unwritten data might exist in the host controller.

The host controller generates the ADMA Error Interrupt when it detects invalid descriptor data (Valid=0) in the ST_FDS state. The host driver can distinguish this error by reading the Valid field of the error descriptor.

Table 23-13. ADMA error state coding

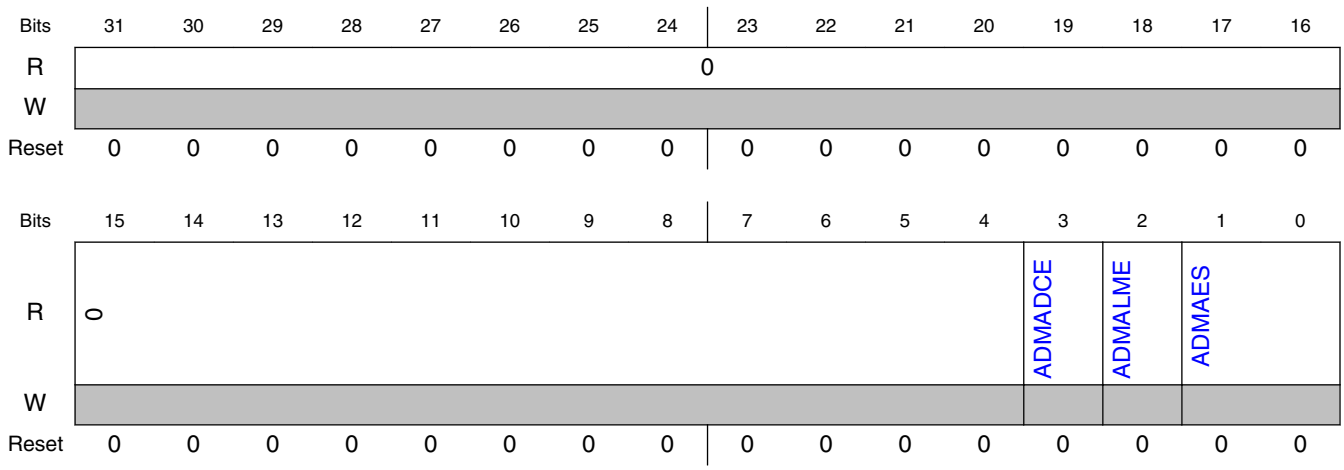
D01-D00	ADMA error state (when error has occurred)	Contents of ADMA System Address register
00	ST_STOP (Stop DMA)	Holds the address of the next executable descriptor command
01	ST_FDS (Fetch descriptor)	Holds the valid descriptor address
10	ST_CADR (Change address)	No ADMA error is generated

Table continues on the next page...

Table 23-13. ADMA error state coding (continued)

D01-D00	ADMA error state (when error has occurred)	Contents of ADMA System Address register
11	ST_TFR (Transfer data)	Holds the address of the next executable descriptor command

23.5.1.22.3 Diagram



23.5.1.22.4 Fields

Field	Function
31-4 —	Reserved
3 ADMADCE	ADMA descriptor error This error occurs when invalid descriptor fetched by ADMA. 0b - No error 1b - Error
2 ADMALME	ADMA length mismatch error This error occurs in the following two cases: • While the block count enable is being set, the total data length specified by the descriptor table is different from that specified by the block count and block length. • Total data length cannot be divided by the block length. 0b - No error 1b - Error
1-0 ADMAES	ADMA error state (when ADMA error is occurred) This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. See ADMA Error Status (ADMA_ERR_STATUS) for more details.

23.5.1.23 ADMA System Address (ADMA_SYS_ADDR)

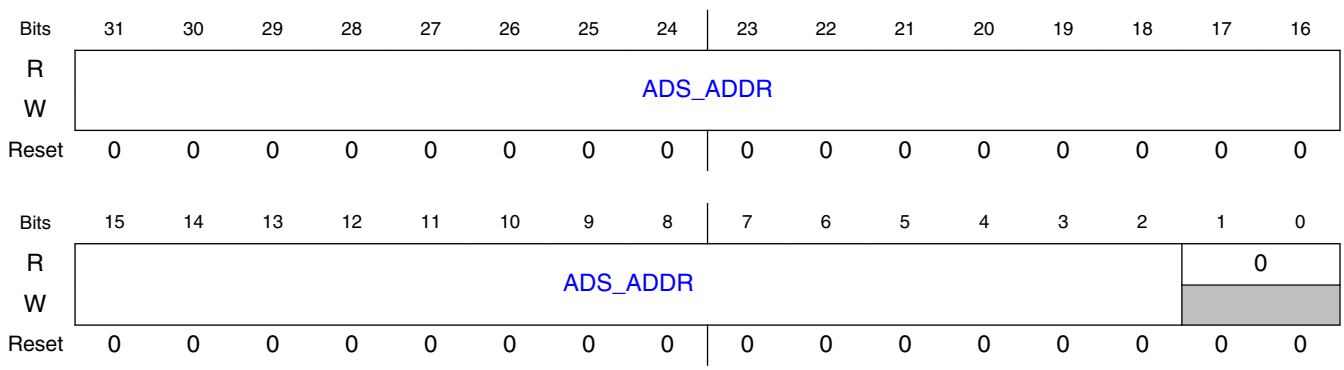
23.5.1.23.1 Offset

Register	Offset
ADMA_SYS_ADDR	58h

23.5.1.23.2 Function

This register contains the physical system memory address used for ADMA transfers.

23.5.1.23.3 Diagram



23.5.1.23.4 Fields

Field	Function
31-2 ADS_ADDR	ADMA system address This register holds the word address of the executing command in the Descriptor table. At the start of ADMA, the host driver sets the start address of the Descriptor table. The ADMA engine increments this register address whenever fetching a Descriptor command. When the ADMA is stopped at the Block Gap, this register indicates the address of the next executable Descriptor command. When the ADMA Error Interrupt is generated, this register holds the valid Descriptor address depending on the ADMA state. The lower 2 bits of this register is tied to '0' so the ADMA address is always word aligned. Because this register supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC field is set. Such restriction is also listed in Software restrictions.
1-0 —	Reserved

23.5.1.24 DLL (Delay Line) Control (DLL_CTRL)

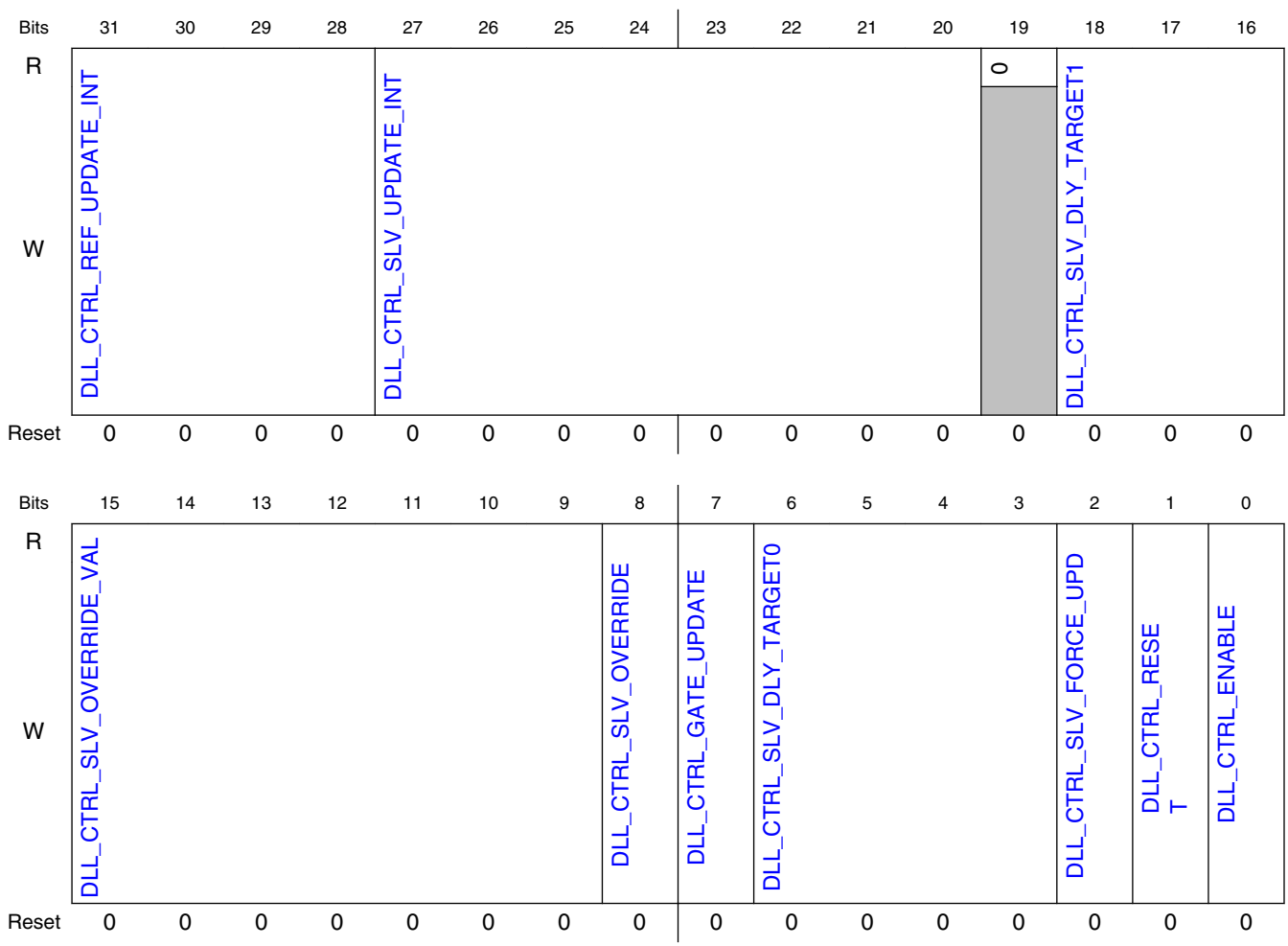
23.5.1.24.1 Offset

Register	Offset
DLL_CTRL	60h

23.5.1.24.2 Function

This register contains control fields for DLL.

23.5.1.24.3 Diagram



23.5.1.24.4 Fields

Field	Function
31-28 DLL_CTRL_REF_UPDATE_INT	DLL control loop update interval The interval cycle is (2 + REF_UPDATE_INT) * REF_CLOCK. By default, the DLL control loop updates every two REF_CLOCK cycles. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20 DLL_CTRL_SLV_UPDATE_INT	Slave delay line update interval If default 0 is used, it means 256 cycles of REF_CLOCK. A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19 —	Reserved
18-16 DLL_CTRL_SLV_DLY_TARGET1	DLL slave delay target1 See DLL_CTRL_SLV_DLY_TARGET0 below.
15-9 DLL_CTRL_SLV_OVERRIDE_VAL	DLL slave override val When SLV_OVERRIDE = 1, this field is used to select 1 of 128 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 128.
8 DLL_CTRL_SLV_OVERRIDE	DLL slave override Set this field to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0
7 DLL_CTRL_GATE_UPDATE	DLL gate update Set this field to 1 to prevent the DLL from updating (because when clock_in exists, glitches might appear during DLL updates). This field might be used by software if such a condition occurs. Clear the bit to 0 to allow the DLL to update automatically.
6-3 DLL_CTRL_SLV_DLY_TARGET0	DLL slave delay target0 The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an ref_clock half-period. The delay is (({DLL_CTRL_SLV_DLY_TARGET1,DLL_CTRL_SLV_DLY_TARGET0} +1)* REF_CLOCK / 2) / 16 So the input read-clock can be delayed relative input data from (REF_CLOCK / 2) / 16 to REF_CLOCK * 4.
2 DLL_CTRL_SLV_FORCE_UPD	DLL slave delay line Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line updates automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.
1 DLL_CTRL_RESET	DLL reset Setting this field to 1 force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, so to create a subsequent reset, RESET must be taken low and then asserted again.
0	DLL and delay chain

Field	Function
DLL_CTRL_ENABLE	Set this field to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE VAL, the DLL does not need to be enabled.

23.5.1.25 DLL Status (DLL_STATUS)

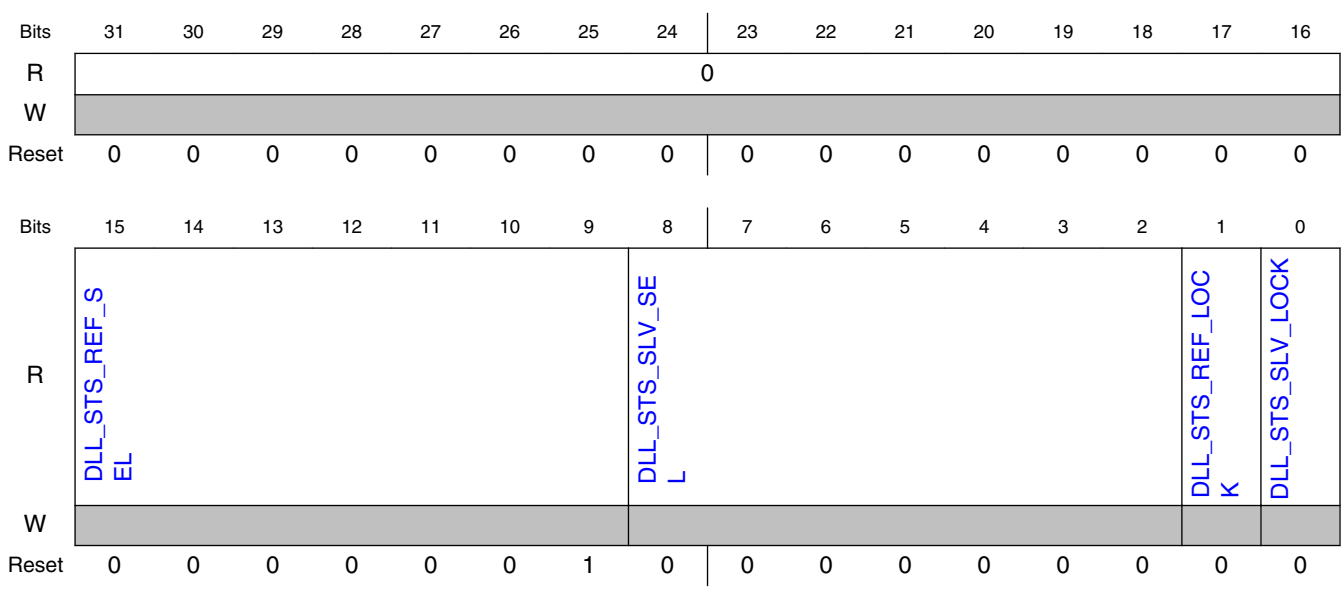
23.5.1.25.1 Offset

Register	Offset
DLL_STATUS	64h

23.5.1.25.2 Function

This register contains the DLL status information. All fields are read only and reads the same as the power-reset value.

23.5.1.25.3 Diagram



23.5.1.25.4 Fields

Field	Function
31-16 —	Reserved
15-9 DLL_STS_REF_SEL	Reference delay line select taps This is encoded by 7 fields for 127 taps.
8-2 DLL_STS_SLV_SEL	Slave delay line select status This is the instant value generated from reference chain. Because the reference chain can only be updated when REF_CLOCK is detected, this value should be the right value to be updated when the reference is locked.
1 DLL_STS_REF_LOCK	Reference DLL lock status This signifies that the DLL has detected and locked to a half-phase ref_clock shift, allowing the slave delay-line to perform programmed clock delays
0 DLL_STS_SLV_LOCK	Slave delay-line lock status This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value

23.5.1.26 CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)

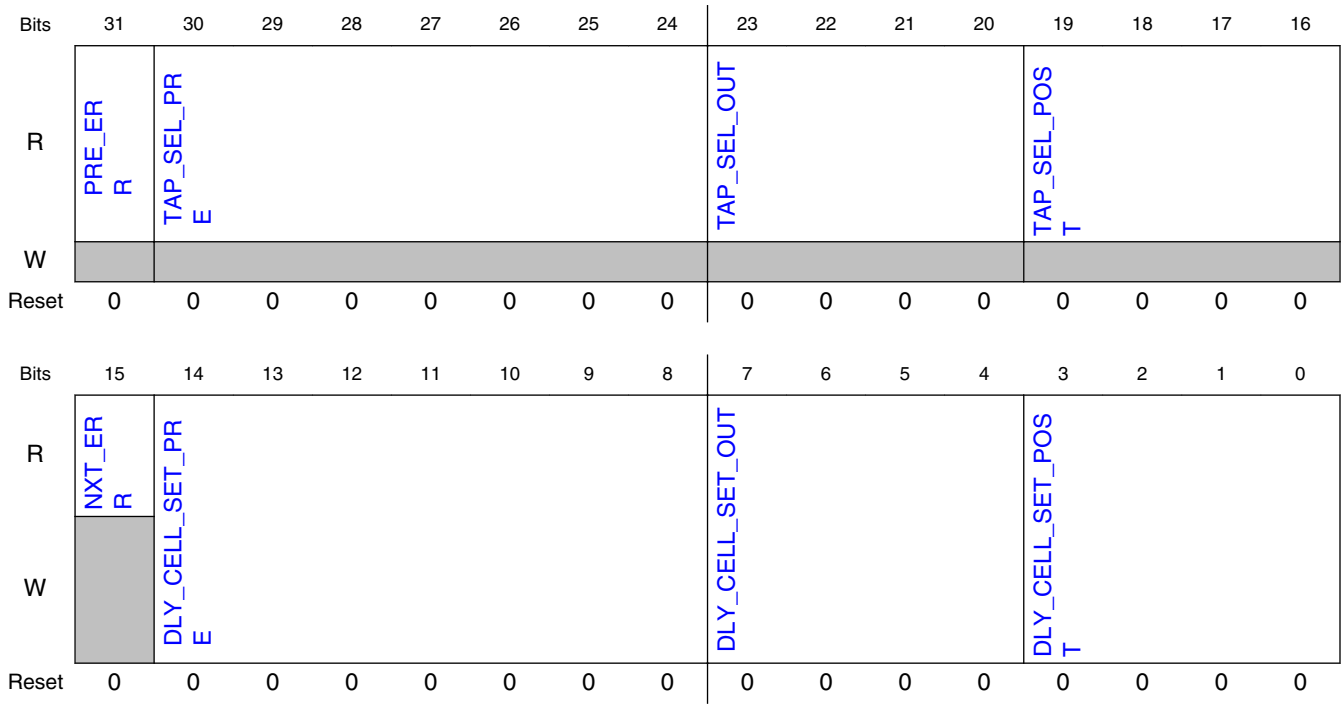
23.5.1.26.1 Offset

Register	Offset
CLK_TUNE_CTRL_STATUS	68h

23.5.1.26.2 Function

This register contains the Clock Tuning Control status information. All fields are read only and reads the same as the power-reset value. This register is added to support SD3.0 UHS-I SDR104 mode and EMMC HS200 mode.

23.5.1.26.3 Diagram



23.5.1.26.4 Fields

Field	Function
31 PRE_ERR	PRE error PRE error which means the number of delay cells added on the feedback clock is too small. It is valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
30-24 TAP_SEL_PRE	TAP_SEL_PRE Reflects the number of delay cells added on the feedback clock between the feedback clock and CLK_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is disabled, TAP_SEL_PRE is always equal to DLY_CELL_SET_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is enabled, TAP_SEL_PRE is updated automatically according to the status of the auto tuning circuit to adjust the sample clock phase.
23-20 TAP_SEL_OUT	Delay cells added on the feedback clock between CLK_PRE and CLK_OUT Reflects the number of delay cells added on the feedback clock between CLK_PRE and CLK_OUT.
19-16 TAP_SEL_POST	Delay cells added on the feedback clock between CLK_OUT and CLK_POST Reflects the number of delay cells added on the feedback clock between CLK_OUT and CLK_POST.
15 NXT_ERR	NXT error NXT error which means the number of delay cells added on the feedback clock is too large. It's valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
14-8	delay cells on the feedback clock between the feedback clock and CLK_PRE

Table continues on the next page...

Field	Function
DLY_CELL_SET_PRE	Set the number of delay cells on the feedback clock between the feedback clock and CLK_PRE.
7-4 DLY_CELL_SET_OUT	Delay cells on the feedback clock between CLK_PRE and CLK_OUT Set the number of delay cells on the feedback clock between CLK_PRE and CLK_OUT.
3-0 DLY_CELL_SET_POST	Delay cells on the feedback clock between CLK_OUT and CLK_POST Set the number of delay cells on the feedback clock between CLK_OUT and CLK_POST.

23.5.1.27 Strobe DLL control (STROBE_DLL_CTRL)

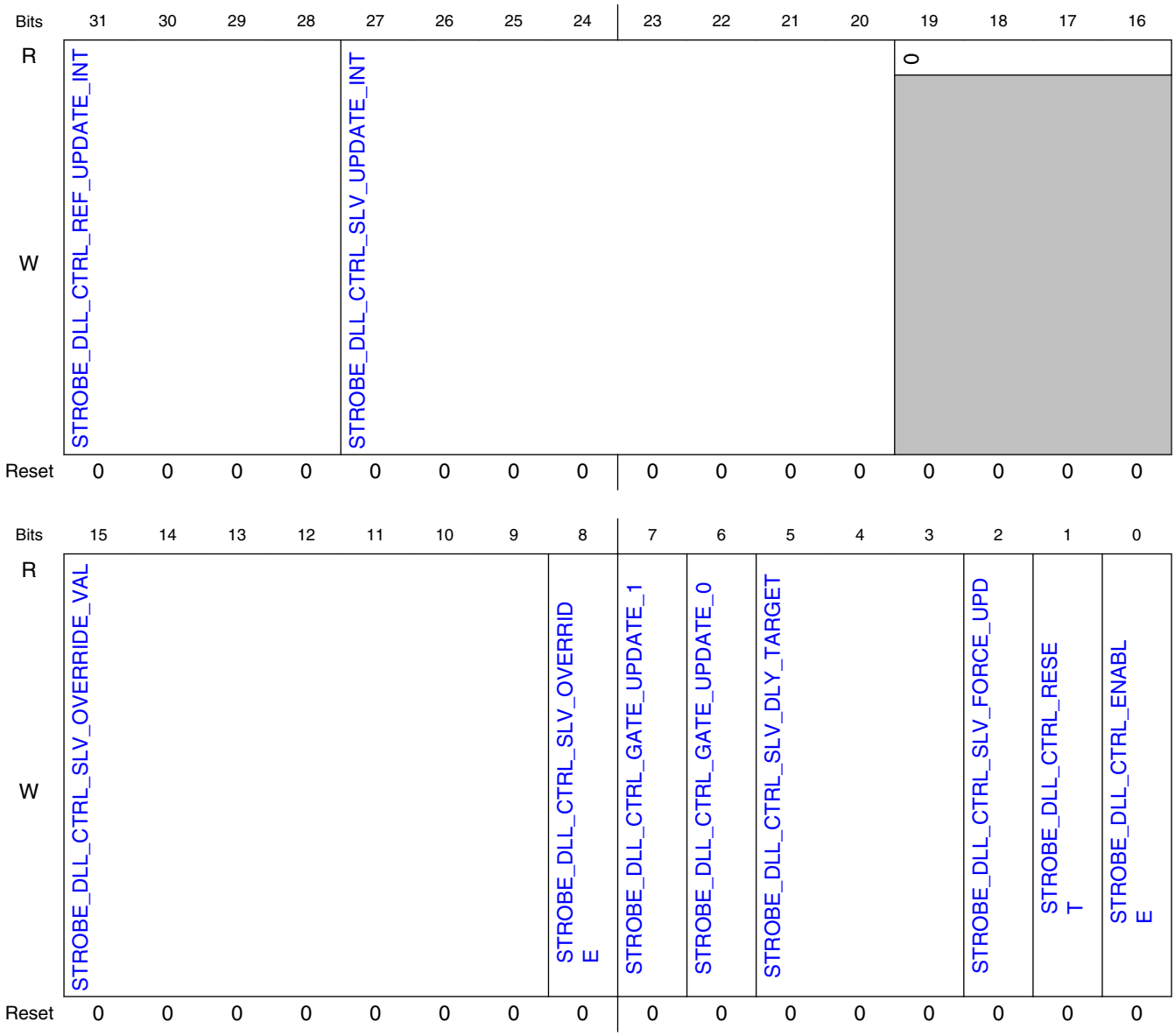
23.5.1.27.1 Offset

Register	Offset
STROBE_DLL_CTRL	70h

23.5.1.27.2 Function

This register contains the strobe DLL control information.

23.5.1.27.3 Diagram



23.5.1.27.4 Fields

Field	Function
31-28 STROBE_DLL_CTRL_REF_UPDATE_INT	Strobe DLL control reference update interval The interval cycle is (2 + STROBE_REF_UPDATE_INT) * STROBE_REF_CLOCK. By default, the DLL control loop updates every two STROBE_REF_CLOCK cycles. **NOTE:** Increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20	Strobe DLL control slave update interval

Table continues on the next page...

Field	Function
STROBE_DLL_CTRL_SLV_UPDATE_INT	Slave delay line update interval. If default 0 is used, it means 256 cycles of STROBE_REF_CLOCK. A value of 0x0F results in 15 cycles and so on. **NOTE:** software can always cause an update of the slave-delay line using the STROBE_SLV_FORCE_UPDATE register. The slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19-16 —	This field is reserved. This read-only field is reserved and always has the value 0.
15-9 STROBE_DLL_CTRL_SLV_OVERRIDE_VAL	Strobe DLL control slave Override value When STROBE_SLV_OVERRIDE = 1, this field is used to manually select one of 128 physical taps. A value of 0 selects tap 1, and a value of 0x7F selects tap 128.
8 STROBE_DLL_CTRL_SLV_OVERRIDE	Strobe DLL control slave override Set this field to 1 to Enable manual override for slave delay chain using STROBE_SLV_OVERRIDE_VAL; set this field to 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if STROBE_SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0.
7 STROBE_DLL_CTRL_GATE_UPDATE_1	Strobe DLL control gate update Set this field to 1 to prevent the DLL from updating (because when STROBE_CLOCK_IN exists, glitches might appear during DLL updates). This field can be used by software if such a condition occurs. Clear the field to 0 to allow the DLL to update automatically.
6 STROBE_DLL_CTRL_GATE_UPDATE_0	Strobe DLL control gate update Set this field to 1 to prevent the DLL from updating (because when STROBE_CLOCK_IN exists, glitches might appear during DLL updates). This field can be used by software if such a condition occurs. Clear the field to 0 to allow the DLL to update automatically.
5-3 STROBE_DLL_CTRL_SLV_DLY_TARGET	Strobe DLL Control Slave Delay Target The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an STROBE_REF_CLOCK half-period. The delay is ((STROBE_DLL_CTRL_SLV_DLY_TARGET + 1) * STROBE_REF_CLOCK / 2) / 16, So the input read-clock can be delayed relative input data from (STROBE_REF_CLOCK / 2) / 16 to (STROBE_REF_CLOCK *4)/16.
2 STROBE_DLL_CTRL_SLV_FORCE_UPD	Strobe DLL control slave force updated Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line should automatically update the STROBE_SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if STROBE_SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.
1 STROBE_DLL_CTRL_RESET	Strobe DLL control reset Setting this field to 1 to force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, to create a subsequent reset, RESET must be taken low and then asserted again.
0 STROBE_DLL_CTRL_ENABLE	Strobe DLL control enable Set this field to 1 to enable the DLL and delay chain; otherwise, set to 0 to bypasses DLL. **NOTE:** Using the slave delay line override feature with STROBE_SLV_OVERRIDE and STROBE_SLV_OVERRIDE VAL, the DLL does not need to be enabled.

23.5.1.28 Strobe DLL status (STROBE_DLL_STATUS)

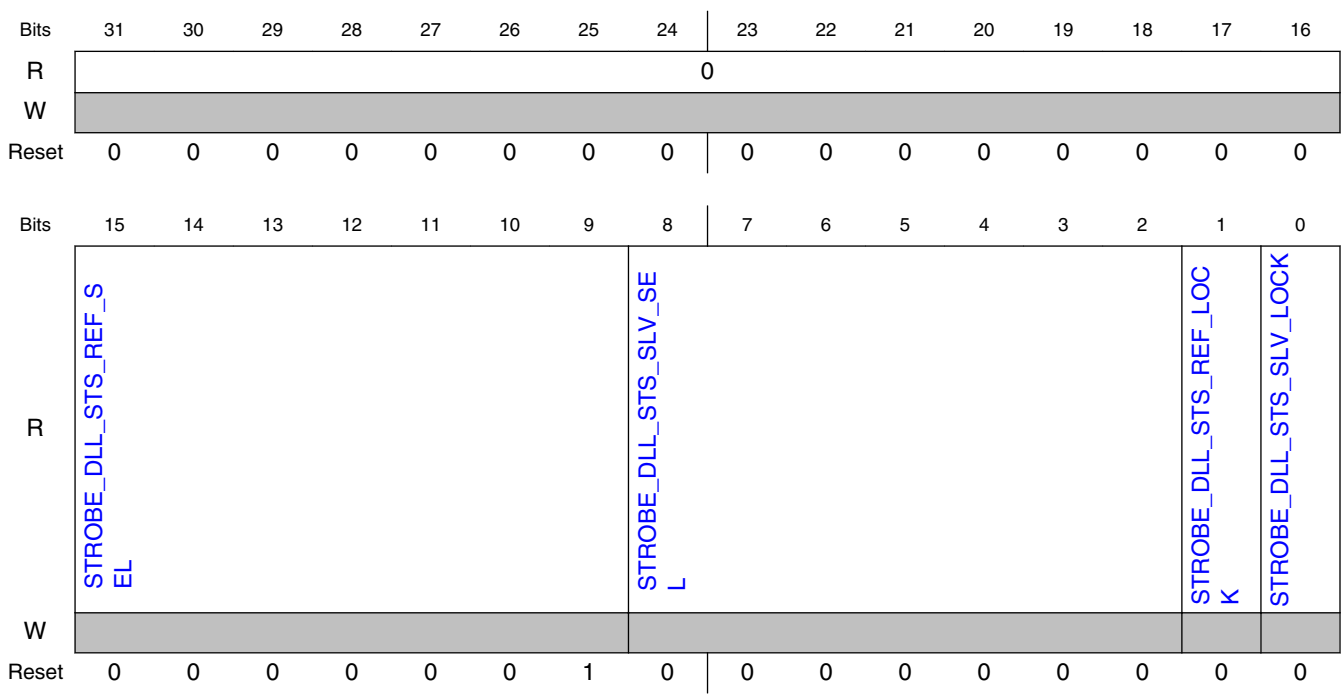
23.5.1.28.1 Offset

Register	Offset
STROBE_DLL_STATUS	74h

23.5.1.28.2 Function

This register contains the strobe DLL status information. All fields are read only and read the same as the power-reset value.

23.5.1.28.3 Diagram



23.5.1.28.4 Fields

Field	Function
31-16	This field is reserved.
—	This read-only field is reserved and always has the value 0.
15-9	Strobe DLL status reference select
	Reference delay line select taps. This is encoded by 7 fields for 127 taps.

Table continues on the next page...

Field	Function
STROBE_DLL_STS_REF_SEL	
8-2 STROBE_DLL_STS_SLV_SEL	Strobe DLL status slave select Slave delay line select status. This is the instant value generated from reference chain. Because the reference chain can only be updated when STROBE_REF_CLOCK is detected, this value can be updated with the right value when the reference is locked.
1 STROBE_DLL_STS_REF_LOCK	Strobe DLL status reference lock This signifies that the DLL has detected and locked to a half-phase REF_CLOCK shift, it allows the slave delay-line to perform programmed clock delays.
0 STROBE_DLL_STS_SLV_LOCK	Strobe DLL status slave lock Slave delay-line lock status. This signifies that a valid calibration has been set to the slave-delay line, and the slave-delay line is implementing the programmed delay value.

23.5.1.29 Vendor Specific Register (VEND_SPEC)

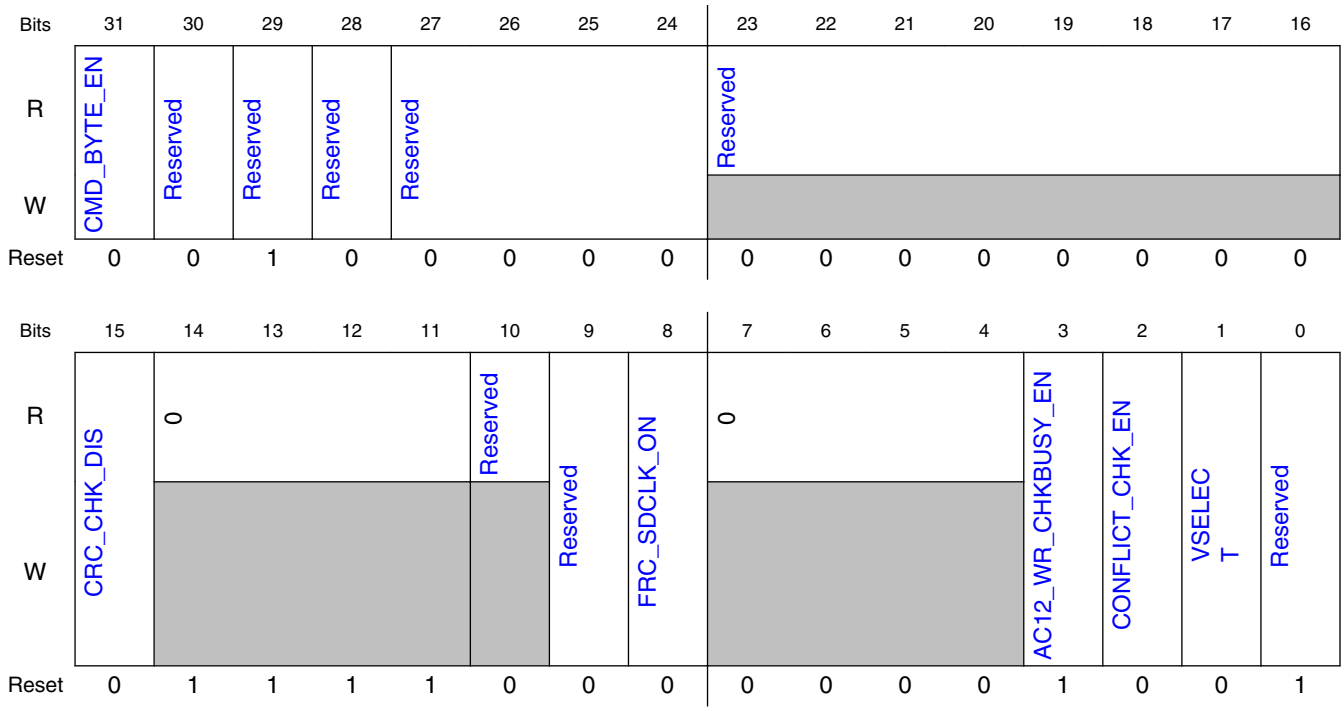
23.5.1.29.1 Offset

Register	Offset
VEND_SPEC	C0h

23.5.1.29.2 Function

This register contains the vendor specific control / status register.

23.5.1.29.3 Diagram



23.5.1.29.4 Fields

Field	Function
31 CMD_BYTE_EN	Byte access This bit controls the byte access. 0b - Disable 1b - Enable
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 1
28 —	Reserved Always write as 0.
27-24 —	Reserved Always write as 4'b0000.
23-16 —	Reserved Always write as 8'h00.
15 CRC_CHK_DIS	CRC Check Disable 0b - Check CRC16 for every read data packet and check CRC fields for every write data packet 1b - Ignore CRC16 check for every read data packet and ignore CRC fields check for every write data packet

Table continues on the next page...

Field	Function
14-11 —	Reserved Reserved
10 —	Reserved Always write as 0.
9 —	Reserved Always write as 0.
8 FRC_SDCLK_ON	Force CLK Force CLK output active 0b - CLK active or inactive is fully controlled by the hardware. 1b - Force CLK active
7-4 —	Reserved
3 AC12_WR_CHKBUSY_EN	Check busy enable Check busy enable after auto CMD12 for write data packet 0b - Do not check busy after auto CMD12 for write data packet 1b - Check busy after auto CMD12 for write data packet
2 CONFLICT_CHK_EN	Conflict check enable It is not implemented in uSDHC IP. 0b - Conflict check disable 1b - Conflict check enable
1 VSELECT	Voltage selection Change the value of output signal VSELECT, to control the voltage on pads for external card. There must be a control circuit out of uSDHC to change the voltage on pads. 0b - Change the voltage to high voltage range, around 3.0 V 1b - Change the voltage to low voltage range, around 1.8 V
0 —	Reserved Always write as 0.

23.5.1.30 MMC Boot (MMC_BOOT)

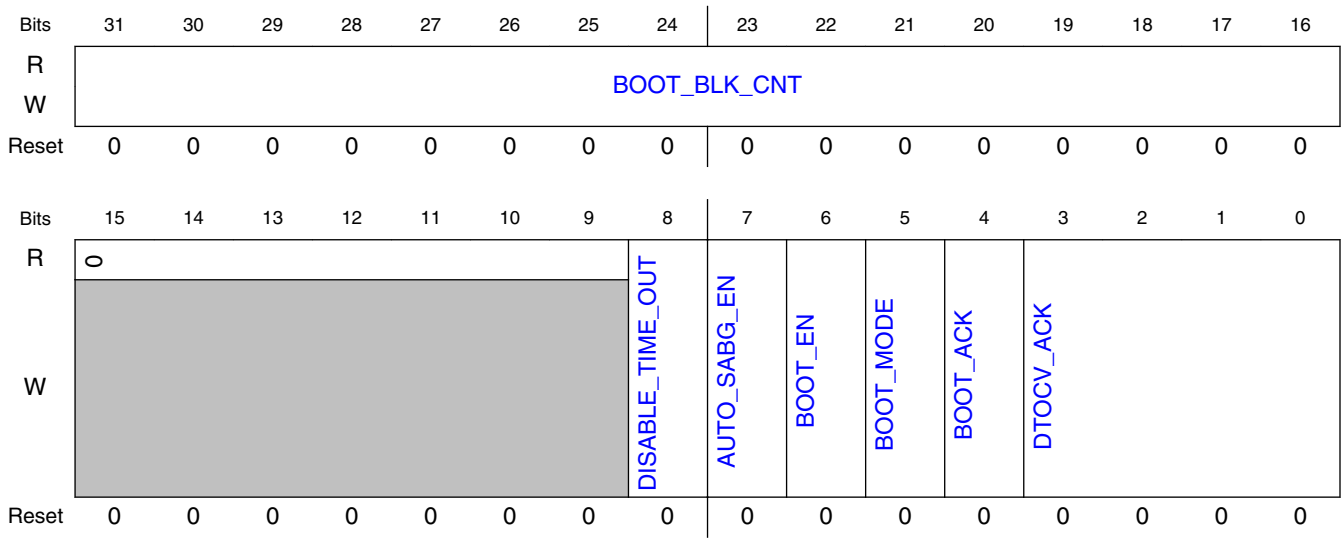
23.5.1.30.1 Offset

Register	Offset
MMC_BOOT	C4h

23.5.1.30.2 Function

This register contains the MMC Fast Boot control register.

23.5.1.30.3 Diagram



23.5.1.30.4 Fields

Field	Function
31-16 BOOT_BLK_CNT	Stop At Block Gap value of automatic mode The value defines the Stop At Block Gap value of automatic mode. When received, card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT) and AUTO_SABG_EN is 1, then Stop At Block Gap. Here, BLK_CNT is defined in the Block Attributes Register, field 31 - 16 of 0x04.
15-9 —	Reserved
8 DISABLE_TIME_OUT	Time out **NOTE:** When this field is set, there is no timeout check no matter whether BOOT_EN is set or not. 0b - Enable time out 1b - Disable time out
7 AUTO_SABG_EN	Auto stop at block gap During boot, enable auto stop at block gap function. This function is triggered, and host stops at block gap when received card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT).
6 BOOT_EN	Boot enable Boot mode enable 0b - Fast boot disable 1b - Fast boot enable
5 BOOT_MODE	Boot mode Boot mode select 0b - Normal boot 1b - Alternative boot
4	BOOT ACK

Table continues on the next page...

Field	Function
BOOT_ACK	Boot ACK mode select 0b - No ack 1b - Ack
3-0 DTOCV_ACK	Boot ACK time out Boot ACK time out counter value. 0000b - SDCLK x 2^14 0001b - SDCLK x 2^15 0010b - SDCLK x 2^16 0011b - SDCLK x 2^17 0100b - SDCLK x 2^18 0101b - SDCLK x 2^19 0110b - SDCLK x 2^20 0111b - SDCLK x 2^21 1110b - SDCLK x 2^28 1111b - SDCLK x 2^29

23.5.1.31 Vendor Specific 2 Register (VEND_SPEC2)

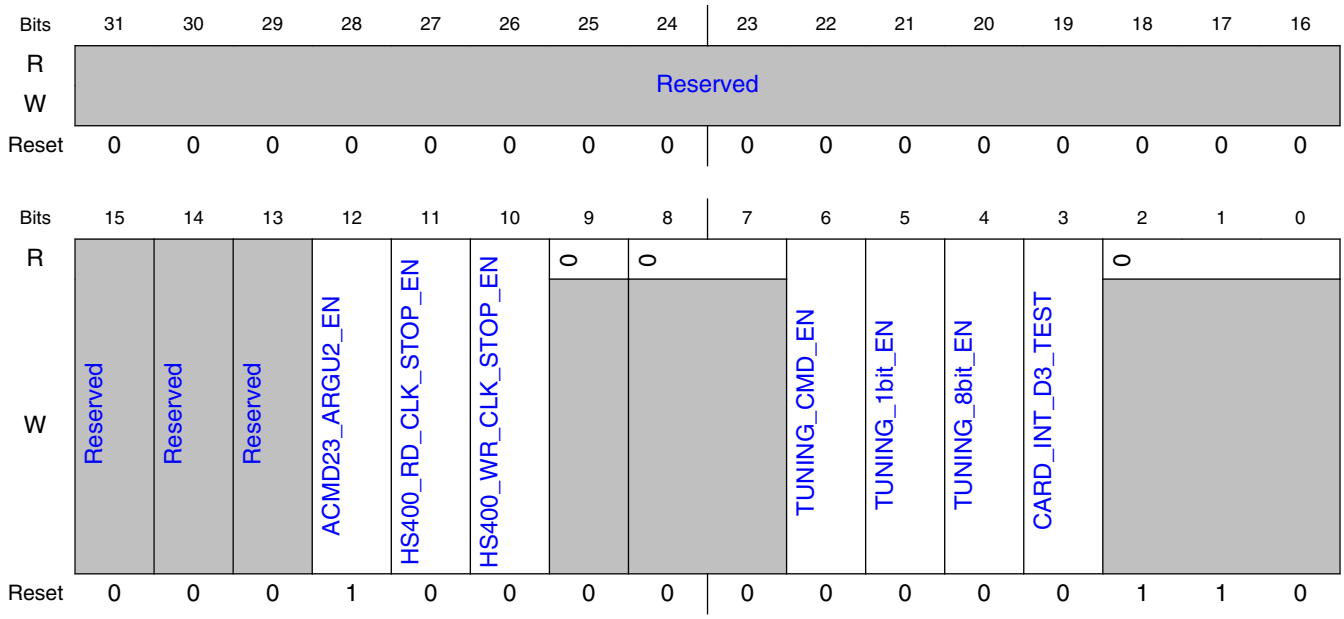
23.5.1.31.1 Offset

Register	Offset
VEND_SPEC2	C8h

23.5.1.31.2 Function

This register contains the vendor specific control 2 register.

23.5.1.31.3 Diagram



23.5.1.31.4 Fields

Field	Function
31-15 —	Reserved
14 —	Reserved
13 —	Reserved
12 ACMD23_ARGU2_EN	Argument2 register enable for ACMD23 0b - Disable 1b - Argument2 register enable for ACMD23 sharing with SDMA system address register. Default is enabled.
11 HS400_RD_CLK_STOP_EN	HS400 read clock stop enable Only stop clock at read block gap.
10 HS400_WR_CLK_STOP_EN	HS400 write clock stop enable Only stop clock at write block gap.
9 —	Reserved
8-7 —	Reserved
6	Tuning command enable

Table continues on the next page...

Field	Function
TUNING_CMD_EN	Enable the auto tuning circuit to check the CMD line. 0b - Auto tuning circuit does not check the CMD line. 1b - Auto tuning circuit checks the CMD line.
5 TUNING_1bit_EN	Tuning 1bit enable Enable the auto tuning circuit to check the DATA0 only. It is used with the TUNING_8bit_EN together.
4 TUNING_8bit_EN	Tuning 8bit enable Enable the auto tuning circuit to check the DATA[7:0]. It is used with the TUNING_1bit_EN together. 0b00 - Tuning circuit only checks the DATA[3:0] 0b01 - Tuning circuit only checks the DATA0 0b10 - Tuning circuit checks the whole DATA[7:0] 0b11 - Invalid **NOTE:** The format of these two fields are [TUNNING_8bit_EN:TUNNING_1bit_EN].
3 CARD_INT_D3_TEST	Card interrupt detection test This field only uses for debugging. 0b - Check the card interrupt only when DATA3 is high. 1b - Check the card interrupt by ignoring the status of DATA3.
2-0 —	Reserved

23.5.1.32 Tuning Control (TUNING_CTRL)

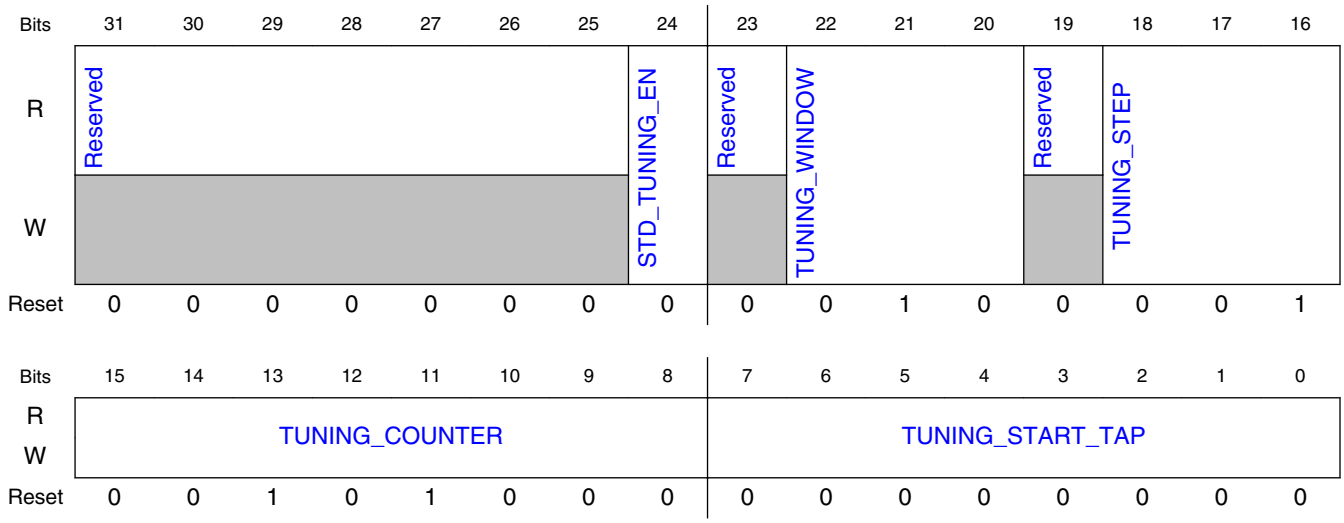
23.5.1.32.1 Offset

Register	Offset
TUNING_CTRL	CCh

23.5.1.32.2 Function

The register contains configuration of tuning circuit.

23.5.1.32.3 Diagram



23.5.1.32.4 Fields

Field	Function
31-25 —	Reserved
24 STD_TUNING_EN	Standard tuning circuit and procedure enable This field is used to enable standard tuning circuit and procedure.
23 —	Reserved
22-20 TUNING_WINDOW	Data window Select data window value for auto tuning
19 —	Reserved
18-16 TUNING_STEP	TUNING_STEP The increasing delay cell steps in tuning procedure.
15-8 TUNING_COUNTER	Tuning counter The MAX repeat CMD19 times in tuning procedure.
7-0 TUNING_START_TAP	Tuning start The start delay cell point when send first CMD19 in tuning procedure.

23.6 Functional description

The following sections provide a brief functional description of the major system blocks, including the data buffer, DMA AHB interface, register bank as well as IP Bus interface, dual-port memory wrapper, data/command controller, clock and reset manager, and clock generator.

23.6.1 Data buffer

The uSDHC module uses one configurable data buffer to transfer data between the system bus (IP bus or advanced high-performance bus (AHB) bus) and the SD card in an optimized manner, maximizing throughput between the two clock domains (IP peripheral clock and the master clock).

The buffer is used as a temporary storage for transferring data between the host system and the card. The watermark levels for read and write are both configurable and can range between 1 to 128 words. The burst lengths for read and write are also configurable and can range between 1 to 31 words. The next figure provides the uSDHC buffer scheme.

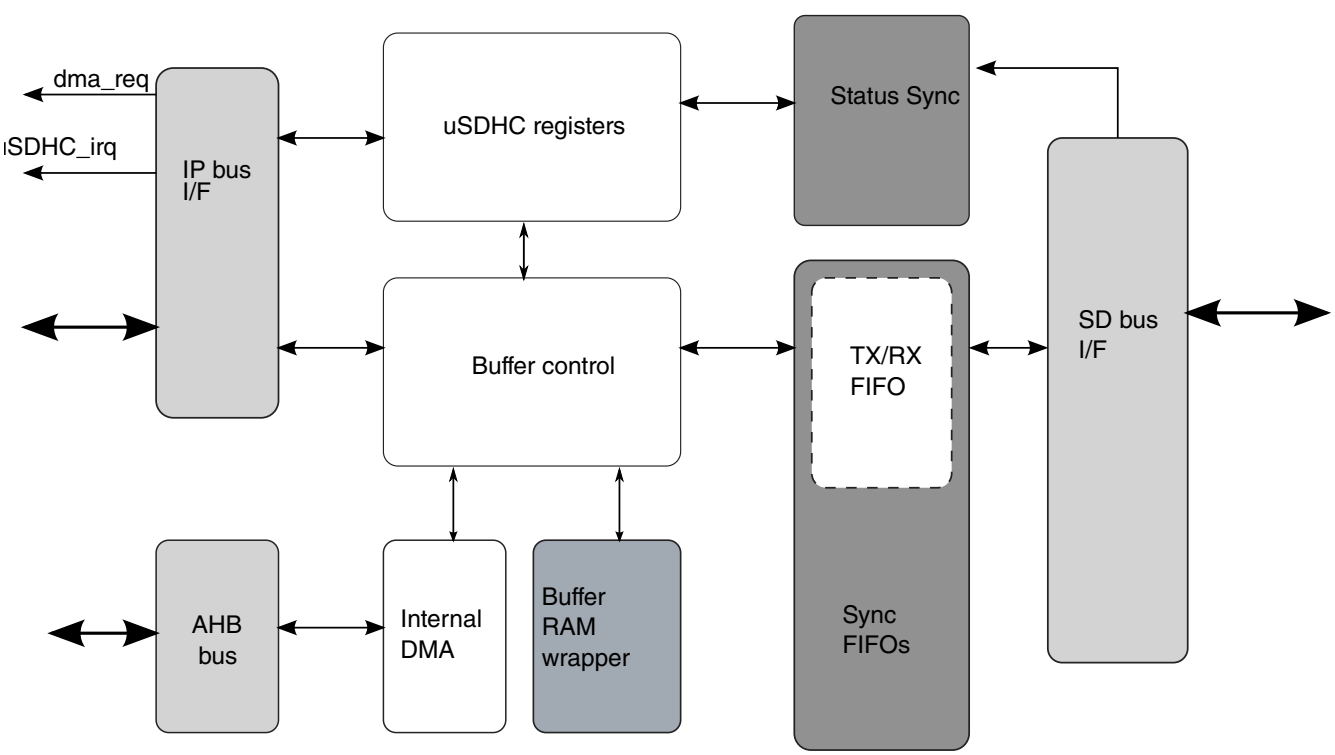


Figure 23-3. uSDHC buffer scheme

Here are 2 the two transfer modes to access the data buffer:

- CPU polling mode:
  - For a host-read operation, when the number of words received in the buffer meets or exceeds the RD_WML watermark value, by polling the BRR bit, the host driver can read the Buffer Data Port register to fetch the amount of words set in the RD_WML register from the buffer. The write operation is similar. For more information on the process of writing operation, see Write operation sequence.
- Internal DMA mode (includes simple and advanced DMA accesses):
  - The internal DMA access, either by simple or advanced DMA, is over the AHB bus.

For a read operation, when there are more words in the buffer than the amount set in the RD_WML register, the internal DMA starts fetching data over the AHB bus. Except for INCR4 and INCR8, the burst type is always the INCR mode and the burst length depends on the shortest of the following factors:

- Burst length configured in the burst length field of the Watermark Level register
- Watermark level boundary
- Block size boundary
- Data boundary configured in the current descriptor (if the ADMA is active)
- 1 KB address boundary defined in the AHB protocol

The Write operation functions in a similar manner—sequential and contiguous access is necessary to ensure that the pointer address value is correct. Random or skipped access is not possible. The byte order, by reset, is little endian mode. The actual byte order is swapped inside the buffer, according to the endian mode configured by software (see the following figures). For a host write operation, the byte order is swapped after the data is fetched from the buffer and ready to send to the SD Bus. For a host read operation, the byte order is swapped before the data is stored in the buffer.

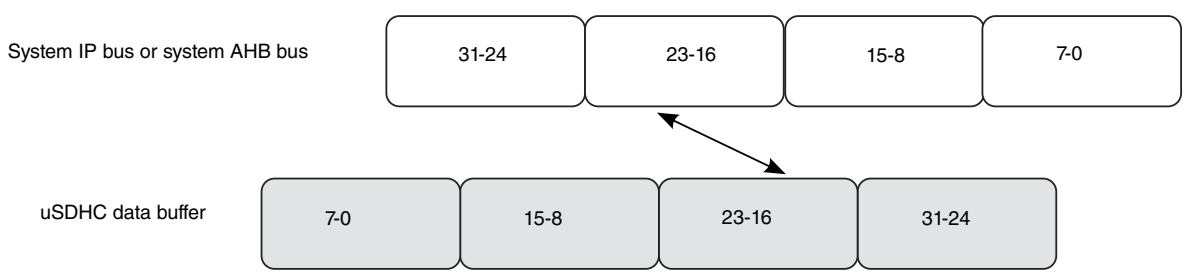


Figure 23-4. Data swap between system bus and uSDHC data buffer in the byte little endian mode

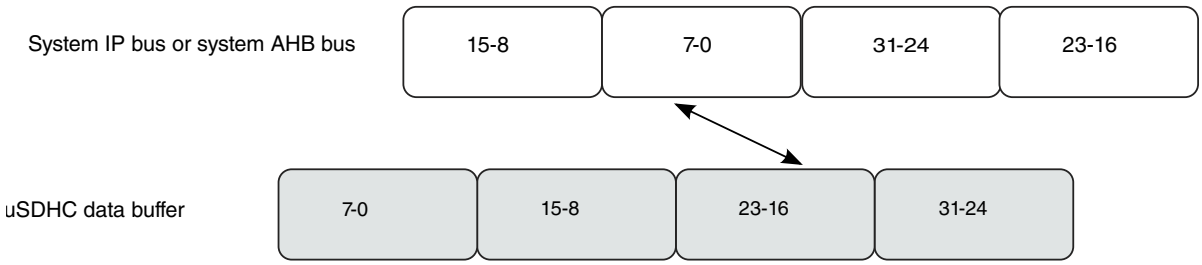


Figure 23-5. Data swap between system bus and uSDHC data buffer in half word big endian mode

23.6.1.1 Write operation sequence

There are 2 ways to write data into the buffer when the user transfers data to the card:

- Processor core polling through the BWR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When the internal DMA is not used, the DMAEN bit in the Transfer Type register is not set when the command is sent, uSDHC asserts a DMA request when the amount of buffer space exceeds the value set in the WR_WML register and is ready for receiving new data. At the same time, uSDHC sets the BWR bit. The buffer write ready interrupt is generated if it is enabled by software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the contents of the DMA System Address register to obtain the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the write operation from that address. It is recommended that a software reset for Data be applied before the transfer is restarted.

The uSDHC module does not start data transmission until the number of words set in the WR_WML register can be held in the buffer. If the buffer is empty and the host system does not write data in time, uSDHC stops the CLK to avoid the data buffer underrun situation.

23.6.1.2 Read operation sequence

There are 2 ways to read data from the buffer when the user transfers data to the card:

- Processor core polling through the BRR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When internal DMA is not used (DMAEN bit in Transfer Type register is not set when the command is sent), uSDHC asserts a DMA request when the amount of data exceeds the value set in the RD_WML register, which is available and ready for system fetching data. At the same time, uSDHC sets the BRR bit. The buffer read ready interrupt is generated if it is enabled by the software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the content of the DMA System Address register to get the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the read operation from that address. It is recommended that a software reset for data be applied before the transfer is restarted.

For any write transfer mode, uSDHC does not start data transmission until the number of words set in the RD_WML register are in buffer. If the buffer is full and the host system does not read data in time, uSDHC stops the CLK to avoid the data buffer overrun situation.

23.6.1.3 Data buffer and block size

The user needs to know the buffer size for the buffer operation during a data transfer to utilize it in the most optimized way. In the uSDHC module, the only data buffer can hold up to 128 words (32-bit) and the watermark levels for write and read can be configured accordingly.

For both read and write, the watermark level can range between 1 to 128 words. For both read and write, the burst length can range between from 1 to 31 words. The host driver may configure the value according to the system situation and requirement.

During a multi-block data transfer, the block length can be set to any value between 1 and 4096 bytes, satisfying the requirements of the external card. The only restriction is from the external card, which can be limited in size or support of a partial block access (which is not the integer times of 512 bytes).

As uSDHC treats each block individually, for block sizes which are not multiples of four (not word-aligned), stuffed bytes are required at the end of each block. For example, if the block size is 7 bytes and there are 12 blocks to write, the system side must write twice for each block. For each block, the ending byte is abandoned by uSDHC because it only sends 7 bytes to the card and picks data from the following system write, resulting in 24 beats of write access in total.

23.6.1.4 Dividing large data transfer

This SDIO command CMD53 definition limits the maximum data size of data transfers according to the following formula:

Max data size = Block size x Block count

The length of a multiple block transfer needs to be in block size units. If the total data length cannot be divided evenly into a multiple of the block size, then there are two ways to transfer the data. These two ways depend on the function and the card design. Option 1 is for the host driver to split the transaction. The remainder of the block size data is then transferred by using a single block command at the end. Option 2 is to add dummy data in the last block to fill the block size. For option 2, the card must manage the removal of the dummy data.

See the figure below for an example showing the division of large data transfers, assuming a kind of WLAN SDIO card that only supports a block size of up to 64 bytes. Although uSDHC supports a block size of up to 4096 bytes, the SDIO can only accept a block size of less than 64 bytes, so the data must be divided (see the example below).

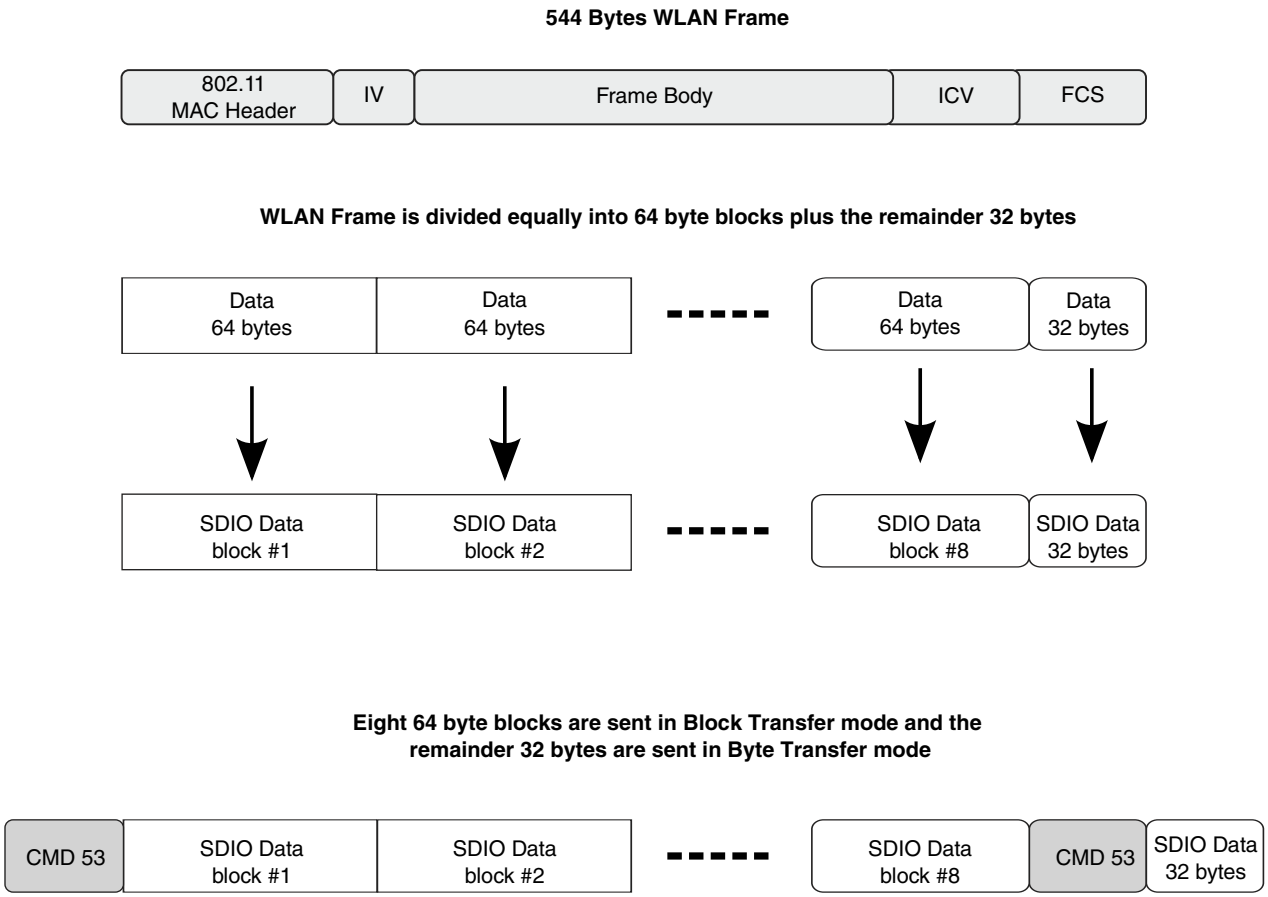


Figure 23-6. Example for dividing large data transfers

23.6.2 DMA AHB interface

The internal DMA implements a DMA engine and the AHB master. When the internal DMA is enabled, the uSDHC_dreq_b is not asserted during the transfer, but the BWR and BRR bits are set if the BWRSEN and BRRSEN bits have been set in the Interrupt Status Enable register.

See the figure below for an illustration of the DMA AHB interface block.

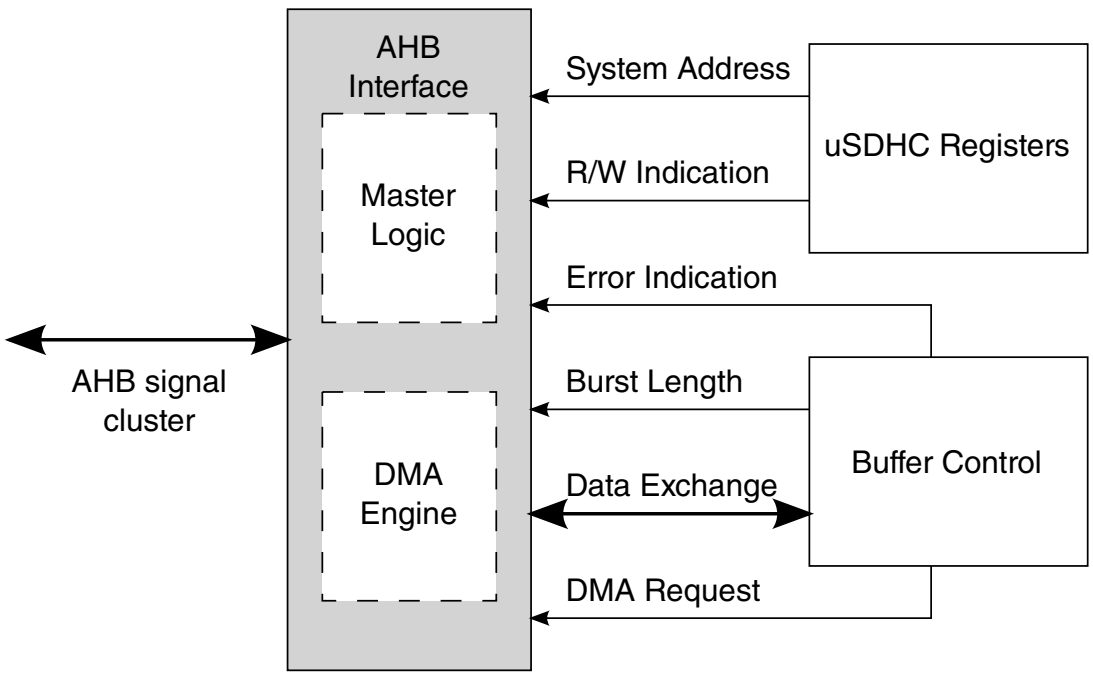


Figure 23-7. DMA AHB interface block

23.6.2.1 Internal DMA request

If the watermark level requirement is met in data transfer or if the last data of current block is ready in the data buffer, and the Internal DMA is enabled, the data buffer block sends a DMA request to AHB interface.

The delay in response from the internal DMA engine depends on the system AHB bus loading and the priority assigned to uSDHC. The DMA engine does not respond to the request during its burst transfer, but is ready to serve as soon as the burst is over. The data buffer de-asserts the request if the data buffer space (for write) or bytes in data buffer is smaller than the watermark level. Upon access to the buffer by internal DMA, the data buffer updates its internal buffer pointer, and when the watermark level is satisfied or the last data of the current block is ready in the data buffer, another DMA request is sent.

The data transfer is in the block unit, and the subsequent watermark level is always set as the remaining number of words. For instance, for a multi block data read with each block size of 31 bytes, and the burst length set to six words. After the first burst transfer, if there are more than two words in the buffer, which might contain some data of the next block), another DMA request is sent. This is because the remaining number of words to send for the current block is (31 - 6 * 4) / 4 = 2. The uSDHC module reads two words out of the buffer, with seven valid bytes and one stuffed byte.

23.6.2.2 DMA burst length

Just like a CPU polling access, the DMA burst length for the internal DMA engine can range between 1 to 16 words. The actual burst length for the DMA depends on the lesser of the configured burst length or the remaining words of the current block.

See the example in Internal DMA request. After six words are read, the burst length is two words, then the next burst length is six words. This is because the next block starts, which is 31 bytes, more than six words. The host driver may take this variable burst length into account. It is also acceptable to configure the burst length as the divisor of the block size, so that each time the burst length is the same.

23.6.2.3 AHB master interface

It is possible that the internal AHB DMA engine could fail during the data transfer. Upon detection of an AHB bus error during DMA transfer, the DMA engine stops the transfer and goes to the idle state. At that point, the internal data buffer stops receiving incoming data and sending out data. The DMAE bit in the Interrupt Status register is generated to host CPU to report a bus error condition.

After the DMAE interrupt is received, the software sends a CMD12 to abort the current transfer and read the DS_ADDR bits of the DMA System Address register to get the starting address of the corrupted block. After the DMA error is fixed, the software should apply a data reset and restart the transfer from this address to recover the corrupted block. DMA operation resumes when the interrupt is serviced by the software.

23.6.2.4 ADMA engine

In the SD host controller standard, a new DMA transfer algorithm called the Advanced DMA (ADMA) is defined. For simple DMA, after the page boundary is reached, a DMA interrupt is generated and the new system address is programmed by the host driver.

The ADMA defines the programmable descriptor table in the system memory. The host driver can calculate the system address at the page boundary and program the descriptor table before executing ADMA. It reduces the frequency of interrupts to the host system. Therefore, higher speed DMA transfers could be realized because the host MCU intervention is not be needed during long DMA-based data transfers.

There are two types of ADMA in host controller: ADMA1 and ADMA2. ADMA1 can support data transfer of 4KB aligned data in system memory, and ADMA2 improves the restriction so that the data of any location and any size can be transferred in system memory. Their formats of Descriptor Table are different.

ADMA can recognize all kinds of descriptors defined in SD host controller Standard, and if the "End" flag is detected in the descriptor, ADMA stops after this descriptor is processed.

23.6.2.4.1 ADMA concept and descriptor format

ADMA1 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Set data length descriptor
- Set data address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA2 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Rsv descriptor
- Set data length and address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA starts read/write operation after it reaches the tran state, using the data length and data address analyzed from most recent descriptor(s).

For ADMA1, the valid data length descriptor is the last set type descriptor before the tran type descriptor. Every tran type triggers a transfer, and the transfer data length is extracted from the most recent set type descriptor. If there is no set type descriptor after the previous Trans descriptor, the data length is the value for previous transfer, or 0 if no set descriptor is ever met.

For ADMA2, the tran type descriptor contains both data length and transfer data address, so only a tran type descriptor can start a data transfer.

See the figure below for the format of the descriptor table for ADMA1.

Address/ Page Field		Address/ Page Field		Attribute Field					
31	12	11	6	5	4	3	2	1	0
Address or Data Length		000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	31- 28	27- 12
0	0	Nop	No Operation	Don't Care	
0	1	Set	Set Data Length	0000	Data Length
1	0	Tran	Transfer Data	Data Address	
1	1	Link	Link Descriptor	Descriptor Address	

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 23-8. Format of the ADMA1 descriptor table

See the figure below for the concept and access method of the descriptor table for ADMA1.

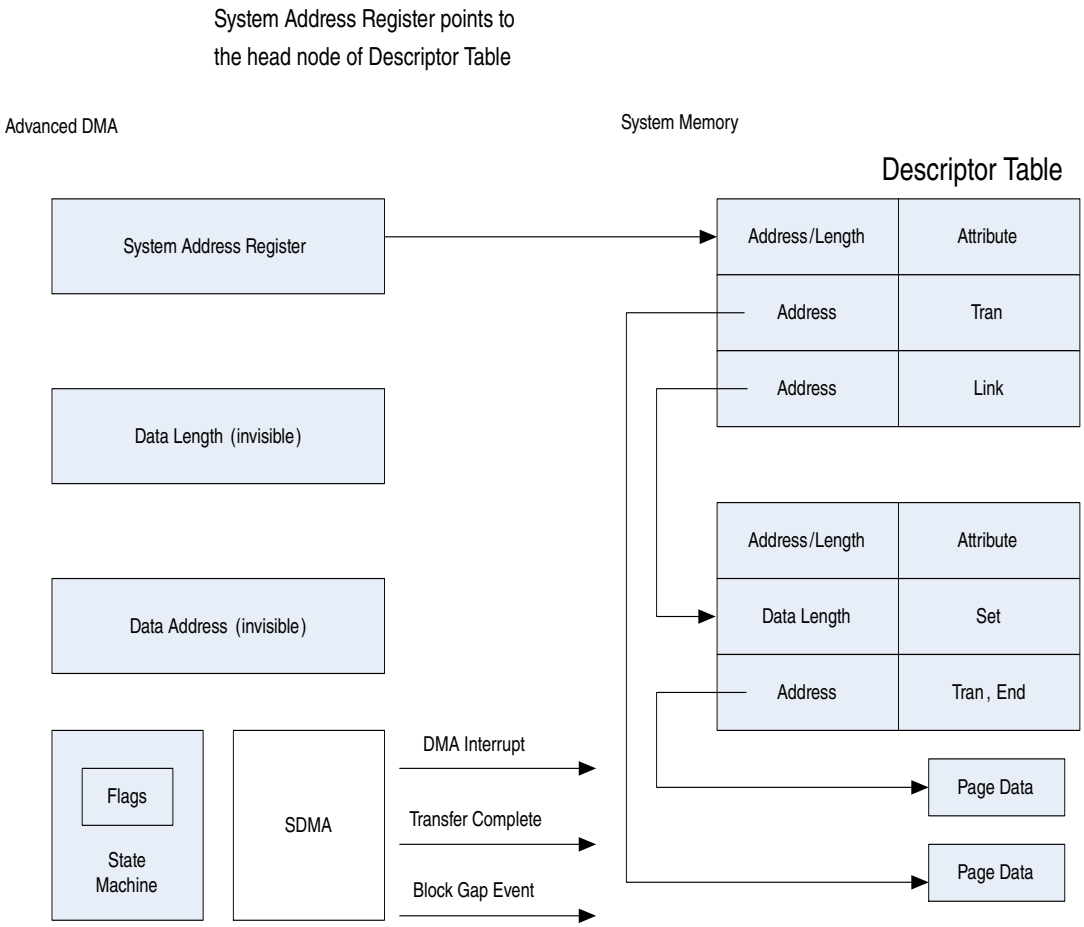


Figure 23-9. Concept and access method of the ADMA1 descriptor table

The figure below explains the ADMA2 format. ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it ignores the higher 32-bit. The Address field should be set to word aligned (lower 2-bit is always set to 0). Data length is in byte unit.

Address Field		Length		Reserved		Attribute Field					
63	32	31	16	15	06	05	04	03	02	01	00
32-bit Address		16-bit length		0000000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	Operation
0	0	Nop	No Operation	Don't Care
0	1	Rsv	Reserved	Same as Nop. Read this line and go to next one
1	0	Tran	Transfer Data	Transfer data with address and length set in this descriptor line
1	1	Link	Link Descriptor	Link to another descriptor

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 23-10. Format of the ADMA2 descriptor table

See the figure below for the concept and access method of the descriptor table for ADMA2.

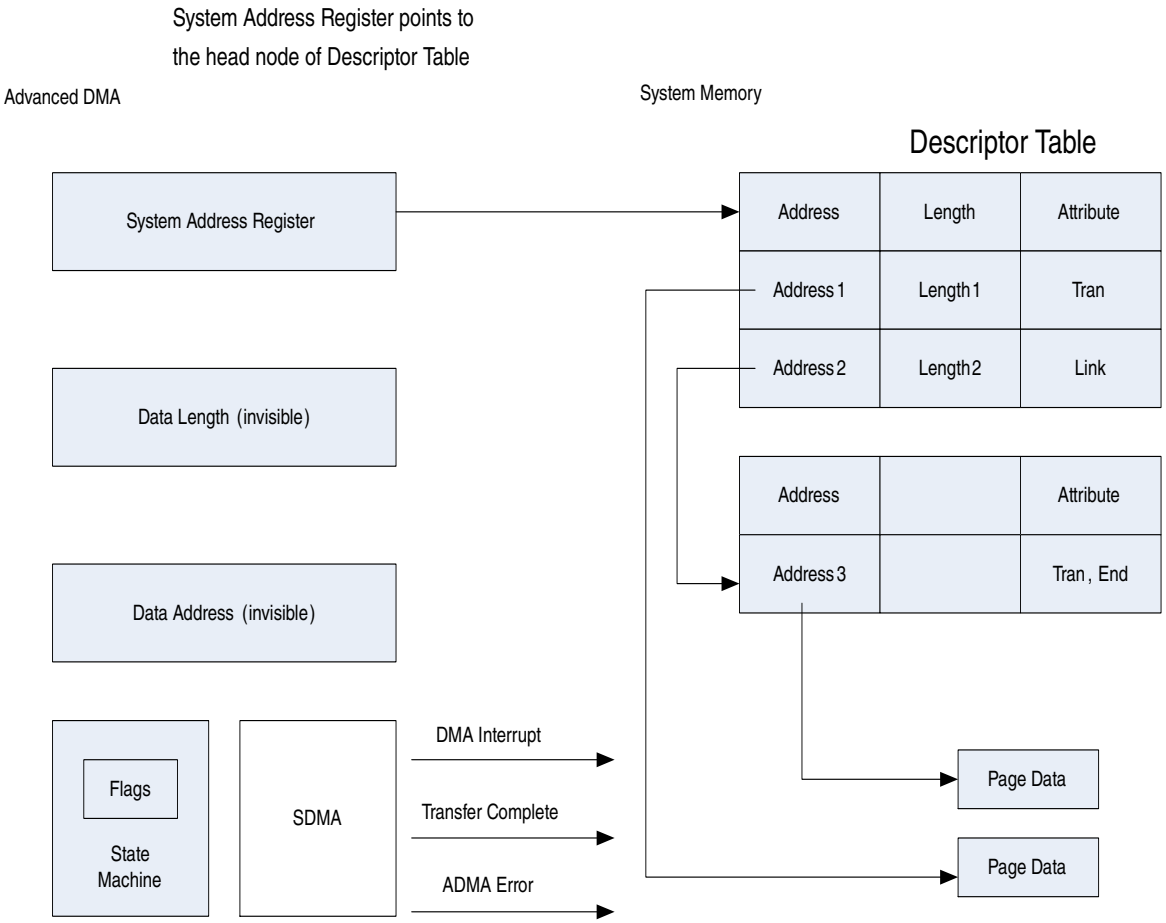


Figure 23-11. Concept and access method of the ADMA2 descriptor table

23.6.2.4.2 ADMA interrupt

If the interrupt flag descriptor is set, ADMA generates an interrupt according to various types of descriptors.

For ADMA1:

- Set type of descriptor: interrupt is generated when data length is set.
- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop type of descriptor: interrupt is generated just after this descriptor is fetched.

For ADMA2:

- Tran type of descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop/Rsv type of descriptor: interrupt is generated just after this descriptor is fetched.

23.6.2.4.3 ADMA error

The ADMA stops whenever an error is encountered. These errors include:

- Fetching descriptor error
- AHB response error
- Data length mismatch error

An ADMA descriptor error is generated when it fails to detect a "valid" flag in the descriptor. If an ADMA descriptor error occurs, the interrupt is not generated even if the "Interrupt" flag of this descriptor is set.

When BLKCNTEN bit is set, data length set in buffer must be equal to the whole data length set in descriptor nodes, otherwise data length mismatch error is generated.

When BLKCNTEN bit is not set, then the whole data length set in descriptor is a multiple of block lengths; otherwise, when data set in the descriptor nodes are not performed at block boundaries, then data mismatch errors occur.

23.6.3 Register bank with IP bus interface

Register accesses through the IP bus interface are on the register bank.

See the figure below for the block diagram.

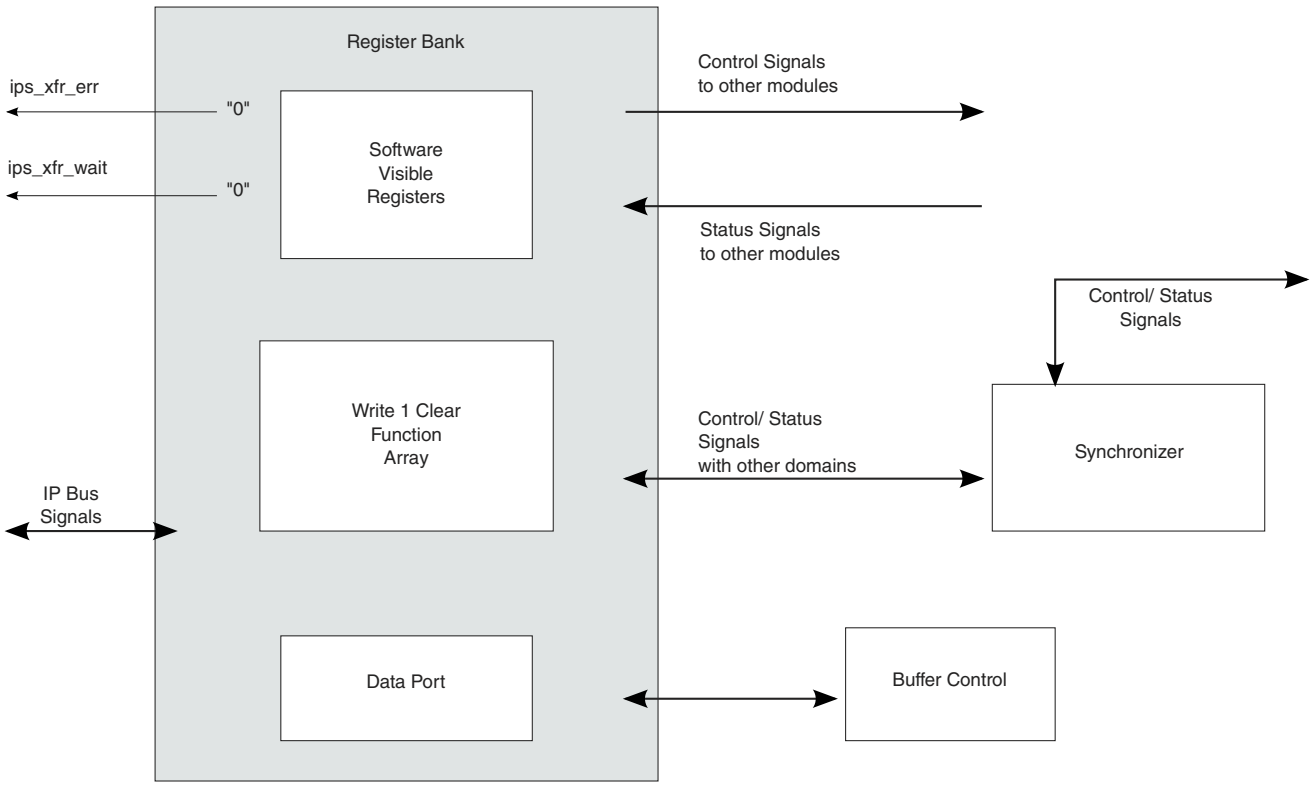


Figure 23-12. Register bank diagram

Only a 32-bit access is allowed, and no partial read/write is supported; therefore, all accesses are word aligned.

23.6.3.1 SD protocol unit

The SD protocol unit deals with all SD protocol affairs.

The SD protocol unit performs the following functions:

- Acts as the bridge between the internal buffer and the SD bus
- Sends the command data as well as its argument serially
- Stores the serial response bit stream into corresponding registers
- Detects the bus state on the CMD/DAT lines
- Monitors the interrupt from the SDIO card
- Asserts the read wait signal
- Gates off the SD clock when buffer is announcing danger status
- Detects the write protect state

The SD protocol unit consists of four submodules:

- SD control misc

- Command control
- Data control
- Clock control

23.6.3.2 SD control misc

In the SD control misc unit, the card detection (including the CD_B and DATA3 used for card detection), write protection, and card interrupt are implemented.

This module monitors the signal level on all the eight data lines and the command lines. It directly routes the level values into the register bank.

The module also detects the Write Protect (WP) line. If WP is active, writes to the register bank are ignored.

This module also drives the LCTL output signal when the LCTL bit is set by the driver.

23.6.3.3 SD clock control

If the internal data buffer is near full (for read) or near empty (for write), the SD clock must be gated off to avoid buffer over/under-run. This module asserts the gate of the output SD clock to shut the clock off. After the buffer has space (for read) or has data (for write), the clock gate of this module opens, and the SD clock is active again.

23.6.3.4 Command control

The Command Control module deals with the transactions on the CMD line.

See the figure below for an illustration of the structure for the Command CRC shift register.

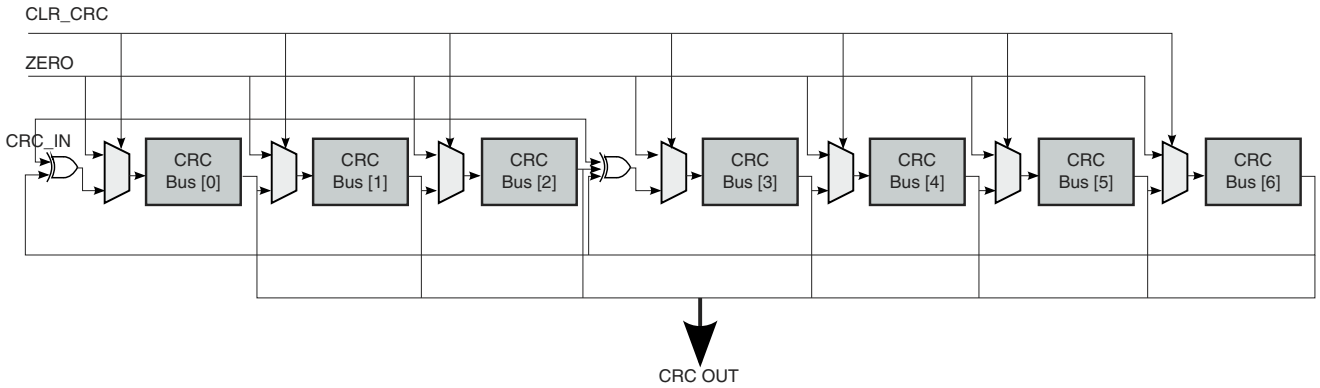


Figure 23-13. Command CRC shift register

The CRC polynomials for the CMD are as follows:

```
Generator polynomial: G(x) = x^7 + x^3 + 1
M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0
CRC[6:0] = Remainder [(M(x) * x^7) / G(x)]
```

23.6.3.5 Data control

The data agent deals with the transactions on the eight data lines. Moreover, this module also detects the busy state on the DATA0 line, and generates the read wait state by the request from the transceiver.

The CRC polynomials for the data are as follows:

```
Generator polynomial: G(x) = x^16 + x^12 + x^5 +1
M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0
CRC[15:0] = Remainder [(M(x) * x^16) / G(x)]
```

23.6.4 Clock and reset manager

This module controls all four kinds reset signals within uSDHC:

- Hardware reset
- Software reset for all logic
- Software reset for the data logic
- Software reset for the command logic

All these signals are fed into this module and stable signals are generated inside the module to reset all other modules. The module also gates off all the inside signals.

23.6.5 Clock generator

The clock generator generates the card CLK by peripheral source clock in two stages. The clock divisor can be configured through register SYS_CTRL [SDCLKFS] is for prescaler configuration while the [DVS] is for divisor configuration. Details can be found in the register function description. See the following figure for the structure of the divider. The term "Base" represents the frequency of peripheral source clock.

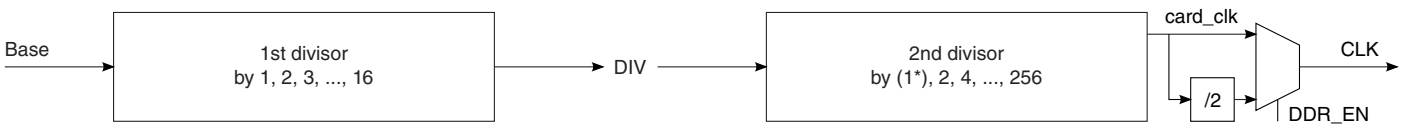


Figure 23-14. Two stages of the clock divider

The first stage outputs an intermediate clock (DIV) that can be Base, Base/2, Base/3, ..., or Base/16.

The second stage is a prescaler and outputs the actual internal working clock (card_clk). This clock is the driving clock for all the sub modules of the SD protocol unit, and helps in syncing FIFOs (see Figure 23-3) to synchronize with the data rate from the internal data buffer. The frequency of the clock output from this stage can be DIV, DIV/2, DIV/4,..., or DIV/256. Therefore, the highest frequency of the card_clk is base, and the next highest is Base/2, while the lowest frequency is Base/4096. If the duty cycle of the base clock is 50%, the duty cycle of card_clk is also 50%, even when the compound divisor is an odd value.

NOTE

CLK is different for the SDR and DDR modes.

- In the SDR mode, CLK is equal to the internal working clock (card_clk).
- In the DDR mode, CLK is equal to card_clk/2.

23.6.6 SDIO card interrupt

Information on interrupts in 1-bit mode, interrupts in 4-bit mode, and card interrupt handling are detailed in the sections below.

23.6.6.1 Interrupts in 1-bit mode

In this case, the DATA1 pin provides the interrupt function. An interrupt is asserted by pulling the DATA1 low from the SDIO card, until the interrupt service is finished to clear the interrupt.

23.6.6.2 Interrupt in 4-bit mode

As the interrupt and data line 1 share pin 8 in a 4-bit mode, an interrupt is only sent by the card and recognized by the host during a specific time. This is known as the interrupt period. The uSDHC module only provides sample the level on pin 8 during the interrupt period. At all other times, the host ignores the level on pin 8 and treats it as the data signal. The definition of the interrupt period is different for operations with single block and multiple block data transfers.

In the case of normal single data block transmissions, the interrupt period becomes active two clock cycles after the completion of a data packet. This interrupt period lasts until after the card receives the end bit of the next command that has a data block transfer associated with it.

For multiple block data transfers in a 4-bit mode, there is only a limited period of time that the interrupt period can be active because of the limited period of data line availability between the multiple blocks of data. This requires stricter definition of the interrupt period. For this case, the interrupt period is limited to two clock cycles. This begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DATA1 line holds low for one clock cycle with the last clock cycle pulling DATA1 high. On completion of the interrupt period, the card releases the DATA1 line into the high Z state. The uSDHC module provides sample of the DATA1 during the interrupt period when the IABG bit in the Protocol Control register is set.

Refer to SDIO Card Specification v1.10 for further information about the SDIO card interrupt.

23.6.6.3 Card interrupt handling

When the CINTIEN bit in the Interrupt Signal Enable Register is set to 0, uSDHC clears the interrupt request to the host system. The host driver should clear this bit before servicing the SDIO interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

The SDIO Card Interrupt Status can be cleared by writing 1 to this bit. But as the interrupt source from the SDIO card does not clear, this bit is set again. To clear this bit, it is required to reset the interrupt source from the external card followed by writing 1 to this bit. In a 1-bit mode, uSDHC detects the SDIO interrupt with or without the SD clock (to support wakeup). In a 4-bit mode, the interrupt signal is sampled during the interrupt period, so there are some sample delays between the interrupt signal from the SDIO card and the interrupt to the Host System Interrupt Controller. When the SDIO status is set and the host driver needs to service this interrupt, the SDIO bit in the Interrupt Control Register of SDIO card is cleared. This is required to clear the SDIO interrupt status latched in uSDHC and to stop driving the interrupt signal to the System Interrupt Controller. The host driver must issue a CMD52 to clear the card interrupt. After completion of the card interrupt service, the SDIO Interrupt Status Enable bit is set to 1 and uSDHC starts sampling the interrupt signal again.

See the figure below for an illustration of the SDIO card interrupt scheme and for the sequences of software and hardware events that take place during a card interrupt handling procedure.

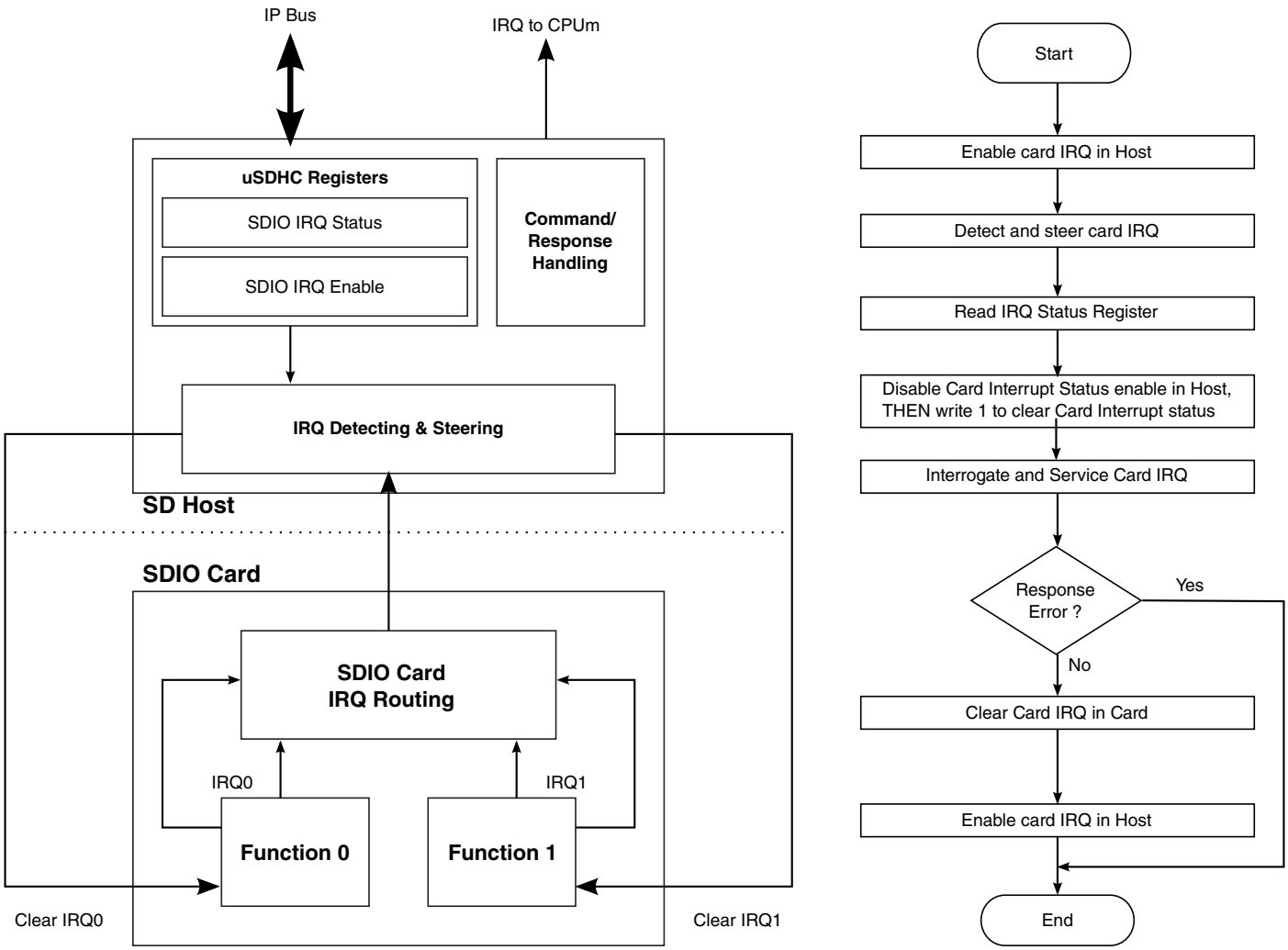


Figure 23-15. Card interrupt scheme, card interrupt detection, and handling procedure

23.6.7 Card insertion and removal detection

The uSDHC module uses either the DATA3 pin or the CD_B pin to detect card insertion or removal. When there is no card on the MMC/SD bus, the DATA3 is pulled to a low voltage level by default.

When any card is inserted to or removed from the socket, uSDHC detects the logic value changes on the DATA3 pin and generates an interrupt. When the DATA3 pin is not used for card detection (for example, it is implemented in GPIO), the CD_B pin must be connected for card detection. Whether DATA3 is configured for card detection or not, the CD_B pin is always a reference for card detection. Whether the DATA3 pin or the CD_B pin is used to detect card insertion, uSDHC sends an interrupt (if enabled) to inform the Host system that a card is inserted.

23.6.8 Power management and wakeup events

When there is no operation between uSDHC and the card through the SD bus, the user can completely disable the peripheral clock and base clock in the chip-level clock control module to save power. When the user needs to use uSDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to uSDHC are disabled, for instance, when the system is in low-power mode, there are some events for which the user needs to enable the clock and handle the event. These events are called wakeup interrupts. The uSDHC module can generate these interrupts even when there are no clocks enabled. The three interrupts that can be used as wakeup events are these:

- Card removal interrupt
- Card insertion interrupt
- Interrupt from SDIO card

The uSDHC module offers a power management feature. By clearing the clock enabled bits in the System Control register, the clocks are gated in the low position to uSDHC. For maximum power saving, the user can disable all the clocks to uSDHC when there is no operation in progress.

These three wakeup events (or wakeup interrupts) can also be used to wakeup the system from low-power modes.

NOTE

To make the interrupt a wakeup event, when all the clocks to uSDHC are disabled or when the entire system is in low power mode, the corresponding wakeup enabled bit needs to be set. Refer to Protocol Control (PROT_CTRL) for more information on the uSDHC Protocol Control register.

23.6.8.1 Setting wakeup events

For uSDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters the sleep mode.

Before the software disables the host clock, it should ensure that all the following conditions have been met:

- No read or write transfer is active
- Data and command lines are not active

- No interrupts are pending
- Internal data buffer is empty

23.6.9 MMC fast boot

The Embedded MultiMediaCard (eMMC4.3) specification adds a fast boot feature that requires hardware support. There are two types of fast boot modes in the eMMC4.3 specification: boot operation and alternative boot operation in the eMMC4.3 specification. Each type also has with-acknowledge and without-acknowledge modes.

In the boot operation mode, the master (MultiMediaCard host) can read boot data from the slave (MMC device) by keeping CMD line low after power-on, or sending CMD0 with argument + 0xFFFFFFFA (optional for slave), before issuing CMD1.

NOTE

For the eMMC4.3 card setting, please see the eMMC4.3 specification.

23.6.9.1 Boot operation

If the CMD line is held low for 74 clock cycles and more after power-up before the first command is issued, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after the CMD line goes low, the slave starts to send the first boot data to the master on the DATA line(s). The master must keep the CMD line low to read all of the boot data.

NOTE

For the purposes of this documentation, fast boot is called "normal fast boot mode".

If boot acknowledge is enabled, the slave has to send acknowledge pattern '010' to the master within 50ms after the CMD line goes low. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode with the CMD line high.

The boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.

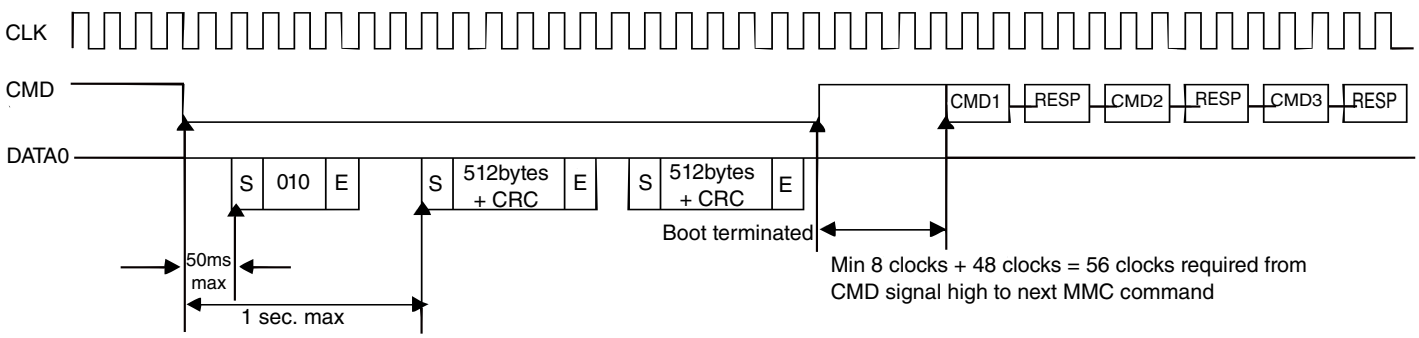


Figure 23-16. MultiMediaCard state diagram (normal boot mode)

23.6.9.2 Alternative boot operation

This boot function is optional for the device. If bit 0 in the extended CSD byte[228] is set to '1', the device supports the alternative boot operation.

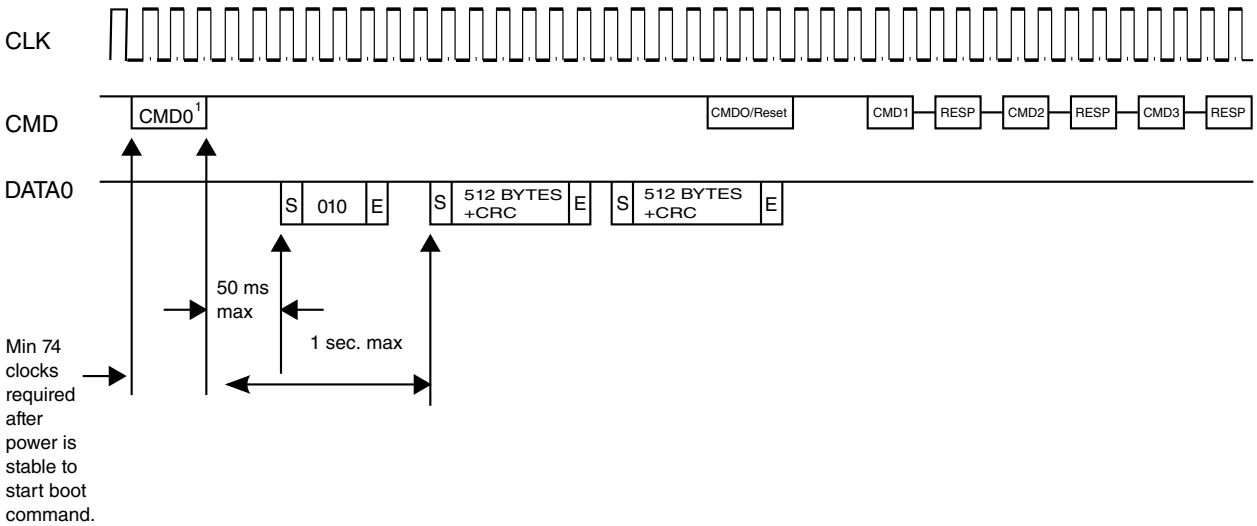
After power-up, if the host issues CMD0 with the argument of 0xFFFFFFFA after 74 clock cycles, before CMD1 is issued or the CMD line goes low, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after CMD0 with the argument of 0xFFFFFFFA is issued, the slave starts to send the first boot data to the master on the DATA line(s).

If boot acknowledge is enabled, the slave has to send the acknowledge pattern '010' to the master within 50ms after the CMD0 with the argument of 0xFFFFFFFA is received. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode by issuing CMD0 (Reset).

Boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.



NOTE 1. CMD0 with argument 0xFFFFFFFA

Figure 23-17. MultiMediaCard state diagram (alternative boot mode)

23.7 Initialization/application of uSDHC

All communication between the system and cards are controlled by the host. The host sends commands of two types: broadcast and addressed (point-to-point).

Broadcast commands are intended for all cards, such as GO_IDLE_STATE, SEND_OP_COND, and ALL_SEND_CID. In the Broadcast mode, all cards are in the open-drain mode to avoid bus contention. Refer to Commands for MMC/SD/SDIO for the commands of bc and bcr categories.

After the broadcast command CMD3 is issued, the cards enter the standby mode. Addressed type commands are used from this point. In this mode, the CMD/DATA I/O pads turns to the push-pull mode to have the driving capability for maximum frequency operation. Refer to Commands for MMC/SD/SDIO for the commands of ac and adtc categories.

23.7.1 Command send & response receive basic operation

Assuming that the data type WORD is an unsigned 32-bit integer, the flow indicated below presents a guideline for sending a command to the card(s):

```
send_command(cmd_index, cmd_arg, other requirements)
{
WORD wCmd; // 32-bit integer to make up the data to write into Transfer Type register, it is
recommended to implement in a bit-field manner
wCmd = (<cmd_index> & 0x3f) >> 24; // set the first 8 bits as '00'+<cmd_index>
set CMDTYP, DPSEL, CICEN, CCCEN, RSTTYP, DTDSEL accorind to the command index;
if (internal DMA is used) wCmd |= 0x1;
if (multi-block transfer) {
          set MSBSEL bit;
          if (finite block number) {
                    set BCEN bit;
                    if (auto12 command is to use) set AC12EN bit;
          }
}
write_reg(CMDARG, <cmd_arg>); // configure the command argument
write_reg(XFERTYP, wCmd); // set Transfer Type register as wCmd value to issue the command
}
wait_for_response(cmd_index)
{
while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set
read IRQ Status register and check if any error bits about Command are set
if (any error bits are set) report error;
write 1 to clear CC bit and all Command Error bits;
}
```

For the sake of simplicity, the function wait_for_response is implemented here by means of polling. For an effective and formal way, the response is usually checked after the Command Complete Interrupt is received. When doing this, make sure that the corresponding interrupt status bits are enabled.

For some scenarios, the response time-out is expected. For instance, after all cards respond to CMD3 and move to the Standby state no response to the Host when CMD2 is sent. The host driver deals with "fake" errors like this with caution.

23.7.2 Card identification mode

When a card is inserted to the socket or the card is reset by the host, the host needs to validate the operation voltage range, identify the cards, request the cards to publish the Relative Card Address (RCA) or to set the RCA for MMC cards.

23.7.2.1 Card detect

See the figure below for a flow diagram showing the detection of MMC, SD, and SDIO cards using uSDHC.

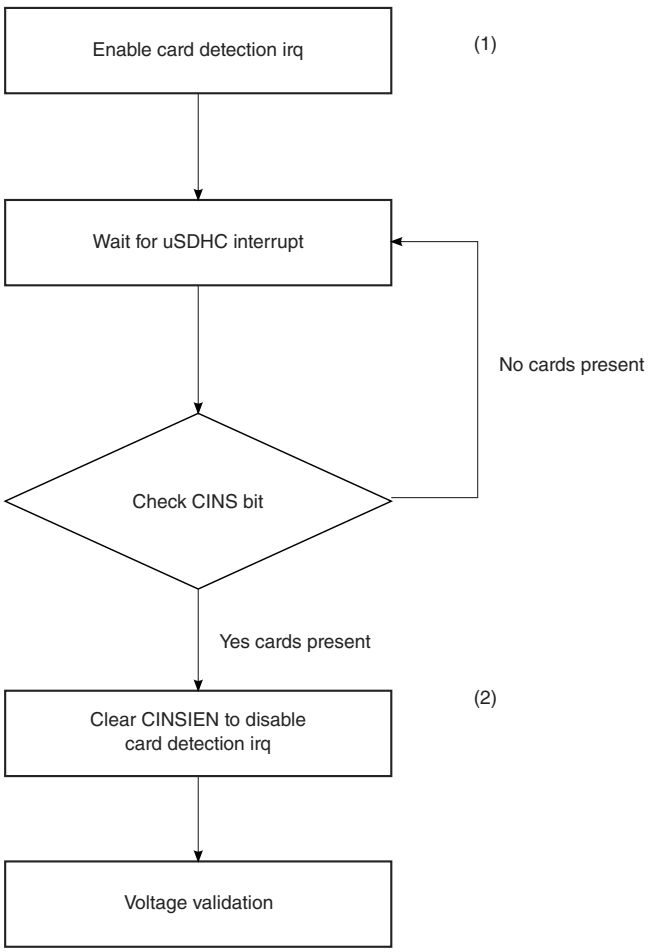


Figure 23-18. Flow diagram for card detection

Here is the card detect sequence:

- Set the CINSIEN bit to enable card detection interrupt.
- When an interrupt from uSDHC is received, check the CINS bit in the Interrupt Status register to see if it was caused by card insertion.
- Clear the CINSIEN bit to disable the card detection interrupt and ignore all card insertion interrupts afterwards.

23.7.2.2 Reset

The host consists of three types of resets:

- Hardware reset (Card and Host) that is driven by Power On Reset (POR).

- Software reset (Host only) is initiated by the write operation on the RSTD, RSTC, or RSTA bits of the System Control register to reset the data part, command part, or all parts of the host controller, respectively.
- Card reset (Card only): The command, "Go_Idle_State" (CMD0), is the software reset command for all types of MMC cards and SD memory cards. This command sets each card into the idle state regardless of the current card state. For an SD I/O card, CMD52 is used to write an I/O reset in CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host needs to validate the voltage range of the card. See the figure below for the software flow to reset both uSDHC and the card.

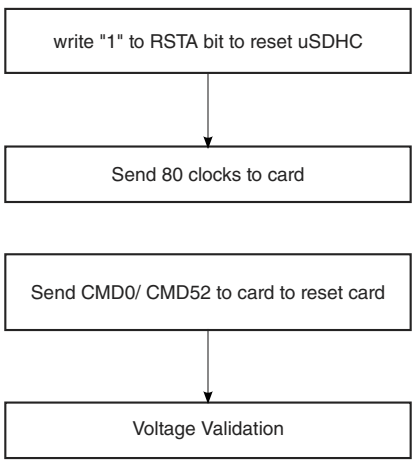


Figure 23-19. Flow chart for resetting uSDHC and SD I/O card

```
software_reset()
{
set_bit(SYSCTRL, RSTA); // software reset the Host
set DTOCV and SDCLKFS bit fields to get the CLK of frequency around 400kHz
configure IO pad to set the power voltage of external card to around 3.0V
poll bits CIHB and CDIHB bits of PRSSTAT to wait both bits are cleared
set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
send_command(CMD_GO_IDLE_STATE, <other parameters>); // reset the card with CMD0
or send_command(CMD_IO_RW_DIRECT, <other parameters>);
}
```

23.7.2.3 Voltage validation

All cards should be able to establish communication with the host using any operation voltage in the maximum allowed voltage range specified in the card specification. However, the supported minimum and maximum values for VDD are defined in the Operation Conditions Register (OCR) and may not cover the whole range.

Cards that store the CID and CSD data in the preload memory are only able to communicate this information under data transfer VDD conditions. This means that if the host and card have non-common VDD ranges, the card is neither able to complete the identification cycle nor able to send CSD data.

Therefore, special commands Send_Op_Cont (CMD1 for MMC), SD_Send_Op_Cont (ACMD41 for SD Memory), and IO_Send_Op_Cont (CMD5 for SD I/O) are used. The voltage validation procedure is designed to provide a mechanism to identify and reject cards that do not match the VDD range(s) desired by the host. This is accomplished when the host sends the desired VDD voltage window as the operand of this command. Cards that cannot perform the data transfer in the specified range must discard themselves from further bus operations and go into the Inactive state. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the Inactive state. This query should be used if the host is able to select a common voltage range or if a notification is sent to the system when a non-usable card in the stack is detected.

The following steps show how to perform voltage validation when a card is inserted:

```
voltage_validation(voltage_range_arguement)
{
label the card as UNKNOWN;
send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO
operation voltage, command argument is zero
if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
          if (0 < number of IO functions) {
                    label the card as SDIO;
                    IORDY = 0;
                    while (!(IORDY in IO OCR response)) { // set voltage range for each IO
function
                              send_command(IO_SEND_OP_COND, <voltage range>, <other
parameter>);
                              wait_for_response(IO_SEND_OP_COND);
                    } // end of while ...
          } // end of if (0 < ...
          if (memory part is present inside SDIO card) Label the card as SDCombo; // this is
an
SD-Combo card
} // end of if (RESP_TIMEOUT ...
if (the card is labelled as SDIO card) return; // card type is identified and voltage range
is
set, so exit the function;
send_command(APP_CMD, 0x0, <other parameters are omitted>); // CMD55, Application specific
CMD
prefix
if (no error calling wait_for_response(APP_CMD, <...>) { // CMD55 is accepted
          send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage
range
for memory part or SD card
          wait_for_response(SD_APP_OP_COND); // voltage range is set
          if (card type is UNKNOWN) label the card as SD;
          return; //
} // end of if (no error ...
else if (errors other than time-out occur) { // command/response pair is corrupted
          deal with it by program specific manner;
} // of else if (response time-out
else { // CMD55 is refuse, it must be MMC card  if (card is already labelled as SDCombo)
{ //
change label
```

```
                    re-label the card as SDIO;
                    ignore the error or report it;
                    return; // card is identified as SDIO card
          } // of if (card is ...
          send_command(SEND_OP_COND, <voltage range>, <...>);
          if (RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted,
either
                    label the card as UNKNOWN;
                    return;
          } // of if (RESP_TIMEOUT ...
} // of else
}
```

23.7.2.4 Card registry

Card registry for the MMC and SD/SDIO/SD combo cards are different. For the SD card, the identification process starts at a clock rate lower than 400 kHz and the power voltage higher than 2.7 V (as defined by the card spec). Currently, the CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated, the host requests the card to send their valid operation conditions.

The response to ACMD41 is the operation condition register of the card. The same command is sent to all the new cards in the system. Incompatible cards are put into the Inactive state. The host then issues the command, All_Send_CID (CMD2), to each card to get its unique card identification (CID) number. Cards that are currently unidentified (in the Ready state), send their CID number as the response. After the CID is sent by the card, the card goes into the Identification state.

The host then issues Send_Relative_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. This RCA is used to address the card for future data transfer operations. After the RCA is received, the card changes its state to the Standby state. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send_Relative_Addr command to the card. The last published RCA is the actual RCA of the card.

The host repeats the identification process with CMD2 and CMD3 for each card in the system until the last CMD2 gets no response from any of the cards in the system.

For MMC operation, the host starts the card identification process in the open-drain mode with the identification clock rate lower than 400 kHz and the power voltage higher than 2.7 V. The open drain driver stages on the CMD line to allow parallel card operation during card identification. After the bus is activated, the host requests the cards to send their valid operation conditions (CMD1). The response to CMD1 is the "wired OR" operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the Inactive state. The host then issues the broadcast command All_Send_CID (CMD2), asking all cards for their unique card identification (CID) number. All

unidentified cards (the cards in the Ready state) simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop sending their CID immediately and must wait for the next identification cycle. As the CID is unique for each card, only one card can be successfully sent its full CID to the host. This card then goes into the Identification state. Thereafter, the host issues Set_Relative_Addr (CMD3) to assign a relative card address (RCA) to the card. After the RCA is received, the state of the card changes to standby, and the card does not react in further identification cycles. Also, its output driver switches from open-drain to push-pull. The host repeats the process, mainly CMD2 and CMD3, until the host receives a time-out condition to recognize the completion of the identification process.

The following steps show how to perform an operation using MMC cards:

```
card_registry()
{
do { // decide RCA for each card until response time-out
          if(card is labelled as SDCombo or SDIO) { // for SDIO card like device
                    send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO card to
publish its
RCA
                    retrieve RCA from response;
          } // end if (card is labelled as SDCombo ...
          else if (card is labelled as SD) { // for SD card
                    send_command(ALL_SEND_CID, <...>);
                    if (RESP_TIMEOUT == wait_for_response(ALL_SEND_CID)) break;
                    send_command(SET_RELATIVE_ADDR, <...>);
                    retrieve RCA from response;
          } // else if (card is labelled as SD ...
          else if (card is labelled as MMC) {
                    send_command(ALL_SEND_CID, <...>);
                    rca = 0x1; // arbitrarily set RCA, 1 here for example
                    send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper
16
bits
          } // end of else if (card is labelled as MMC ...
} while (response is not time-out);
}
```

23.7.3 Card access

Information about Block Write, Block Read, Suspense Resume, ADMA Usage, Transfer Error, and Card Interrupt are detailed in the sections below.

23.7.3.1 Block write

Information on Normal Write, DDR Write, and Write with Pause are detailed in the sections below.

23.7.3.1.1 Normal Write

During a block write (CMD24 - 27, CMD60, CMD61), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. If the CRC fails, the card indicates that the failure on the DATA line. The transferred data is discarded and not written, and all further transmitted blocks (in multiple block write mode) are ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR error bit in the status register, and while ignoring all further data transfer, waits in the Receive-data-State for a stop command. The write operation is also aborted if the host tries to write over a write-protected area.

For MMC and SD cards, programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card reports an error and does not change any register contents.

For all types of cards, some may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the DATA line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) or other means for SDIO cards at any time, and the card responds with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby state and release the DATA line without interrupting the write operation. When re-selecting the card, it reactivates the busy indication by pulling data to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card incorporates the internal DMA and the write operation is a multi-block write with the Auto CMD12 enabled. For the other methods (CPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to write to a card is described below:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:

- For SD/MMC cards, use SET_BLOCKLEN (CMD16).
- For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).

3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

23.7.3.1.2 DDR write

uSDHC supports the dual data rate mode.

The software flow to write to a card in the DDR mode is described as below:

1. Check the card status and wait until the card is ready for data.
2. Set the uSDHC number block register (NOB), where nob is 5, for instance.
3. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
4. Wait for the Transfer Complete interrupt.
5. Check the status bit to see if a write CRC error occurred or another error that occurred during the auto12 command sending and response receiving.

23.7.3.1.3 Write with Pause

The write operation can be paused during the transfer. Instead of stopping the CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the Stop At Block Gap Request (SABGREQ) bit in the Protocol Control register to pause the transfer between the data blocks. As there is no time-out condition in a write operation during the data blocks, a write to all types of cards can be paused in this way, and if the DATA0 line is not required to de-assert to release the busy state, no suspend command is needed.

Similar to the flow described in Normal Write, the write with pause is shown with the same kind of write operation:

1. Check the card status and wait until the card is ready for data.

2. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Set the SABGREQ bit.
7. Wait for the Transfer Complete interrupt.
8. Clear the SABGREQ bit.
9. Check the status bit to see if a write CRC error occurred.
10. Set the CREQ bit to continue the write operation.
11. Wait for the Transfer Complete interrupt.
12. Check the status bit to see if a write CRC error occurred or some another error that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of the BLKCNT field in the Block Attribute register. As the data transfer and the setting of the SABGREQ bit are concurrent, and the delay of register read and the register setting, the actual number of blocks left may not be exactly the value read earlier. The host driver reads the value of BLKCNT after the transfer is paused and the Transfer Complete interrupt is received.

It is also possible that the last block has begun when the Stop At Block Gap Request is sent to the buffer. In this case, the next block gap is the end of the transfer. These types of requests are ignored, the driver should treat these as a non-pause transfer, and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the host system is not stopped, and the transfer is active until the data buffer is full. Because of this, it is recommended to avoid using the Suspend command for the SDIO card. This is because when such a command is sent, uSDHC interprets the system that switches to another function on the SDIO card and flush the data buffer. uSDHC takes the Resume command as a normal command with data transfer, and it is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set as well as the AC12EN bit. However, the uSDHC module automatically sends a CMD12 to mark the end of the multi-block transfer.

23.7.3.2 Block read

Information about Normal read, DDR read, Read with Pause, and Delay Line (DLL) in Read Path are detailed in the sections below.

23.7.3.2.1 Normal read

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification. A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the Transfer state. For multi blocks read, data blocks are continuously transferred until a stop command is issued.

The software flow to read from a card incorporates the internal DMA and the read operation is a multi-block read with the Auto CMD12 enabled. For the other methods (CPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to read from a card is described below:

1. Check the card status and wait until card is ready for data.
2. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer read ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

23.7.3.2.2 DDR read

The uSDHC module supports dual data rate mode.

The software flow to write to a card in the DDR mode is described below:

1. Check the card status ans wait until the card is ready for data.
2. Set the uSDHC number block register (NOB) where nob is 5, for instance.
3. Disable the buffer write ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
4. Wait for the Transfer Complete interrupt.
5. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

23.7.3.2.3 Read with Pause

The read operation is not generally able to pause. Only the SDIO card (and SDCombo card working under I/O mode) supporting the Read Wait feature can pause during the read operation. If the SDIO card supports Read Wait (SRW bit in CCCR register is 1), the host driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks.

Before setting the SABGREQ bit, ensure that the RWCTL bit in the Protocol Control register is set, otherwise uSDHC does not assert the Read Wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit after the Read Wait capability of the SDIO card is recognized.

Similar to the flow described in Normal read, the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on the SDIO card to confirm that the card supports the Read Wait mode.
2. Set the RWCTL bit.
3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
5. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
6. Set the uSDHC number block register (NOB), where nob is 5, for instance.
7. Disable the buffer read ready interrupt, configure the DMA setting, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
8. Set the SABGREQ bit.
9. Wait for the Transfer Complete interrupt.
10. Clear the SABGREQ bit.

11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the Transfer Complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

Similar to the Write operation, it is possible to meet the ending block of the transfer when paused. In this case, uSDHC ignores the Stop At Block Gap Request and treats it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause is transferred to the host system. No matter if the Suspend command is sent or not, the internal data buffer is not flushed.

If the Suspend command is sent and the transfer is later resumed by means of a Resume command, uSDHC takes the command as a normal one accompanied with data transfer. It is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set, as well as the AC12EN bit. However, the uSDHC automatically sends CMD12 to mark the end of multi-block transfer.

23.7.3.2.4 Delay Line (DLL) in Read Path

The DLL is newly added to assist in sampling read data. The DLL provides the ability to programmatically select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT).

The reasons why DLL is needed for uSDHC are these:

- The path of read data traveling from card to host varies.
- In the SD/MMC DDR mode, the minimum input setup and hold time are both at 2.5 ns.

The data sampling window is so small that the delay of loopback clock needs to be accurate and consistent regardless of PVT. The DLL takes the divided card_clk as the reference clock and loopback clock as the input clock. It then generates a delayed version of the input clock according to the programmed target delay.

The DLL can be disabled or bypassed, and it can also be manually set for a fixed delay in the override mode. The override value set is the number of delay cells. In the override mode, there is no need to set the DLL_enable. Another DLL mode is target value mode. In this mode, the DLL automatically adjusts the number of delay cells according to the target value set by the user and PVT changes. Be aware that the target value is in units of

1/32 of the clock reference period. If the card_clk is 100Mhz, then the reference clock period is 10ns; setting target value of 16 means 5ns =(16/32)*10ns. The software can disable automatic update by the setting dll_gate_update bit.

As the user may change the frequency of card_clk from time to time by changing SDCLKFS[7:0]/DVS[3:0], the software must adjust the delay value to ensure it works correctly when the reference clock (card_clk) is changed. The following is the correct flow, which should be ignored if DLL_CTRL_ENABLE is not set.

Step 1: Set the DLL_CTRL_RESET bit

Step 2: Configure the SDCLKFS[7:0] and DVS[3:0]

Step 3: Wait until SDSTB is asserted

Step 4: Clear the DLL_CTRL_RESET bit

Step 5: Wait until both the DLL_STS_SLV_LOCK and DLL_STS_REF_LOCK are asserted

Step 6: Set the DLL_CTRL_SLV_FORCE_UPD

Step 7: Clear the DLL_CTRL_SLV_FORCE_UPD

NOTE

The software should make sure that the DLL_CTRL_SLV_FORCE_UPD lasts for at least one card_clk. So, the software may need to add some delay between step 6 and step 7.

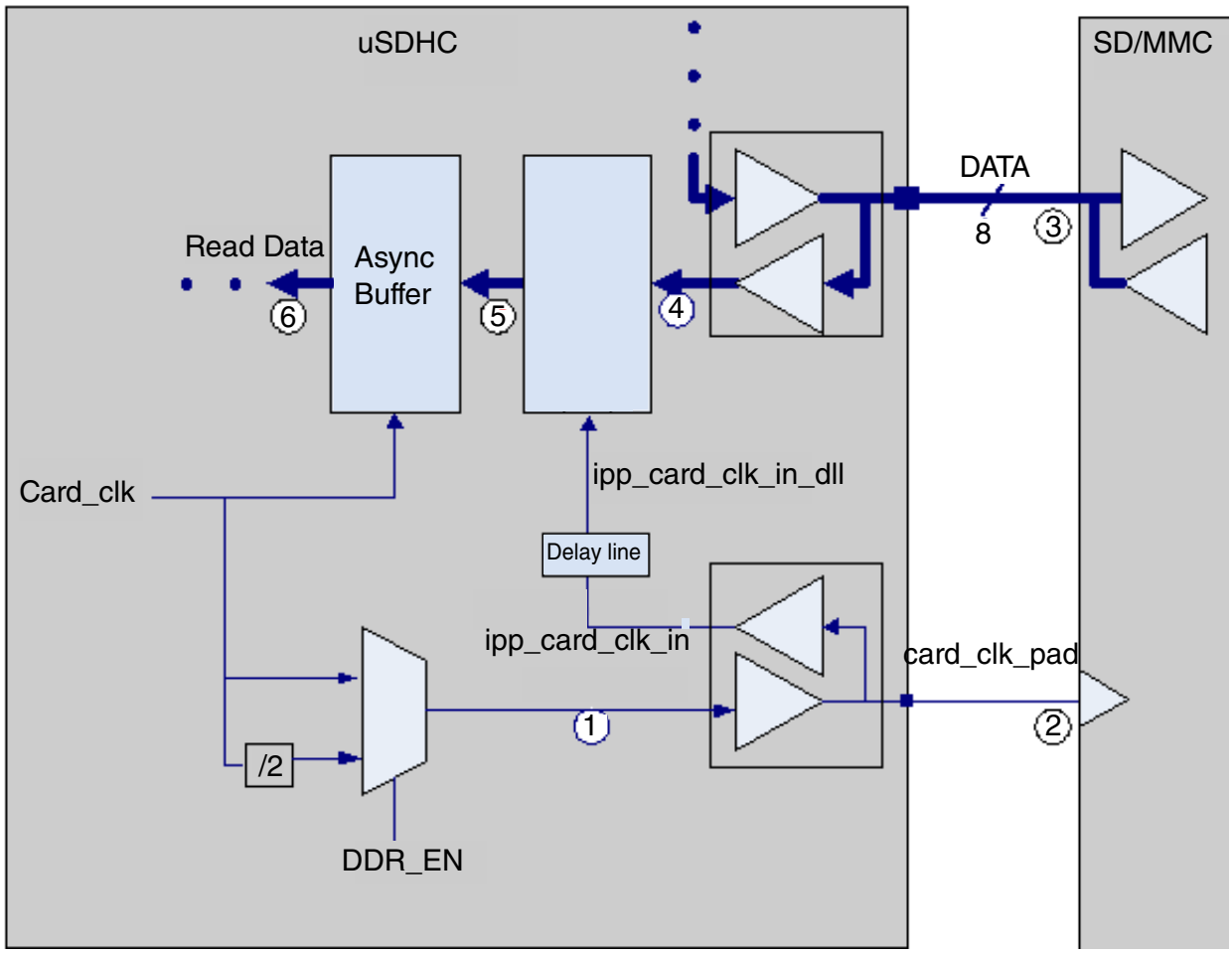


Figure 23-20. DLL in read path

23.7.3.3 Suspend Resume

The uSDHC module supports the Suspend Resume operations of SDIO cards, although slightly different than the suggested implementation of Suspend in the SDIO card specification.

23.7.3.3.1 Suspend

After setting the SABGREQ bit, the host driver may send a Suspend command to switch to another function of the SDIO card. The uSDHC module does not monitor the content of the response, so it does not know if the Suspend command succeeded or not. Accordingly, it does not de-assert Read Wait for read pause. To solve this problem, the host driver does not mark the Suspend command as "Suspend", (that is, setting the CMDTYP bits to 01). Instead, the driver sends this command as if it were a normal

command, and only when the command succeeds, and the BS bit is set in the response, can the Driver send another command marked as "Suspend" to inform uSDHC that the current transfer is suspended. Here is the sequence for the Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, send the Suspend command to suspend the active function. The CMDTYP bit field must be 2'b00.
3. Check the BS bit of the CCCR in the response. If it is 1, repeat this step until the BS bit is cleared or abandon the suspend operation according to the Driver strategy.
4. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 2'b01, so uSDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, uSDHC stops driving DATA2 and goes to the idle state.
5. Save the context registers in the system memory for later use, including the DMA System Address Register (for internal DMA operation), and the Block Attribute Register.
6. Begin operation for another function on the SDIO card.

23.7.3.3.2 Resume

To resume the data transfer, a Resume command is issued:

1. To resume the suspended function, restore the context register with the saved value in step #5 of the Suspend operation.
2. Send the Resume command: In the Transfer Type register, all the bit fields are set to the value as if this were another ordinary data transfer instead of a transfer resume (except the CMDTYP is set to 2'b10).
3. If the Resume command has responded, the data transfer is resumed.

23.7.3.4 ADMA usage

To use the ADMA in a data transfer, the host driver must prepare the correct descriptor chain prior to sending the read/write command.

The steps to prepare the correct descriptor chain are these:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length should be even numbers of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4KB address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1 ~ 3.

4. Repeat steps 1 ~ 3 until all descriptors are created.
5. In the last descriptor, set the End flag to 1 and make sure the total length of all descriptors match the product of the block size and block number configured in the Block Attribute Register.
6. Set the ADMA System Address Register to the address of the first descriptor and set the DMAS field in the Protocol Control Register to 01 to select the ADMA.
7. Issue a write or read command with the DMAEN bit set to 1 in the Transfer Type Register.

Steps 1 ~ 5 are independent of step 6, so step 6 can finish before steps 1 ~ 5. Regarding the descriptor configuration, it is recommended not to use the Link descriptor as it requires extra system memory access.

23.7.3.5 Transfer error

Information about CRC, Internal DMA, Transfer ADMA, and Auto CMD12 errors are detailed in the sections below.

23.7.3.5.1 CRC error

It is possible at the end of a block transfer that a write CRC status error or read CRC error occurs. For this type of error, the latest block received is discarded. This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one.

For a multi-block transfer, the host driver issues a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the AC12EN and BCEND bits are set, uSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 is sent by uSDHC. In this case, the host driver re-sends or re-obtains the last block with a single block transfer.

23.7.3.5.2 Internal DMA error

During the data transfer with internal Simple DMA, if the DMA engine encounters an error on the AHB bus, the DMA operation is aborted, and the DMA error interrupt is sent to the host system. When acknowledged by such an interrupt, the host driver calculates the start address of data block in which the error occurs.

The start address can be calculated by either:

- Reading the DMA System Address register: The error occurs during the previous burst. Considering the block size, the previous burst length and the start address of

the next burst transfer, it is straight forward to obtain the start address of the corrupted block.
- Reading the BLKCNT field of the Block Attribute register: Considering the number of blocks left, the total number to transfer, the start address of transfer, and the size of each block, the start address of corrupted block can be determined. When the BCEN bit is not set, the contents of the Block Attribute register do not change, so this method does not work.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and restart the transfer from the corrupted block to recover from the error.

23.7.3.5.3 Transfer ADMA error

There are three kinds of possible ADMA errors: The AHB transfer, invalid descriptor, and data-length mismatch. Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set.

For acknowledging the status, the host driver should recover the error as shown below and re-transfer from the place of interruption.

- AHB transfer error: Such errors may occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and re-transfer the block that was corrupted, or to the next block if no block is corrupted.
- Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and recreate the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
- Data-length mismatch error: It is similar to recover from this error. The Host Driver polls relating registers to retrieve the transfer context, apply a reset for the data part, configure a new descriptor chain and make another transfer if there is data left. Like the previous scenario of the invalid descriptor error, the data length must match the new transfer.

23.7.3.5.4 Auto CMD12 error

After the last block of the multi-block transfer is sent or received, and the AC12EN bit is set when the data transfer is initiated by the data command, uSDHC automatically sends a CMD12 to the card to stop the transfer.

When errors with this command occur, it is recommended to the host driver to deal with the situations in the following manner:

- Auto CMD12 response time-out: It is not certain whether the command is accepted by the card or not. The host driver clears the auto CMD12 error status bits and re-send CMD12 until it is accepted by the card.
- Auto CMD12 response CRC error: As the card responds to CMD12, it aborts the transfer. The host driver may ignore the error and clear the error status bit.
- Auto CMD12 conflict error or not sent: The command is not sent; therefore, the host driver sends a CMD12 manually.

23.7.3.6 Card interrupt

The external cards can inform the host controller by means of some special signals. For the SDIO card, it can be the low-level on the DATA1 line during some special period. The uSDHC module only monitors the DATA1 line and supports the SDIO interrupt.

When the SDIO interrupt is captured by uSDHC, and the host system is informed by uSDHC asserting the uSDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt source is controlled by the external card, the interrupt from the SDIO card must be serviced before the CINT bit is cleared by written. Refer to Card interrupt handling for the card interrupt handling flow.

23.7.4 Switch function

A switch command is issued by the host driver to enable new features added to the SD/MMC spec. The SD/MMC cards can transfer data at bus widths other than 1-bit. Different speed modes are also defined. To enable these features, a switch command is issued by the host driver.

For SDIO cards, the high-speed mode/DDR50/SDR50/SDR104 are enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed. For SD cards, the high-speed mode/DDR50/SDR50/SDR104 are queried and enabled by a CMD6 (with the mnemonic symbol as SWITCH_FUNC). For MMC cards, the high-speed mode/HS200/HS400 are queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The SDR4-bit, SDR8-bit, DDR4-bit, and DDR8-bit width of MMC is also enabled by the SWITCH command, but with a different argument.

These new functions can also be disabled by a software reset. For SDIO cards, it can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring to the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

For the sake of simplicity, the following pseudocode examples do not show current capability check, which is recommended in the function switch process.

23.7.4.1 Query, enable, and disable SDIO high-speed mode

```
enable_sdio_high_speed_mode(void)
{
send CMD52 to query bit SHS at address 0x13;
if (SHS bit is '0') report the SDIO card does not support high speed mode and return;
send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is
cleared;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

23.7.4.2 Query, enable, and disable SD high-speed mode/DDR50/ SDR50/SDR104

```
enable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0xFFFFFx and read 64 bytes of data accompanying the R1 response;
(high speed mode,x=1; SDR50,x=2; SDR104,x=3; DDR50,x=4;)
wait data transfer done bit is set;
check if the bit x of received 512 bits is set;
if (bit 401 is '0') report the SD card does not support high speed mode and return;
if (bit 402 is '0') report the SD card does not support SDR50 mode and return;
if (bit 403 is '0') report the SD card does not support SDR104 mode and return;
if (bit 404 is '0') report the SD card does not support DDR50 mode and return;
send CMD6, with argument 0x80FFFFFx and read 64 bytes of data accompanying the R1 response;
(high speed mode,x=1; SDR50,x=2; SDR104 x=3; DDR50 x=4;)
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 50MHz for high speed mode, 100MHz for SDR50, 200MHz for SDR104, 50MHz for
DDR50;
(data transactions like normal peers)
}
disable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0x80FFFFF0 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
```

```
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

23.7.4.3 Query, enable, and disable MMC high-speed mode

```
enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support high speed mode and
return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
extract the value of CARD_TYPE field to check the 'high speed mode' in this MMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report MMC switching to high speed mode failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}
```

23.7.4.4 Set MMC bus width

```
change_mmc_bus_width(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support multiple bit width and
return;
send CMD6 with argument 0x3B70x00; (8-bit(dual data rate), x=6; 4-bit(dual data rate), x=5;8-
bit, x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}
```

23.7.5 ADMA operation

Here are the codes for the ADMA1 and ADMA2 operations.

23.7.5.1 ADMA1 operation

```
Set_adma1_descriptor
{
if (to start data transfer) {
// Make sure the address is 4KB align.
Set 'Set' type descriptor;
{
Set Act bits to 01;
Set [31:12] bits data length (byte unit);
}
Set 'Tran' type descriptor;
{
Set Act bits to 10;
Set [31:12] bits address (4KB align);
}
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to 11;
Set [31:12] bits the next descriptor address (4KB aligned);
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set End bit to 1;
}
if (to generate interrupt for this descriptor) {
Set Int bit to 1;
}
Set Valid bit to 1;
}
```

23.7.5.2 ADMA2 operation

```
Set_adma2_descriptor
{
if (to start data transfer) {
// Make sure the address is a 32-bit boundary (lower 2-bit are always '00').
Set higher 32-bit of descriptor for this data transfer initial address;
Set [31:16] bits data length (byte unit);
Set Act bits to '10';
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to '11';
// Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
Set higher 32-bit of descriptor for the next descriptor address;
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set 'End' bit '1';
}
if (to generate interrupt for this descriptor) {
Set 'Int' bit '1';
}
Set the 'Valid' bit to '1';
}
```

23.7.6 Fast boot operation

23.7.6.1 Normal fast boot flow

Here are the steps of normal fast boot flow:

1. Software must configure the init_active bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. Software must configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 0 (normal fast boot), and bit 4 to select the ack mode. If the data is sent through the DMA mode, the software should configure bit 7 to enable the automatic stop at block gap feature, and configure bit 3-bit 0 to select the ack timeout value according to the SD CLK frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software must configure the Protocol control register to set Data Transfer Width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure the Command Argument Register to set argument if needed (no need in normal fast boot).
6. Software must configure the Transfer Type Register to start the boot process. In normal boot mode, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN retain the default value, where DPSEL bit is set to 1, DTDSEL is set to 1 and MSBSEL is set to 1.
7. DMAEN should be configured as 0 in the polling mode and if BCEN is configured as 1, it is recommended to configure the number of blocks in the Block Attributes Register to the maximum value. If in DDR fast boot mode, DDR_EN needs to be set to 1.
8. When step 6 is configured, the boot process begins. The software needs to poll the data buffer ready status to read the data from the buffer in time. If a boot timeout happens (ack times out or the first data read times out), an interrupt is triggered, and the software must configure MMC Boot Register to bit 6 to 0 to disable boot. This makes CMD high, then after at least 56 clocks, it is ready to begin a normal initialization process.
9. If there is no timeout, software needs to determine when the data read is finished and then configure MMC Boot Register bit 6 to 0 to disable boot. This render CMD line high and command completed asserted. After at least 56 clocks, it is ready to begin the normal initialization process.

10. Reset the host and then can begin the normal process.

23.7.6.2 Alternative fast boot flow

Here are the steps of alternative fast boot flow:

1. Software needs to configure init_active bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software needs to configure MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 1 (alternative boot), and bit 4 to select the ack mode or not. If data needs to be sent through the DMA mode, then configure bit 7 to enable the automatic stop at block gap feature. Software should also configure bit 3-bit 0 to select the ack timeout value according to the SD clock frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in the DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software needs to configure the Protocol control register to set the data transfer width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFA.
6. Software needs to configure the Transfer Type Register to start the boot process by CMD0 with the 0xFFFFFFFA argument. In alternative boot, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. Note DMAEN should be configured as 0 in the polling mode, and if BCEN is configured as 1 in polling mode, it is recommended to configure the block count in the Block Attributes Register to the maximum value. If in the DDR fast boot mode, DDR_EN needs to be set to 1.
7. When step 6 is configured, the boot process begins. Software needs to poll the data buffer ready status to read the data from the buffer in time. If there is a boot timeout (ack data timeout in 50ms or data timeout in 1s), the host sends out the interrupt and the software needs to send CMD0 with reset and then configure the boot enable bit to 0 to stop this process.
8. If there is no time out, the software needs to decide when to stop the boot process, and send out the CMD0 with reset and then after the command is completed, configure the MMC Boot Register bit 6 to stop the process. After 8 clocks from the command completion, the slave (card) is ready for the identification step.
9. Reset the host and then begin the normal process.

23.7.6.3 Fast boot application case (in DMA mode)

In the boot application case, because the image destination and the image size are contained in the beginning of the image, it is necessary to switch DMA parameters on the fly during MMC fast boot.

In fast boot, the host can use Advanced DMA2 (ADMA2) with two destinations.

The detailed flow is described below:

1. The software needs to configure INIT_ACTIVE bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. The software needs to configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot) or 1 (alternative boot); and bit 4 to select the ack mode. In DMA mode, configure bit 7 to 1 to enable the automatic stop at block gap feature. Also configure bits[31-16] to set the (BLK_CNT - VALUE1). Here VALUE1 is the value of the block count that needs to transfer the first time, so that the host stops at the block gap when the uSDHC controller gets VAULE1 blocks from the device. Also, configure bits[3-0] to select the ack timeout value according to the SD clock frequency.
3. The software then needs to configure the Block Attributes Register to set block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes. In DMA mode, it is recommended to set the block count (BLK_CNT) to the max value (16'hffff).
4. The software needs to configure Protocol Control Register to set DTW (data transfer width). If in DDR fast boot mode, the DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software enable ADMA2 by configuring Protocol Control Register bits [9-8].
6. The software needs to set at least three pairs of ADMA2 descriptor in boot memory (that is, in IRAM, at least six words). The first pair descriptor defines the start address (that is, IRAM) and data length (that is,512byte*VALUE1) of the first part boot code. The software also needs to set the second pair descriptor, the second start address (any value that is writable), and data length is suggested to set 1~2word (record as VALUE2). Note that the second couple desc also transfers useful data even at lease 1 word, because our ADMA2 cannot support 0 data_length data transfer descriptor.
7. The software needs to configure Command Argument Register to set argument to 0xFFFFFFFA in alternative fast boot and do not need to be set in normal fast boot.
8. The software needs to configure Transfer Type Register to start the boot process. CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. DMAEN is configured as 1 in the DMA mode. And, if BCEN is configured

as 1, then configure blk no in Bock Attributes Register to the max value. And, if in the DDR fast boot mode, DDR_EN needs to be set to 1.

9. When step 8 is configured, boot process begins, the first VALUE1 block number data gets transferred. The software needs to poll the TC bit (bit1 in Interrupt Status Register) to determine first transfer is ended. Also, the software needs to polling the BGE bit (bit2 in Interrupt Status Register) to determine if the first transfer stops at the block gap.
10. When TC and BGE bits are set to 1, the software can analyze the first code of VALUE1 block, initializes the new memory device, if required, and sets the third pair of descriptors to define the start address and length of the remaining part of the boot code (VALUE3, the remain boot code block). Remember to set the last descriptor with END.
11. The software needs to configure the MMC Boot Register (offset 0xc4) again. Set bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot), to 1 (alternative boot); and bit 4 to select the ack mode or not. In the DMA mode, configure bit 7 to 1 for enabling the automatically stop at block gap feature. Also, configure bit31-bit16 to set the (BLK_CNT - (VALUE1+1+VALUE3)), that host stops at block gap when the uSDHC controller gets (VALUE1+1+VALUE3)) blocks from device totally include the blocks received in step 9. And need to configure bit 3-bit0 to select the ack timeout value according to the sd clk frequency. Note that the software does not need to configure the BLK_CNT again, because it is counted down automatically by the uSDHC controller.
12. The software needs to clear the TC and BGE bits and the software needs to clear SABGREQ (bit 16 in the Protocol control register) and set CREQ (bit17 in the Protocol control register) to 1 to resume the data transfer. Host transfers the VALUE2 and VALUE3 data to the destination that is set by descriptor.
13. The software needs to do poll BGE bit to determine if the fast boot is over.

Note:

1. When ADMA boot flow starts, for uSDHC, it is like a normal ADMA read operation. So, set ADMA2 descriptor as the normal ADMA2 transfer.
2. Need a few words length memory to keep descriptor.
3. For the 1~2-word data in second descriptor setting, it is a useful data, so the software needs to deal the data because of the application case.

23.8 Commands for MMC/SD/SDIO

A table containing the list of commands for the MMC/SD/SDIO cards is provided here.

Refer to the corresponding specifications for more details about the command information.

There are four kinds of commands defined to control the MultiMediaCard:

1. Broadcast commands (bc), no response
2. Broadcast commands with response (bcr), response from all cards simultaneously
3. Addressed (point-to-point) commands (ac), no data transfer on the DATA
4. Addressed (point-to-point) data transfer commands (adtc)

Response: A response is a token that is sent from the card to the host as an answer to a previously received command. A response is transferred serially on the CMD line.

Table 23-14. Commands for MMC/SD/SDIO cards

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all MMC and SD memory cards to idle state.
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all MMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3[1]	ac	[31:6] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/ SEND_RELATIVE_ADDR	Assigns relative address to the card.
CMD4	bc	[31:0] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards.
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO cards in idle state to send them operation conditions register contents in the response on the CMD line.
CMD6[2]	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 ~ 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer to "SD Physical Specification V1.1" for more details.
CMD6[3]	ac	[31:26] Set to 0 [25:24] Access [23:16] Index	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to "The

Table continues on the next page...

Table 23-14. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
		[15:8] Value [7:3] Set to 0 [2:0] Cmd Set			MultiMediaCard System Specification Version 4.0 Final draft 2" for more details.
CMD7	ac	[31:6] RCA [15:0] stuff bits	R1b	SELECT/ DESELECT_CARD	Toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.
CMD9	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	adtc	[31:0] data address	R1	READ_DAT_UNTIL_STOP	Reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:6] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	Reserved				
CMD15	ac	[31:6] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	adtc	[31:0] reserved bits(all 0)	R1	SEND_TUNING_BLOCK	64 bytes tuning pattern is sent for SDR50 and SDR104.
CMD20	adtc	[31:0] data address	R1	WRITE_DAT_UNTIL_STOP	Writes data stream from the host, starting at the given address, until a STOP_TRANSMISION follows.

Table continues on the next page...

Table 23-14. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD21	adtc	[31:0] stuff bits	R1	SEND_TUNING_BLOCK	128 clocks of tuning pattern (64 byte in 4 bit mode or 128 byte in 8 bit mode) is sent for HS200 optimal sampling point detection.
CMD22-23	Reserved				
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command is issued only once per card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31	Reserved				
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase.
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase.

Table continues on the next page...

Table 23-14. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register, and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This command accesses application dependent registers which are not defined in the MMC standard.
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode.
CMD41	Reserved				
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43~51	Reserved				
CMD52	ac	[31:0] stuff bits	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	[31:0] stuff bits	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.
CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather that a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57-59	Reserved				
CMD60	adtc	[31] WR [30:24] stuff bits [23:16] address	R1b	RW_MULTIPLE_REGISTER	These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the

Table continues on the next page...

Table 23-14. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
		[15:8] stuff bits [7:0] byte count			device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 is used to read and write these registers.
CMD61	adtc	[31] WR [30:16] stuff bits [15:0] data unit count	R1b	RW_MULTIPLE_BLOCK	The host issues a RW_MULTIPLE_BLOCK (CMD61) to begin the data transfer.
CMD62-63	Reserved				
ACMD6[4]	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13[4]	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD22[4]	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SECTORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.
ACMD23[4]	ac	[31:23] stuff bits [22:0] Number of blocks	R1	SET_WR_BLK_ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for fast Multiple Block WR command). "1"=default(one write block).
ACMD41[4]	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) contents in the response on the CMD line.
ACMD42[4]	ac	[31:1] stuff bits [0] set_cd	R1	SET_CLR_CARD_DETECT	Connect(1)/Disconnect(0) the 50KOhm pull-up resistor on CD_B/DATA3 of the card.
ACMD51[4]	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

1. CMD3 differs for MMC and SD cards. For MMC cards, it is referred to as SET_RELATIVE_ADDR, with a response type of R1. For SD cards, it is referred to as SEND_RELATIVE_ADDR, with a response type of R6 (with RCA inside).
2. CMD6 differs completely between high speed MMC cards and high-speed SD cards. Command SWITCH_FUNC is for high-speed SD cards.
3. Command SWITCH is for high speed MMC cards. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH_ERROR status bit in the EXT_CSD register is set. The Access Bits are shown in Table 2.
4. ACMDs is preceded with the APP_CMD command. Commands listed are used for SD only, other SD commands not listed are not supported on this module.

The access bits for the EXT_CSD access modes are shown below.

Table 23-15. EXT_CSD access modes

Bits	Access name	Operation

Table continues on the next page...

Table 23-15. EXT_CSD access modes (continued)

00	Command set	The command set is changed according to the Cmd Set field of the argument.
01	Set bits	The bits in the pointed byte are set, according to the bits set to 1 in the Value field.
10	Clear bits	The bits in the pointed byte are cleared, according to the bits set to 1 in the Value field.
11	Write byte	The Value field is written into the pointed byte.

23.9 Software restrictions

23.9.1 Initialization active

The driver cannot set INITA bit in System Control register when any of the command line or data lines are active, so the driver must ensure both CDIHB and CIHB bits are cleared.

23.9.2 Software polling procedure

For polling read or write, after the software begins a buffer read or write, it must access exactly the number of times as the values set in the Watermark Level Register; moreover, if the block size is not a multiple of the value in the Watermark Level Register (read and write respectively), the software must access exactly the remaining number of words at the end of each block.

For example, for a read operation, if the RD_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block number is 2, then the access times for the burst sequence in the whole transfer process must be 4, 4, 2, 4, 4, 2.

23.9.3 Suspend operation

To suspend the data transfer, the software must inform uSDHC that the suspend command is successfully accepted. To achieve this, after the Suspend command is accepted by the SDIO card, software must send another normal command marked as suspend command (CMDTYP bits set as '01') to inform uSDHC that the transfer is suspended.

If software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remaining number of blocks before sending the normal command marked as suspend, otherwise on sending such a 'suspend' command, uSDHC treats the current transfer is aborted and change the BLKCNT register to its original value, instead of retaining the remaining number of blocks.

23.9.4 Data length setting

For either ADMA (ADMA1 or ADMA2) transfer, the data in the data buffer must be word aligned, so the data length set in the descriptor must be a multiple of 4.

23.9.5 (A)DMA address setting

To configure the ADMA1/ADMA2/DMA address register, when TC bit is set, the register always updates itself with the internal address value to support dynamic address synchronization, so the software must ensure that the TC bit is cleared prior to configuring the ADMA1/ADMA2/DMA address register.

23.9.6 Data port access

Data port does not support parallel access. For example, during an internal DMA access, it is not allowed to write any data to the data port by CPU; or during a CPU read operation, it is also prohibited to write any data to the data port, by either CPU or internal DMA. Otherwise the data is corrupted inside the uSDHC buffer.

23.9.7 Change clock frequency

The uSDHC module does not automatically gate off the card clock when the host driver changes the clock frequency. To prevent possible glitch on the card clock, clear the FRC_SDCLK_ON bit when changing the clock divisor value (SDCLKFS or DVS in System Control Register) or setting the RSTA bit.

Also, before changing the clock divisor value, the host driver should make sure that the SDSTB bit is high.

23.9.8 Multi-block read

For pre-defined multi-block read operation, that is, the number of blocks to read has been defined by previous CMD23 for MMC, or pre-defined number of blocks in CMD53 for SDIO/SDCombo, or whatever multi-block read without abort command at card side, an abort command, either automatic or manual CMD12/CMD52, is still required by uSDHC after the pre-defined number of blocks are done, to drive the internal state machine to idle mode.

In this case, the card may not respond to this extra abort command and uSDHC gets response timeout. It is recommended to manually send an abort command with RSPTYP[1:0] both bits cleared.

Chapter 24
Clocking

24.1 Introduction

This chapter details the clock sources, distribution and management within the i.MX 7ULP. These functions are under joint control of the System Clock Generation (SCG) modules, Peripheral Clock Control (PCC) modules, and Core Mode Controller (CMC)[1] blocks.

NOTE

References in this chapter to “Core 0” or “Processor A” correspond to the Cortex M4 core. References in this chapter to “Core 1” or “Processor B” correspond to the Cortex A7 core.

The clocking scheme provides clear separation between M4 domain and A7 domain. Except for a few clock sources shared between two domains, such as the System Oscillator clock, the Slow IRC (SIRC), and the Fast IRC clock (FIRC), clock sources and clock management are separated and contained within each domain.

M4 clock management consists of SCG0, PCC0, PCC1, and CMC0 modules.

A7 clock management consists of SCG1, PCC2, PCC3, and CMC1 modules.

24.2 Clock distribution

The SCG modules generate and distribute clocks on the device. SCG functions include:

- clock reference selection
- generation of clock used to derive processor, system, peripheral bus and external memory interface clocks
- source selection for peripheral clocks
- control of power-saving clock-gating mode

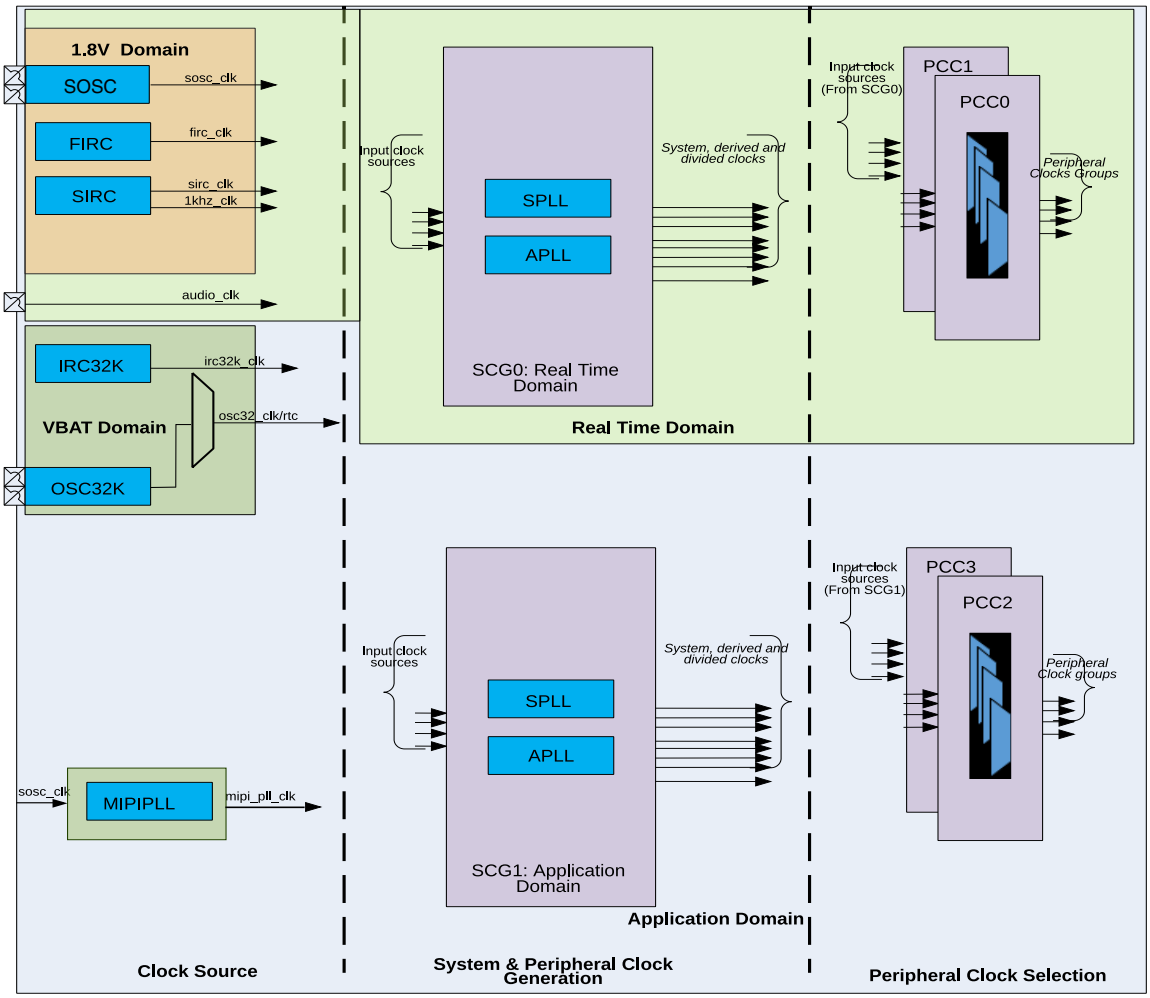
1. MSMC, CMC, and RMC are 3 separate but tightly coupled blocks. The memory map documented in MSMC chapter encompasses features related to all 3 blocks, being actually implemented into the CMC block alone. Therefore, there might be some references to CMC and RMC blocks throughout this document.

PCC modules control clock selection, optional division and clock gating mode for peripherals. PCCs within a domain create two groups of clock options called BUS and PLAT clocks. Therefore, for each domain, four clock selection options are available namely:

- PCC0_BUS, PCC0_PLAT, PCC1_BUS and PCC1_PLAT for M4 domain
- PCC2_BUS, PCC2_PLAT, PCC3_BUS and PCC3_PLAT for the A7 domain.

Details of clock options available for these groups are shown in Table 24-2 and Table 24-3. Among other clocking options, each peripheral is assigned a PCC slot. The peripheral slot and PCC group assignment for each peripheral is shown in Table 24-4. Clocks for peripherals are selected based on PCC slot and sources available to that group. Clock selection for peripherals are shown in Peripheral clocks. PCC slots referred in that section should be inferred from Table 24-4 and Table 24-5.

The following figure shows a high-level illustration of clock sources and clock generation for i.MX 7ULP. Actual implementation of clock sources and clock gating for cores, buses and peripherals are detailed in the SCG chapter.



Notes:
1. Refer SCG section for details for system clock selection
2. Refer PCC section for Peripheral clock selection
3. Refer the Clock source table for clock details

Figure 24-1. Clock overview block diagram

NOTE

- To bypass system oscillator and directly apply clock from pin, SCG_SOSCCFG[EREFS] should be set to 0. The direct clock should be applied on the EXTAL pin.
- For using oscillator reference, SCG_SOSCCSR[SOSCEN] and SCG_SOSCCFG[EREFS] should both be set to 1.

24.3 External clock sources

In normal functional mode, this device operates off two primary external reference clocks: System oscillator clock (SOSC) and RTC oscillator clock (ROSC):

- System oscillator clock is a high frequency reference clock with a frequency in the range of 16 MHz to 32 MHz. This clock is used as a reference clock to the on-chip PLLs which generate all the required high frequency clocks.
- RTC oscillator clock is the 32.768 kHz constant frequency, real-time clock.

24.4 Internal clock sources

This device is capable of generating these internal reference clocks:

- The FIRC is the fast IRC clock with nominal frequency in the range from 48 to 60 MHz. In addition, the FIRC provides a clock selection option for peripherals.
- The SIRC is the slow IRC clock with nominal frequency of 16 MHz. The SIRC provides a clock selection option for peripherals.
- The IRC1K generates 1 kHz clock that is enabled in all modes of operation, including all low power modes.
- The RTC OSC has the capability to provide nominal 32 kHz (not recommended for accurate clock and normal operation) IRC in absence of the external OSC reference clock if the VBAT domain is enabled.

NOTE

The internal oscillator is automatically multiplexed in the clocking system when the system detects a loss of clock. The internal oscillator will provide clocks to the same on-chip modules as the external 32 kHz oscillator. The internal oscillator is not precise relative to a crystal. While it will provide a clock to the system, it generally will not be precise enough for long-term time keeping. The internal oscillator is anticipated to be useful for quicker start-up times and tampering prevention, but should not be used as the exclusive source for the 32 kHz clocks. An external 32 kHz clock source must be used for production systems.

24.5 Clock generation

24.5.1 Oscillators

The system oscillator, in conjunction with an external crystal or resonator, generates a reference clock for the device. The system oscillator module supports 16-32 MHz crystals or resonators. It also provides the option for an external input clock to EXTAL signal directly.

The RTC oscillator is in the VBAT domain. The RTC oscillator module, in conjunction with an external crystal, generates a 32.768 kHz real-time reference clock for the RTC and will always be enabled and supplying clock to SRTC. This is the default clock source.

24.5.2 PLLs

Following are the five PLLs in this device clock structure.

- PLL0 (SPLL) is a fixed-frequency PLL. In addition to the main clock output of 480 MHz, PLL0 (SPLL) also includes 4 Phase Fractional Dividers (PFDs) that can generate other clock frequencies for core, system buses and peripherals. PLL0 is the system PLL for M4 domain.
- PLL1 (APLL) is a FracN PLL. This PLL supports frequency tuning for audio applications. PLL1 (APLL) output clock is dedicated to audio peripherals, and it should be disabled for power saving when audio applications are not in use.
- PLL2 (SPLL) and PLL3 (APLL) are FracN PLLs with additional PFDs. Each PFD has an independently programmable divider value, N, ranging from 12-35, which controls its output frequency. Output clocks of PLL2 (SPLL), PLL3 (APLL) and their PFDs are available clock options for A7, GPU, A7 system buses, DDR and other peripherals in the A7 domain.
- The fifth PLL is the DSI PLL. It is embedded in the DSI PHY block. The output clock of this PLL has been ported to outside and now this PLL can selected as a clock source for other modules as well.

NOTE

- FIRC is not a valid a clock reference to PLLs on i.MX 7ULP. When PLL is enabled, XOSC should be used as PLL clock reference.
- In case of RAW PLLs, although it is possible to change the PLL parameters, these are tuned to 528 MHz for fractional PLLs and 480 MHz for integer PLLs. So, for the fractional

PLLs, MULT*refclk must always be equal to 528 MHz and for the integer PLL MULT*refclk = 480 MHz.
- PLL output frequencies must always be higher than 16 MHz.

24.6 Core, Platform and System Bus clocks

There are several asynchronous system clock domains on this device. The primary domains are:

- A7 Core and complex
- DDR, GPU-3D, GPU-2D, NIC0, NIC1 and System buses
- M4 Core, Platform and System buses

These clocks are generated in similar scheme as shown in the following figures.

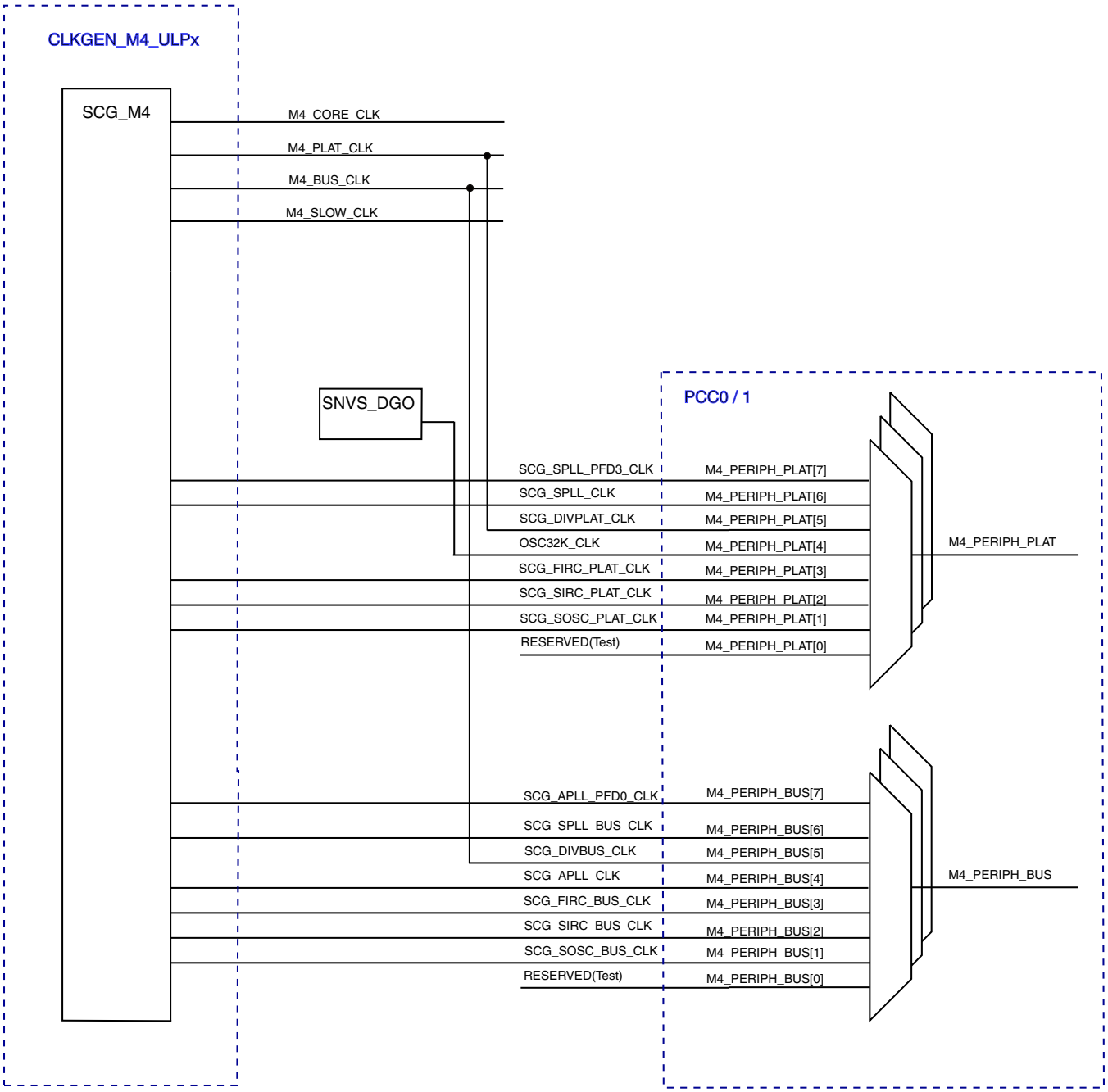


Figure 24-2. System clock generation for M4 domain

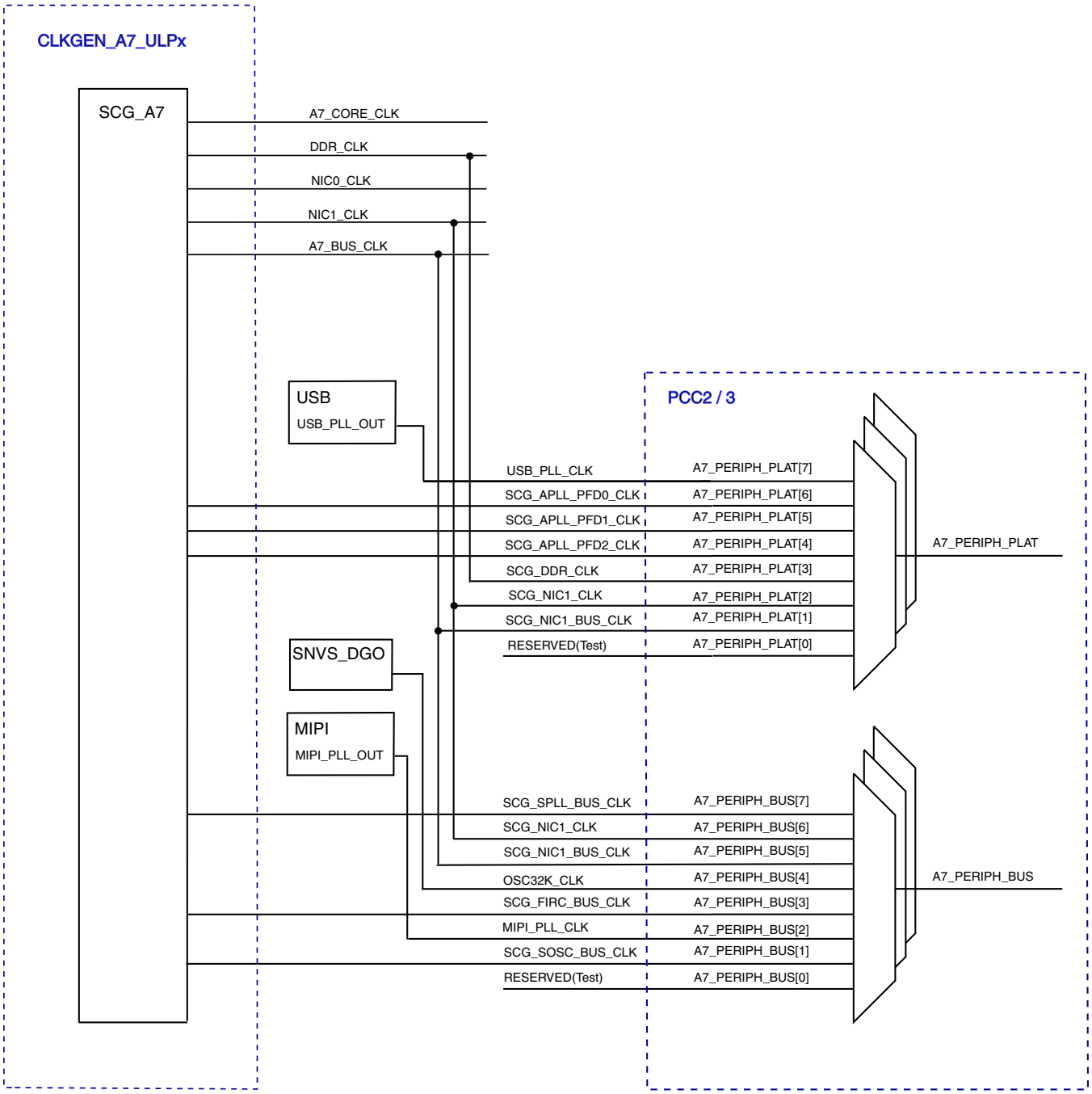


Figure 24-3. System clock generation for A7 domain

NOTE

See Figure 26-2 and Figure 26-3 for details on SCG clocking architecture on M4 and A7 domains.

Table 24-1. Native and Derived clock sources within SCG

S.No	Source	Use/comment
1	sosc_clk	Native clock. Reference for PLL0/1/2/3. Source for A7_SOSC_CLK, M4_SOSC_CLK
2	firc_clk	Native clock. Source for ca7_bus_clk, cm4.firc_bus_clk, ca7.firc_plat_clk, cm4.firc_plat_clk
3	sirc_clk	Native clock. Source for M4_SIRC_CLK
4	cm4.sircdiv1	Derived clock. Source for M4_PERIPH_PLAT PCC mux.
5	cm4.fircdiv1	Derived clock. Source for M4_PERIPH_PLAT PCC mux.
6	cm4.soscdiv1	Derived clock. Source for M4_PERIPH_PLAT PCC mux.
7	cm4.sircdiv2	Derived clock. Source for M4_PERIPH_BUS PCC mux.
8	cm4.fircdiv2	Derived clock. Source for M4_PERIPH_BUS PCC mux.
9	cm4.soscdiv2	Derived clock. Source for M4_PERIPH_BUS PCC mux.
10	ca7.fircdiv2	Derived clock. Source for A7_PERIPH_BUS PCC mux.
11	ca7.soscdiv2	Derived clock. Source for A7_PERIPH_BUS PCC mux.
12	audio_master_clk0	External clock from PTA4 or PTA14. SAI0 MCLK opt1, SAI1 MCLK opt 2
13	audio_master_clk1	External clock from PTA23 or PTB7. SAI1 MCLK opt1, SAI0 MCLK opt 2
14	1khz_clk (from sirc)	Native clock. Source for M4_IRC1K_CLK and A7_IRC1K_CLK
15	osc_32k (rtc_32k)	Native clock. Source for A7_PERIPH_BUS and M4_PERIPH_PLAT PCC muxes. Also source for DGO_OSC32K_CLK, M4_OSC32K_CLK and A7_OSC32K_CLK and RTC_CLKOUT
16	irc_32k	Used within VBAT only
17	cm4.spll_clk	SPLL source clocks. SPLL output that depends on SPLL's parameter (from SCG_SPLLCFG register) to select between PFD and RAW/VCO. Source for CM4_PERIPH_PLAT PCC mux.
18	cm4.spll.pfd0_clk	Derived clock. Not used
19	cm4.spll.pfd1_clk	Derived clock. Not used
20	cm4.spll.pfd2_clk	Derived clock. Not used
21	cm4.spll.pfd3_clk	Derived clock. Source for M4_PERIPH_PLAT PCC mux.
22	cm4.spll_bus_clk	Derived clock. Source for M4_PERIPH_BUS PCC mux.
23	cm4.apll_clk	APLL source clocks. APLL output that depends on APLL's parameter (from SCG_APLLCFG register) to select between PFD and RAW/VCO. Source for CM4_PERIPH_BUS PCC mux
24	cm4.apll.pfd0_clk	Derived clock. Source for M4_PERIPH_BUS PCC mux.
25	cm4.divcore_clk	Derived clock. Source for M4_CORE_CLK
26	cm4.divplat_clk	Derived clock. Source for M4_PERIPH_PLAT PCC mux. Source for M4_PLAT_CLK
27	cm4.divbus_clk	Derived clock. Source for M4_PERIPH_BUS PCC mux. Source for M4_BUS_CLK
28	cm4.divslow_clk	Derived clock. Source for M4_SLOW_CLK

Table continues on the next page...

Table 24-1. Native and Derived clock sources within SCG (continued)

S.No	Source	Use/comment
30	ca7.spll_clk	SPLL source clocks. SPLL output that depends on SPLL's parameter (from SCG_SPLLCFG register) to select between PFD and RAW/VCO.
31	ca7.spll.pfd0_clk	Derived clock. Not used
32	ca7.spll.pfd1_clk	Derived clock. Not used
33	ca7.spll.pfd2_clk	Derived clock. Not used
34	ca7.spll.pfd3_clk	Derived clock. Not used
35	ca7.apll_clk	APLL source clocks. APLL output that depends on APLL's parameter (from SCG_APLLCFG register) to select between PFD and RAW/VCO.
36	ca7.apll.pfd0_clk	Derived clock. Source for A7_PERIPH_PLAT PCC mux.
37	ca7.apll.pfd1_clk	Derived clock. Source for A7_PERIPH_PLAT PCC mux.
38	ca7.apll.pfd2_clk	Derived clock. Source for A7_PERIPH_PLAT PCC mux.
39	ca7.apll.pfd3_clk	Derived clock. Not used
40	ca7.divcore_clk	Derived clock. Source for A7_CORE_CLK
41	ca7.spll_bus_clk	Derived clock. Source for A7_PERIPH_BUS PCC mux.
42	ca7.ddr_clk	Derived clock. Source for DDR_CLK. Source for A7_PERIPH_PLAT PCC mux.
43	ca7.nic0_clk	Derived clock. Source for NIC0_CLK
44	ca7.nic1_clk	Derived clock. Source for NIC1_CLK. Source for A7_PERIPH_PLAT and A7_PERIPH_BUS PCC mux.
45	ca7.nic1_bus_clk	Derived clock. Source for A7_BUS_CLK. Source for A7_PERIPH_PLAT and A7_PERIPH_BUS PCC mux.
46	ca7.nic1_ext_clk[1]	Derived clock. Not used
47	gpu_aclk	Derived clock. Source for GPU_AXI_CLK
48	mipi_pll_clk	Native clock. MIPI/DSI clock. Source for A7_PERIPH_BUS PCC mux.
49	usb_pll_clk	USB PLL RAW/VCO clock. Source for A7_PERIPH_PLAT PCC mux.

1. Though this clock is not implemented, SCG1 (A7) contains NICCCR[NIC1_DIVEXT] and NICCSR[NIC1_DIVEXT] fields to generate FlexBus clock through CLKOUT pin

Table 24-2. Clock sources mapped to different PCC0/1 clock selects PCS

PCS	PCC0[PCS] (PCC0_PLAT)	PCC0[PCS] (PCC0_BUS)[1]	PCC1[PCS] (PCC1_PLAT)	PCC1[PCS] (PCC1_BUS)[1]
7	cm4.spll.pfd3_clk	cm4.apll.pfd0_clk	cm4.spll.pfd3_clk	cm4.apll.pfd0_clk
6	cm4.spll_clk	cm4.spll_bus_clk	cm4.spll_clk	cm4.spll_bus_clk
5	cm4.divplat_clk	cm4.divbus_clk	cm4.divplat_clk	cm4.divbus_clk
4	osc_32k (rtc_32k)	cm4.apll_clk	osc_32k (rtc_32k)	cm4.apll_clk
3	cm4.fircdiv1	cm4.fircdiv2	cm4.fircdiv1	cm4.fircdiv2
2	cm4.sircdiv1	cm4.sircdiv2	cm4.sircdiv1	cm4.sircdiv2
1	cm4.soscdiv1	cm4.soscdiv2	cm4.soscdiv1	cm4.soscdiv2

Table continues on the next page...

Table 24-2. Clock sources mapped to different PCC0/1 clock selects PCS (continued)

PCS	PCC0[PCS] (PCC0_PLAT)	PCC0[PCS] (PCC0_BUS)[1]	PCC1[PCS] (PCC1_PLAT)	PCC1[PCS] (PCC1_BUS)[1]
0	OFF	OFF	OFF	OFF

1. When using SAIs (SAI0: PCC0 channel 55, SAI1: PCC1 channel 42) PCC dividers, the divider input frequency has to be limited to frequencies below 50 MHz.

Table 24-3. Clock sources mapped to different PCC2/PCC3 clock selects PCS

PCS	PCC2[PCS] (PCC2_PLAT)	PCC2[PCS] (PCC2_BUS)	PCC3[PCS] (PCC3_PLAT)	PCC3[PCS] (PCC3_BUS)
7	usb_pll_clk	ca7.spll_bus_clk	usb_pll_clk	ca7.spll_bus_clk
6	ca7.apll.pfd0_clk	ca7.nic1_clk	ca7.apll.pfd0_clk	ca7.nic1_clk
5	ca7.apll.pfd1_clk	ca7.nic1_bus_clk	ca7.apll.pfd1_clk	ca7.nic1_bus_clk
4	ca7.apll.pfd2_clk	osc_32k (rtc_32k)	ca7.apll.pfd2_clk	osc_32k (rtc_32k)
3	ca7.ddr_clk	ca7.fircdiv2	ca7.ddr_clk	ca7.fircdiv2
2	ca7.nic1_clk	mipi_pll_clk	ca7.nic1_clk	mipi_pll_clk
1	ca7.nic1_bus_clk	ca7.soscdiv2	ca7.nic1_bus_clk	ca7.soscdiv2
0	OFF	OFF	ca7.nic1_clk	OFF

Table 24-4. PCC Slots and Corresponding PCC to Peripheral Mapping Table: Real Time Domain

PCC:Slot No.	Peripheral	PCC Mux used
0:37	WDOG0	PCC0_BUS
0:45	LPIT0	PCC0_BUS
0:48	TPM0	PCC0_BUS
0:49	TPM1	PCC0_BUS
0:50	FLEXIO0	PCC0_BUS
0:51	LPI2C0	PCC0_BUS
0:52	LPI2C1	PCC0_BUS
0:53	LPI2C2	PCC0_BUS
0:54	LPI2C3	PCC0_BUS
0:55	SAI0	PCC0_BUS
0:56	LPSPI0	PCC0_BUS
0:57	LPSPI1	PCC0_BUS
0:58	LPUART0	PCC0_BUS
0:59	LPUART1	PCC0_BUS
0:65	ADC0	PCC0_BUS
0:68	DAC0	PCC0_BUS
0:69	DAC1	PCC0_BUS
1:20	TPIU/SWO	PCC1_PLAT

Table continues on the next page...

Table 24-4. PCC Slots and Corresponding PCC to Peripheral Mapping Table: Real Time Domain (continued)

PCC:Slot No.	Peripheral	PCC Mux used
1:37	QSPI_OTFAD	PCC1_PLAT
1:40	TPM2	PCC1_BUS
1:41	TPM3	PCC1_BUS
1:42	SAI1	PCC1_BUS
1:43	LPUART2	PCC1_BUS
1:44	LPUART3	PCC1_BUS
1:45	ADC1	PCC1_BUS

Table 24-5. PCC Slots and Corresponding PCC to Peripheral Mapping Table: Application Domain

PCC:Slot No.	Peripheral	PCC Mux used
2:37	LPTPM4	PCC2_BUS
2:38	LPTPM5	PCC2_BUS
2:39	LPIT1	PCC2_BUS
2:41	LPSPI2	PCC2_BUS
2:42	LPSPI3	PCC2_BUS
2:43	LPI2C4	PCC2_BUS
2:44	LPI2C5	PCC2_BUS
2:45	LPUART4	PCC2_BUS
2:46	LPUART5	PCC2_BUS
2:49	FLEXIO1	PCC2_BUS
2:51	USB0	PCC2_PLAT
2:52	USB1	PCC2_PLAT
2:55	USDHC0	PCC2_PLAT
2:56	USDHC1	PCC2_PLAT
2:61	WDG1	PCC2_BUS
2:67	WDG2	PCC2_BUS
3:33	LPTPM6	PCC3_BUS
3:34	LPTPM7	PCC3_BUS
3:36	LPI2C6	PCC3_BUS
3:37	LPI2C7	PCC3_BUS
3:38	LPUART6	PCC3_BUS
3:39	LPUART7	PCC3_BUS
3:41	DSI	PCC3_BUS
3:42	LCDIF	PCC3_PLAT
3:80	GPU3D	PCC3_PLAT
3:81	GPU2D	PCC3_PLAT

When this device comes out of reset, the M4 core always boots up first and is responsible for setting up the initial system clocks. It configures system clocking and informs the A7 of its clock frequency and the available clock options. In the event that both cores boot independently, each core will have the option to change the system clocks and the peripheral clocks in its domain.

24.6.1 Clock ratio restrictions

The clock ratio restrictions among the core, platform and IP bus clocks are listed as follows:

- A7 core clock frequency is higher than A7 platform clock frequency.
- Clock ratio must be integers between A7 fast platform (NIC0) and A7 slow platform (NIC1).

NOTE

Use A7 SPLL for core clock and A7 APLL for DDR/NIC clocks.

- Clock ratio must be integers between A7 slow platform and A7 system IP bus.
- Clock ratio must be integers between M4 core/platform and M4 system IP bus.
- M4 slow clock must be slower and an integer division of M4 system IP bus.
- A7 Slow platform (NIC1) clock frequency should be higher than A7 System IP bus clock (NIC1_BUS clock).

NOTE

See Operational requirements for additional restrictions on M4 bus clock/M4 slow clock ratio.

24.7 Generating audio frequencies

Fractional PLL (PLL1) in M4 domain have the ability to generate fractional multiplier that can be used to generate accurate audio frequencies.

$$PLL\ Output\ Freq = Fref\ x\ (MULT + \frac{NUM}{DENOM})$$

where:

- Fref = PLL input reference frequency
- MULT = 7-bit PLL multiplier

- NUM = 30-bit Numerator for fractional multiplier
- DENOM = 30-bit Denominator for fractional multiplier

Here's an example of PLL settings to generate desired audio frequency.

- Desired Freq = 44.1 KHz
- Over Sampled Freq = 256 x 44.1 KHz = 11.289 MHz
- Multiplier = 22.578 [MULT = 22, NUM = 578, DENOM = 1000]
- VCO Output = 24 MHz x 22.578 = 541.872 MHz
- VCO Output/DIV = 11.289 MHz = Over sampled Audio Freq where DIV = 48 (APLL Post Clock Dividers)

NOTE

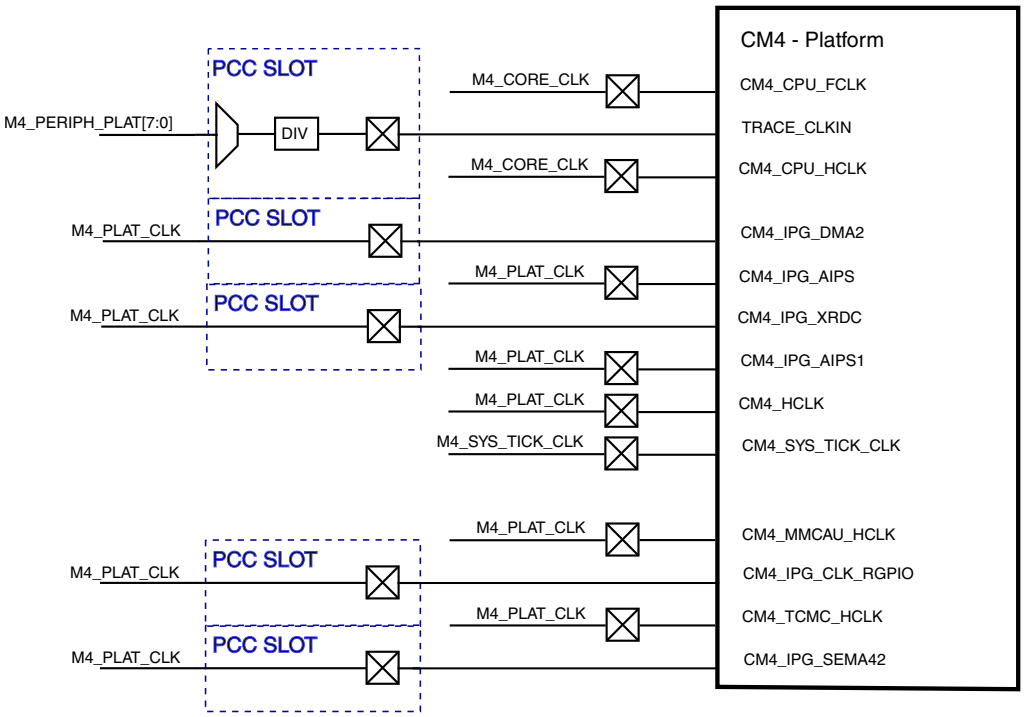
If this clock signal is being directed to either of the SAI PCC channels, APLL Post Clock dividers have to be configured to output a clock signal with a frequency under 50 MHz, due to limitations to the input frequency of the SAI PCC dividers.

Other audio frequencies can be similarly generated by manipulating multiplier (MULT), numerator (NUM) and denominator (DENOM).

24.8 Peripheral clocks

Most peripherals require fixed frequency clocks for their functional logics, such as generating the serial bit stream or reference timer. Therefore, peripheral functional clocks must be independent from the peripheral system bus clock, which can be changed during power mode transition. For peripherals that don't support independent system bus clocks, an asynchronous IPS gasket can be implemented at the IP bus interface.

Each module that requires a functional peripheral clock is provided a separate clock tree with source selection and programmable divider. Peripheral clocks are generated inside the PCC modules and routed to the corresponding peripheral. The following figures show examples of peripheral clock generation for M4 and A7 cores.



Note: Refer to PCC block diagram for clock source mapping

Figure 24-4. M4 peripheral clock generation

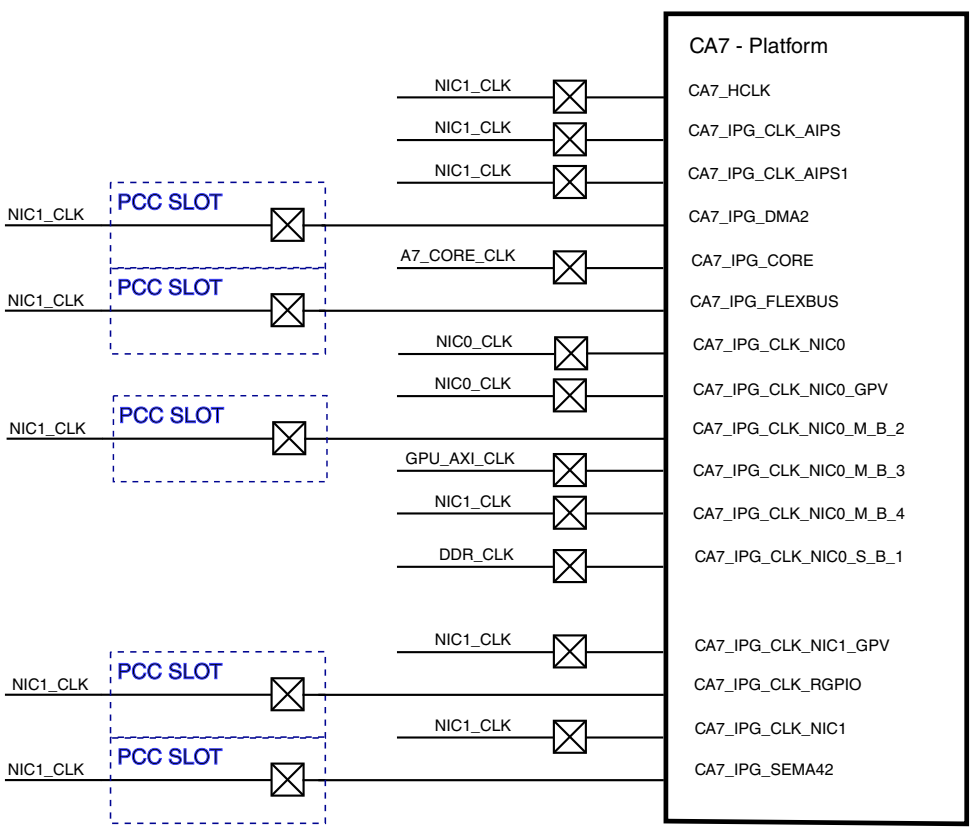


Figure 24-5. A7 peripheral clock generation

The following table lists peripheral clock frequencies and the indication of platform and IP bus clocks. Some peripherals have a local clock generator that can further divide the clock, as required, for the desired serial rate.

Table 24-6. Peripheral clock frequencies

Module	A7 Fast Platform Clk	A7 Slow Platform Clk	A7 System IP Bus Clk	M4 Platform Clk	M4 System IP Bus Clk
AIPS-Lite	--	--	--	Yes	Yes
AHB-PBridge	--	Yes	Yes	--	--
AXBS	--	--	--	Yes	Yes
NIC0	Yes	--	--	--	--
NIC1	--	Yes	--	--	--
AXI RAMC0	Yes	--	--	--	--
AXI RAMC1	--	Yes	--	--	--
AHB RAMC	--	--	--	Yes	--
MMDC	Yes	--	Yes	--	--
FlexBus	--	Yes	Yes	--	--
QSPI	--	--	--	Yes	Yes
DMA1	--	Yes	Yes	--	--
DMA0	--	--	--	Yes	Yes
GPU-3D	Yes	Yes	--	--	--
GPU-2D	--	Yes	--	--	--
LPUART0-3	--	--	--	--	Yes
LPUART4-7	--	--	Yes	--	--
LPSPI0-1	--	--	--	--	Yes
LPSPI2-3	--	--	Yes	--	--
LPI2C0-3	--	--	--	--	Yes
LPI2C4-7	--	--	Yes	--	--
USB Controllers	--	Yes	Yes	--	--
USB PHY	--	--	Yes	--	--
USB HSIC	--	--	Yes	--	--
uSDHC	--	Yes	Yes	--	--
RGPIO2P0	--	--	--	--	Yes
RGPIO2P1	--	--	Yes	--	--
FlexIO0	--	--	--	--	Yes
FlexIO1	--	--	Yes	--	--
LPIT0	--	--	--	--	Yes
LPIT1	--	--	Yes	--	--
TPM0-3	--	--	--	--	Yes
TPM4-7	--	--	Yes	--	--
LPTMR	--	--	--	--	Yes
EWM	--	--	--	--	Yes

Table continues on the next page...

Table 24-6. Peripheral clock frequencies (continued)

Module	A7 Fast Platform Clk	A7 Slow Platform Clk	A7 System IP Bus Clk	M4 Platform Clk	M4 System IP Bus Clk
DSI	--	Yes	Yes	--	--
LCDIF	--	Yes	Yes	--	--
VIU	--	Yes	Yes	--	--
SAI0-1	--	--	--	--	Yes
CAAM[1]	--	Yes	Yes	--	--
SNVS	--	--	--	--	Yes
CRC	--	--	--	--	Yes
TRNG[1]	--	--	--	--	Yes
LTC[1]	--	--	--	--	Yes
JTAG	--	--	--	--	Yes
XRDC	--	--	--	--	Yes
SEM42	--	--	--	--	Yes
MU	--	--	Yes	--	Yes
WDOG0	--	--	--	--	Yes
WDOG1	--	--	Yes	--	--
WDOG2 (Secure WDOG)	--	--	Yes	--	--
ADC0-1	--	--	--	--	Yes
DAC	--	--	--	--	Yes
CMP0-1	--	--	--	--	Yes
TPIU/SWO	--	--	--	--	--

1. See i.MX 7ULP Security Reference Manual for complete chapter

24.8.1 Application domain

24.8.1.1 Processor/Platform

24.8.1.1.1 CA7 Platform

24.8.1.1.1.1 DMA Channel MUX clocking

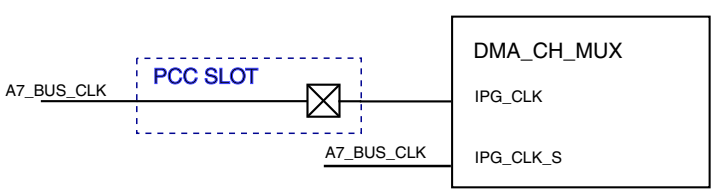


Figure 24-6. DMA Channel MUX clock diagram

24.8.1.1.1.2 DMA RAM

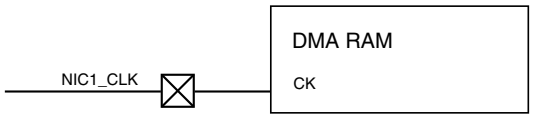


Figure 24-7. DMA RAM clock diagram

24.8.1.1.2 Internal memory

24.8.1.1.2.1 ROM

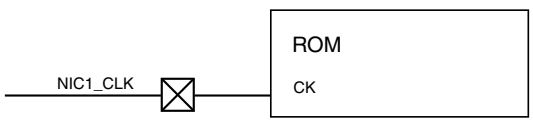


Figure 24-8. ROM clock diagram

24.8.1.1.2.2 OCRAM clocking

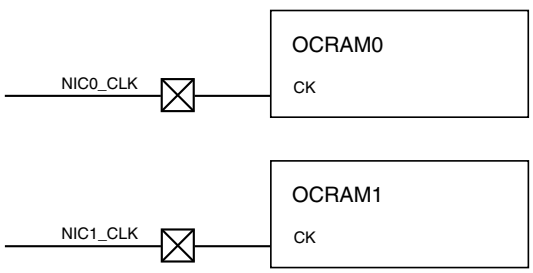


Figure 24-9. OCRAM clock diagram

24.8.1.1.3 External memory

24.8.1.1.3.1 uSDHC clocking

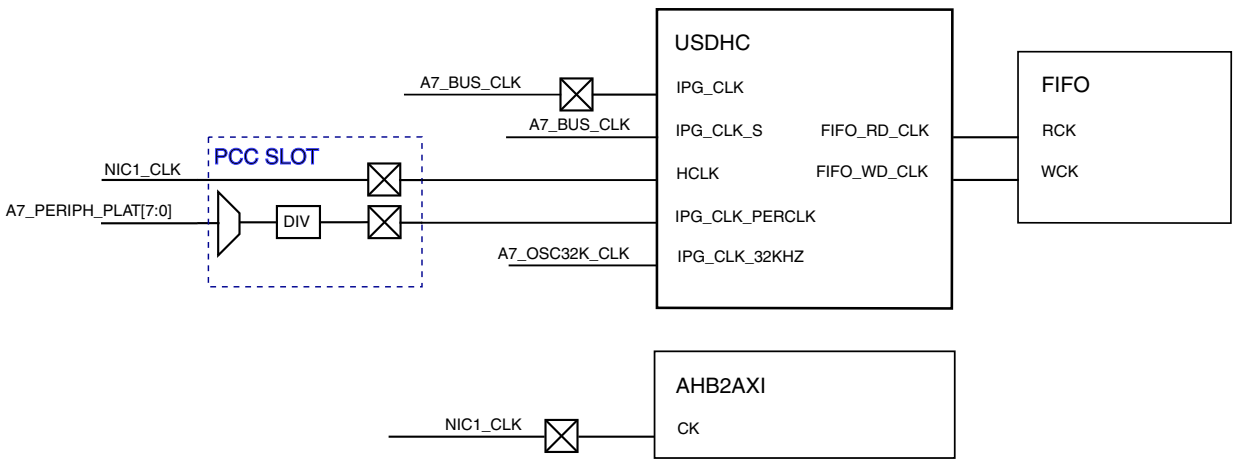


Figure 24-10. uSDHC clock diagram

24.8.1.1.3.2 MMDC clocking

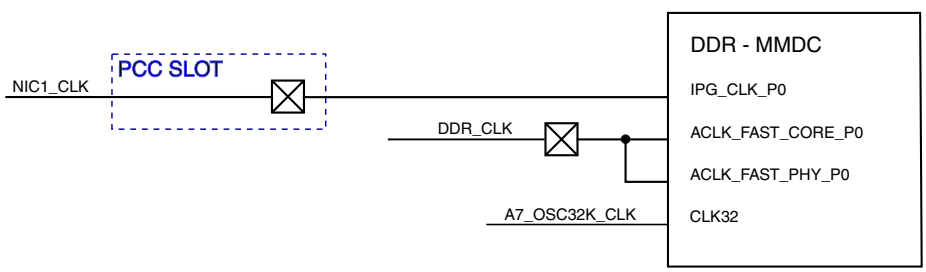


Figure 24-11. MMDC clock diagram

24.8.1.1.4 Multimedia

24.8.1.1.4.1 GPU2D clocking

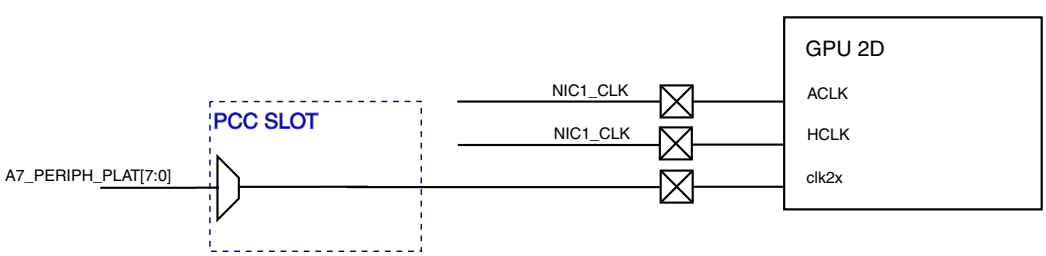


Figure 24-12. GPU 2D clock diagram

24.8.1.1.4.2 GPU3D clocking

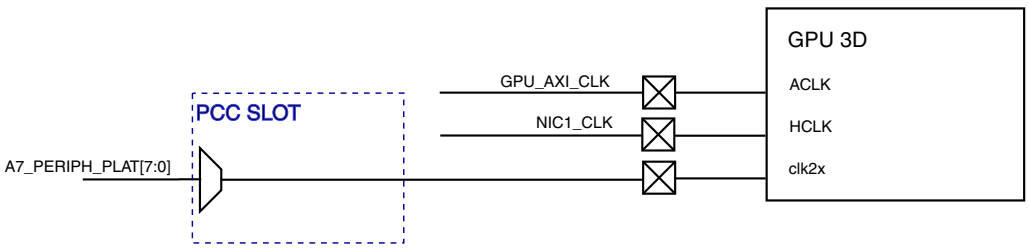


Figure 24-13. GPU3D clock diagram

24.8.1.1.4.3 LCDIF clocking

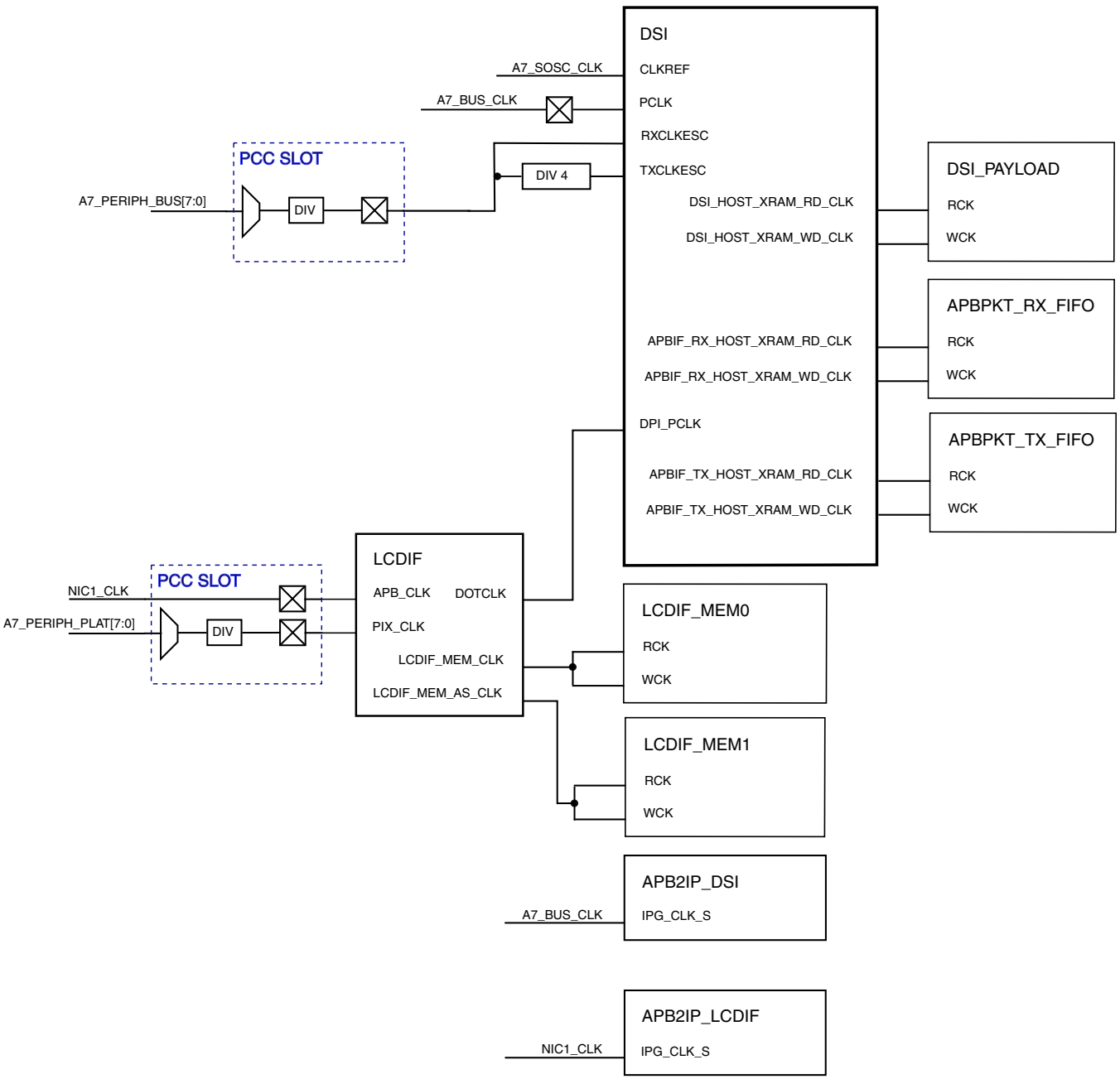


Figure 24-14. LCDIF clock diagram

24.8.1.1.4.4 VIU clocking

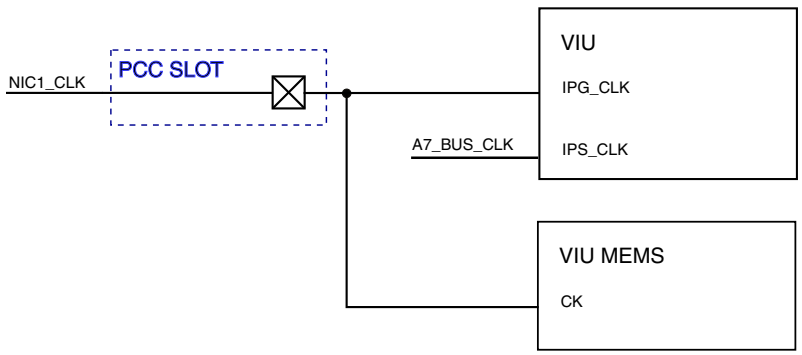


Figure 24-15. VIU clock diagram

24.8.1.1.5 Connectivity

24.8.1.1.5.1 LPUART clocking

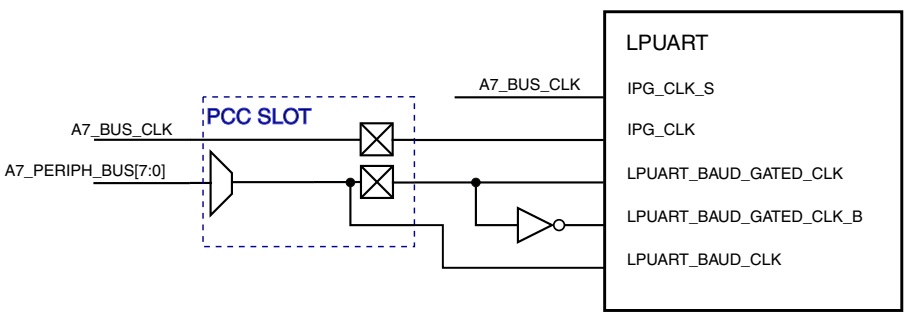


Figure 24-16. LPUART clock diagram

24.8.1.1.5.2 LPI2C clocking

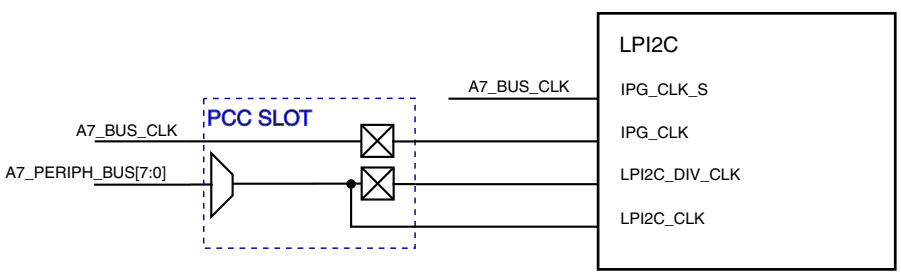


Figure 24-17. LPI2C clock diagram

24.8.1.1.5.3 LPSPI clocking

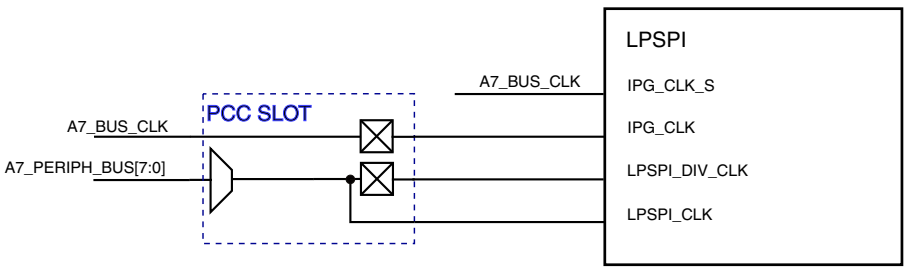
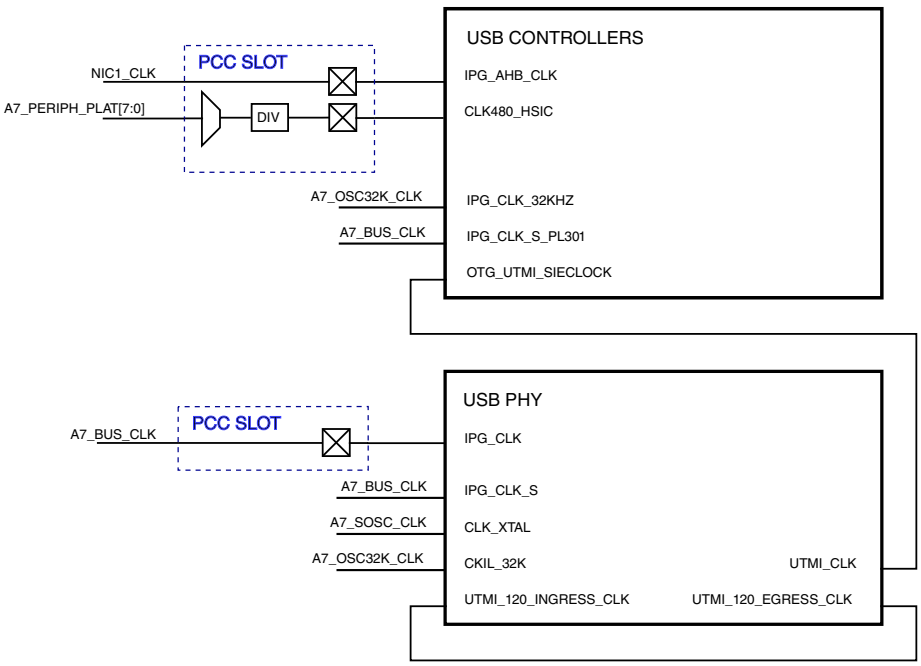


Figure 24-18. LPSPI clock diagram

24.8.1.1.5.4 USB clocking



24.8.1.1.5.5 GPIO clocking

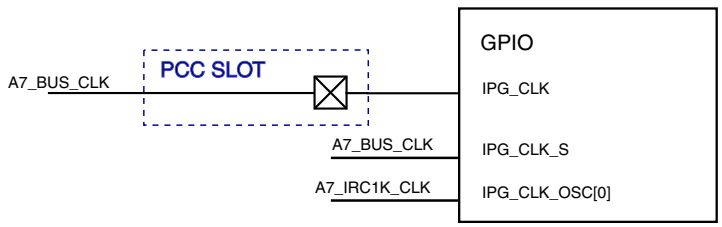


Figure 24-20. GPIO clock diagram

24.8.1.1.5.6 FlexIO clocking

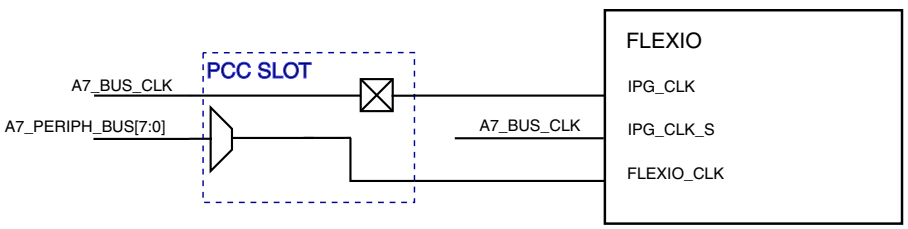


Figure 24-21. FlexIO clock diagram

24.8.1.1.6 Timers

24.8.1.1.6.1 LPTPM clocking

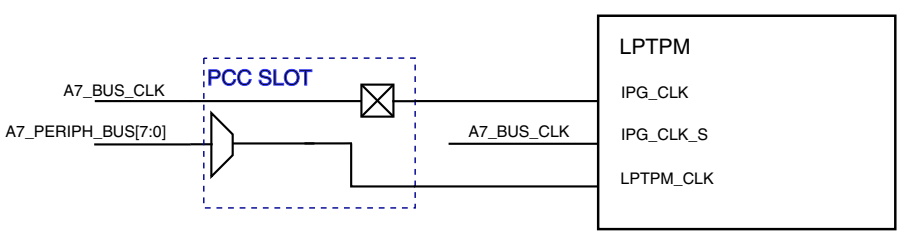


Figure 24-22. LPTPM clock diagram

24.8.1.1.6.2 LPIT clocking

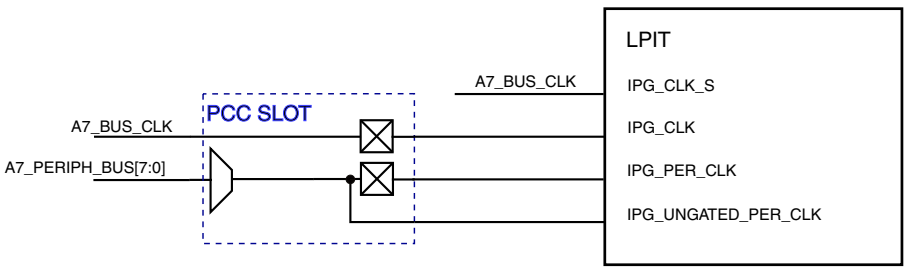


Figure 24-23. LPIT clock diagram

24.8.1.1.6.3 WDOG clocking

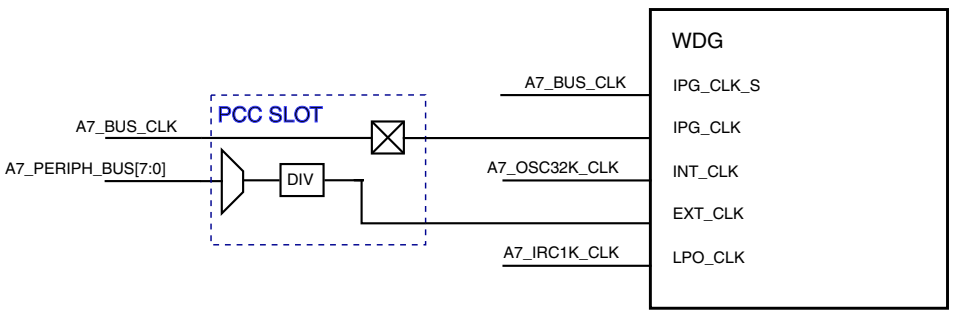


Figure 24-24. WDOG clock diagram

24.8.1.1.7 Security

24.8.1.1.7.1 CAAM clocking

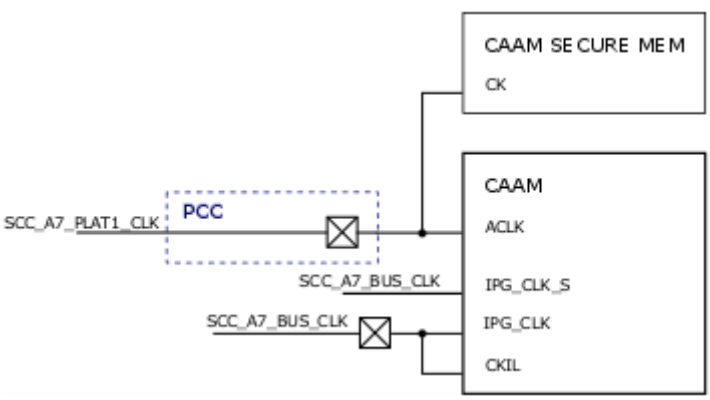


Figure 24-25. CAAM clock diagram

NOTE

The complete CAAM chapter is available only in the i.MX 7ULP Security Reference Manual.

24.8.1.1.8 System control

24.8.1.1.8.1 WKPU clocking

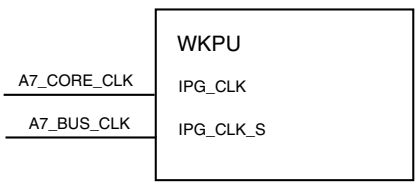


Figure 24-26. WKPU clock diagram

24.8.1.1.9 Others

24.8.1.1.9.1 TRGMUX clocking

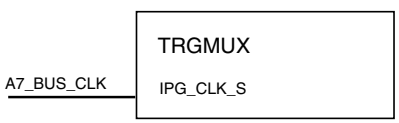


Figure 24-27. TRIGMUX clock diagram

24.8.2 Real-time domain

24.8.2.1 Processor/Platform

24.8.2.1.1 CM4 Platform

24.8.2.1.1.1 DMA Channel MUX clocking

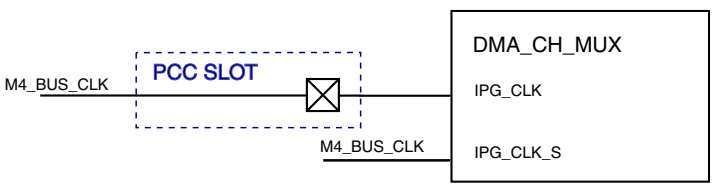


Figure 24-28. DMA Channel MUX clock diagram

24.8.2.1.1.2 DMA RAM

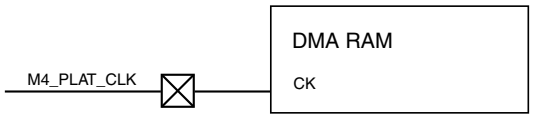


Figure 24-29. DMA RAM clock diagram

24.8.2.1.2 Internal memory

24.8.2.1.2.1 ROM

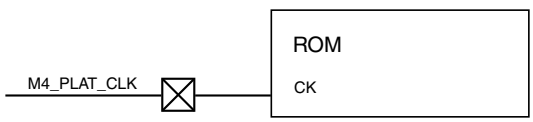


Figure 24-30. ROM clock diagram

24.8.2.1.2.2 TCM clocking

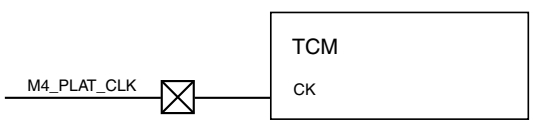


Figure 24-31. TCM clock diagram

24.8.2.1.2.3 Cache clocking

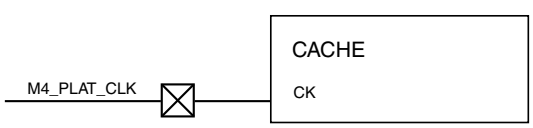


Figure 24-32. Cache clock diagram

24.8.2.1.3 External memory

24.8.2.1.3.1 QSPI OTFAD clocking

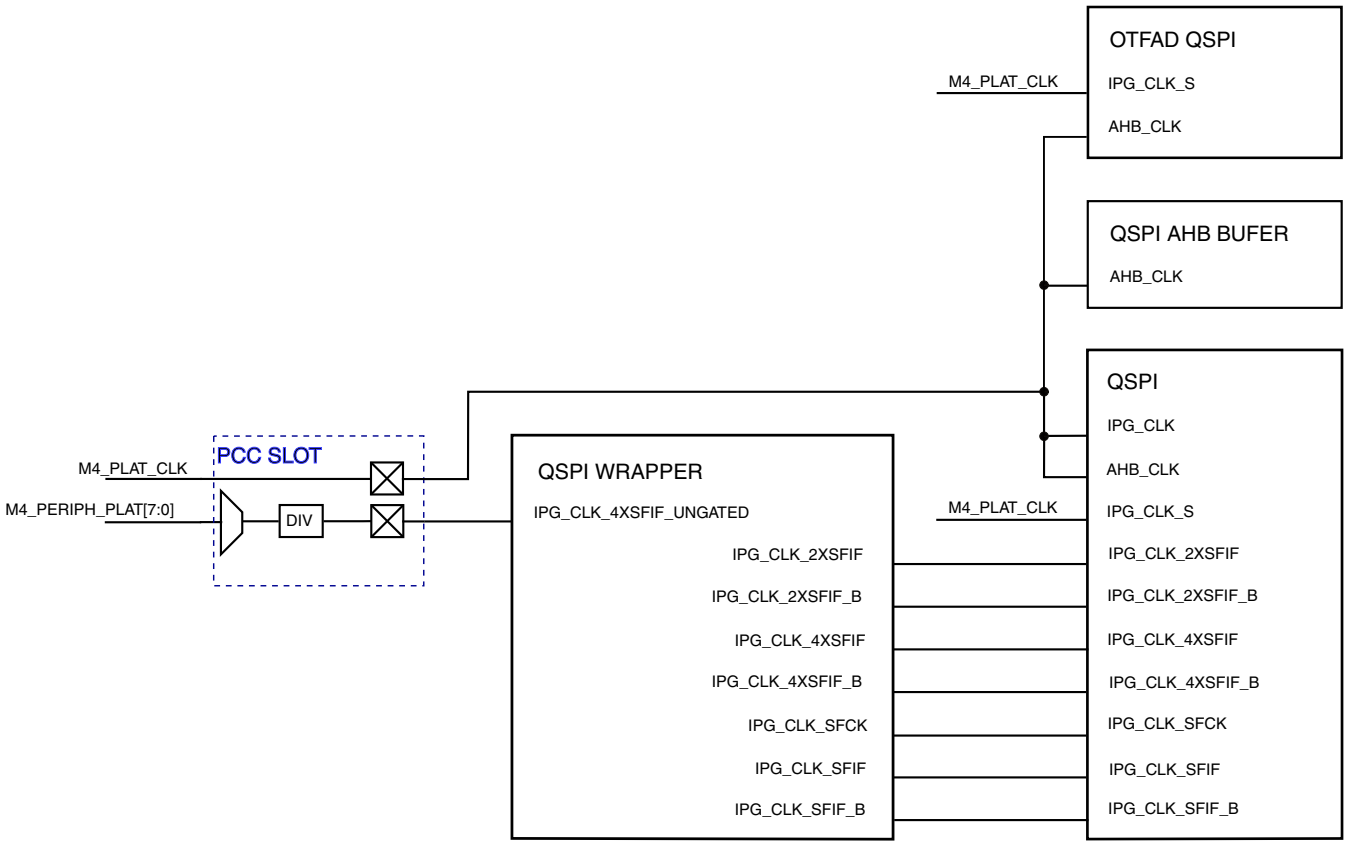


Figure 24-33. QSPI OTFAD clock diagram

NOTE

The complete OTFAD chapter is only available in the i.MX 7ULP Security Reference Manual.

24.8.2.1.4 Connectivity

24.8.2.1.4.1 LPUART clocking

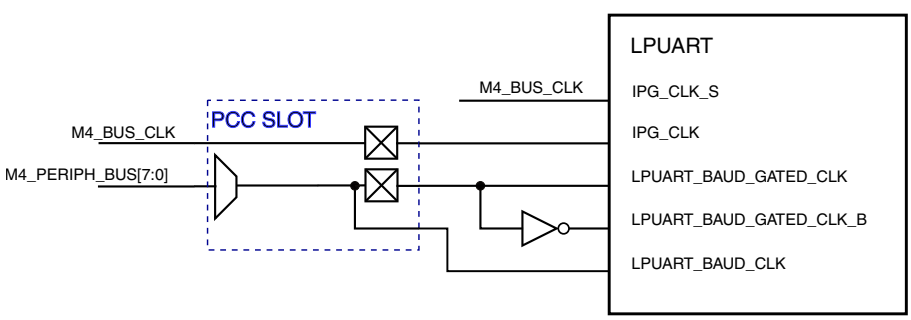


Figure 24-34. LPUART clock diagram

24.8.2.1.4.2 LPI2C clocking

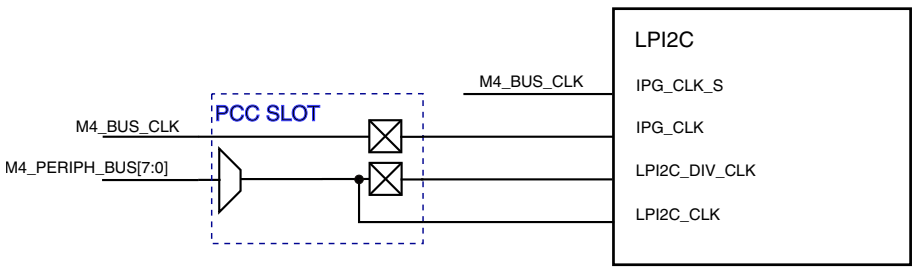


Figure 24-35. LPI2C clock diagram

24.8.2.1.4.3 LPSPI clocking

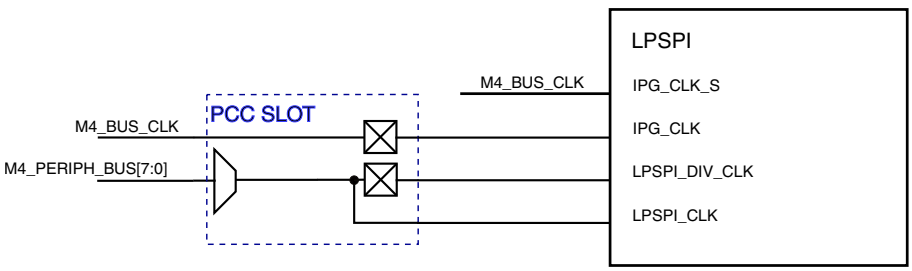


Figure 24-36. LPSPI clock diagram

24.8.2.1.4.4 SAI clocking

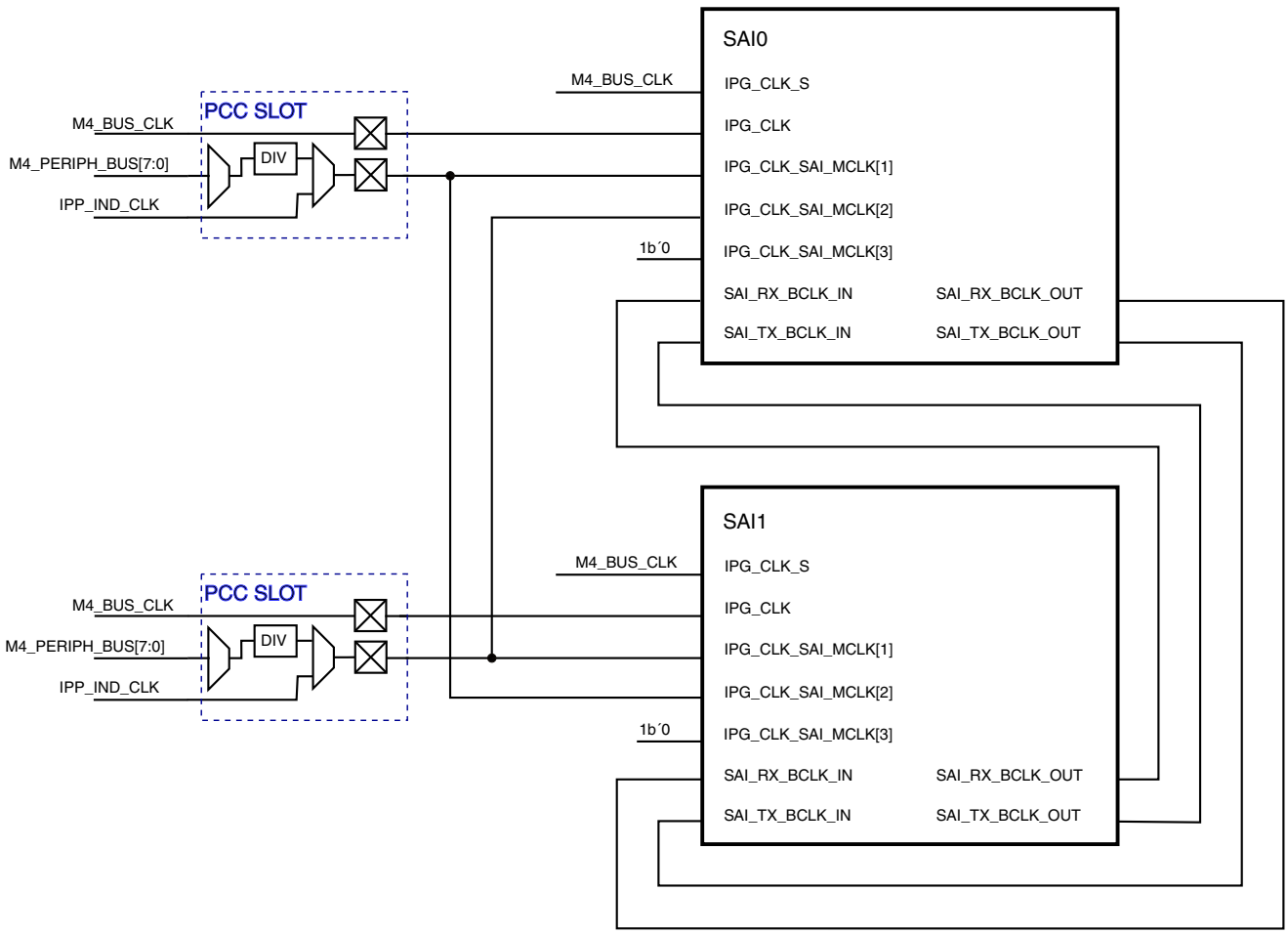


Figure 24-37. SAI clock diagram

24.8.2.1.4.5 GPIO clocking

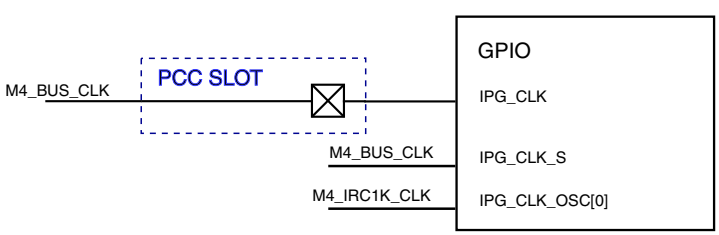


Figure 24-38. GPIO clock diagram

24.8.2.1.4.6 FlexIO clocking

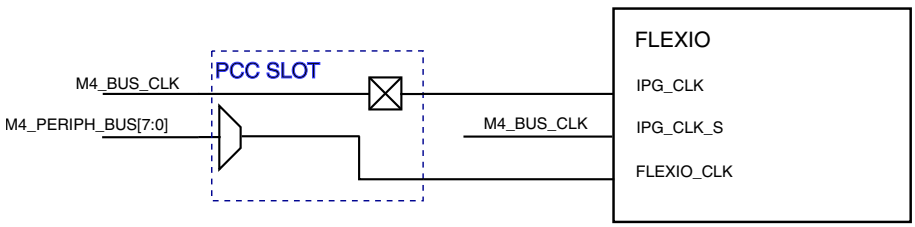


Figure 24-39. FlexIO clock diagram

24.8.2.1.5 Timers

24.8.2.1.5.1 LPTPM clocking

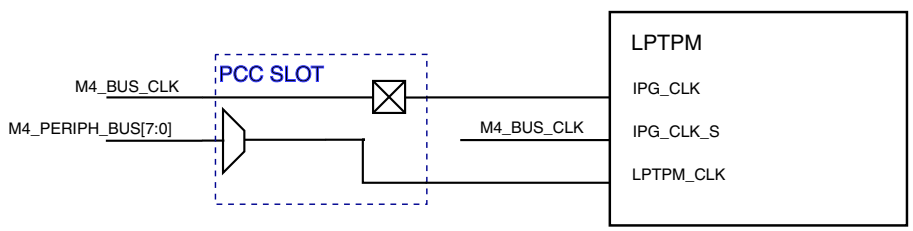


Figure 24-40. LPTPM clock diagram

24.8.2.1.5.2 LPIT clocking

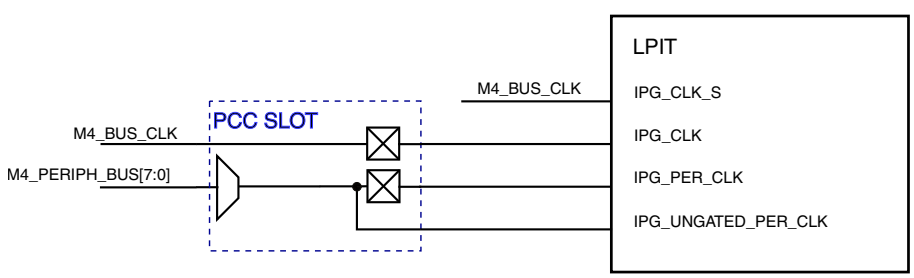


Figure 24-41. LPIT clock diagram

24.8.2.1.5.3 WDOG clocking

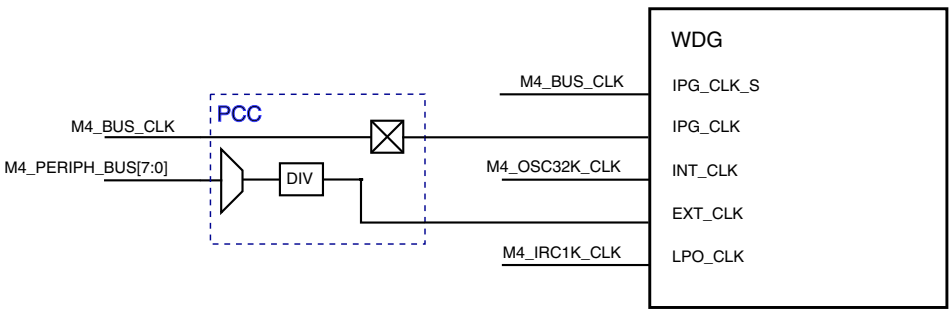


Figure 24-42. WDOG clock diagram

24.8.2.1.6 eFuses

24.8.2.1.6.1 OCOTP clocking

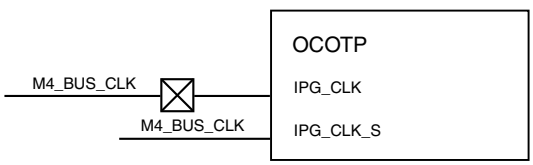


Figure 24-43. OCOTP clock diagram

24.8.2.1.7 Analog

24.8.2.1.7.1 ADC clocking

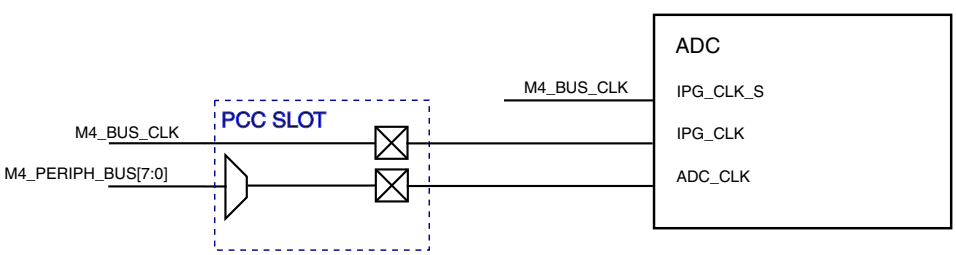


Figure 24-44. ADC clock diagram

24.8.2.1.7.2 DAC clocking

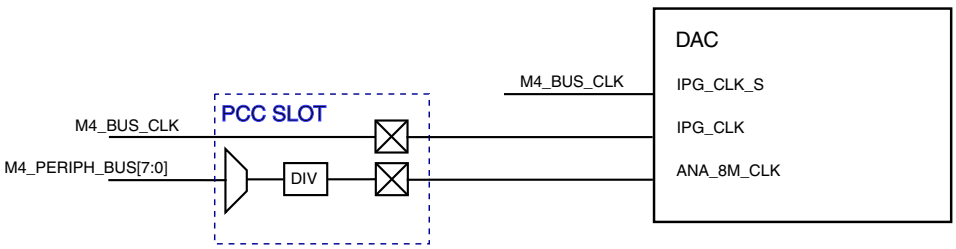


Figure 24-45. DAC clock diagram

24.8.2.1.8 Security

24.8.2.1.8.1 RNG clocking

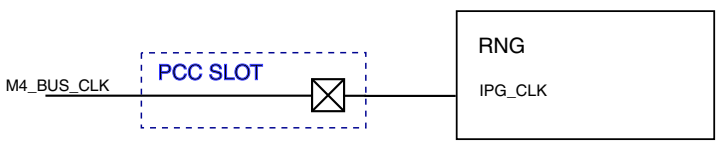


Figure 24-46. RNG clock diagram

NOTE

The complete TRNG chapter is only available in the i.MX 7ULP Security Reference Manual.

24.8.2.1.8.2 CRC clocking

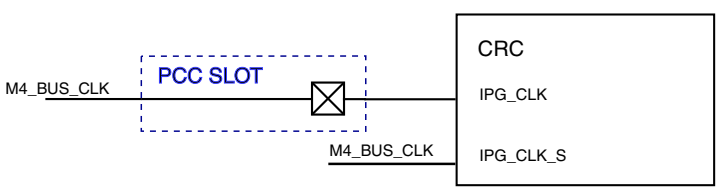


Figure 24-47. CRC clock diagram

24.8.2.1.8.3 LTC clocking

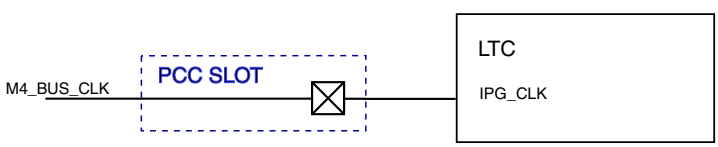


Figure 24-48. LTC clock diagram

NOTE

The complete LTC chapter is only available in the i.MX 7ULP Security Reference Manual.

24.8.2.1.8.4 SNVS clocking

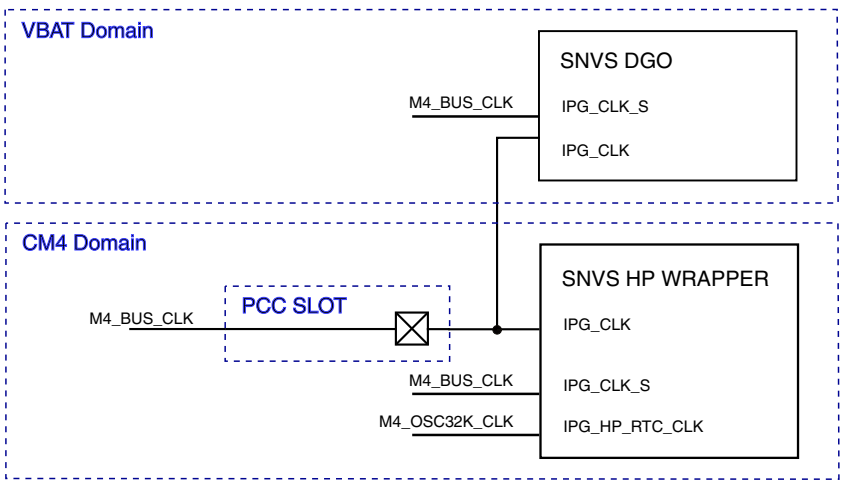


Figure 24-49. SNVS clock diagram

24.8.2.1.9 System control

24.8.2.1.9.1 SIM clocking

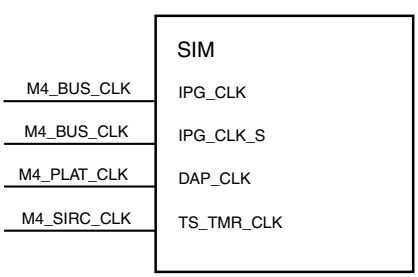


Figure 24-50. SIM clock diagram

24.8.2.1.9.2 AWIC clocking

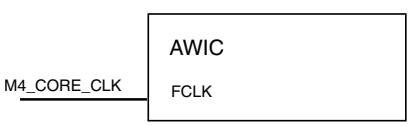


Figure 24-51. AWIC clock diagram

24.8.2.1.10 Others

24.8.2.1.10.1 TRGMUX clocking

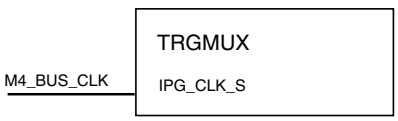


Figure 24-52. TRIGMUX clock diagram

24.8.2.1.10.2 MU clocking

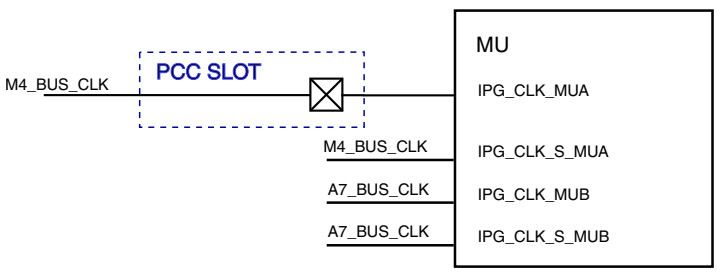


Figure 24-53. MU clock diagram

24.8.3 Always on domain

24.8.3.1 Wake-Up

24.8.3.1.1 LLWU

24.8.3.1.1.1 LLWU clocking

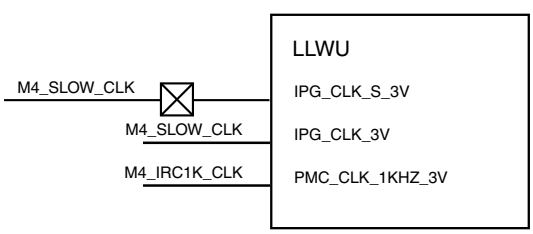


Figure 24-54. LLWU clock diagram

24.8.3.1.2 Timers

24.8.3.1.2.1 LPTMR clocking

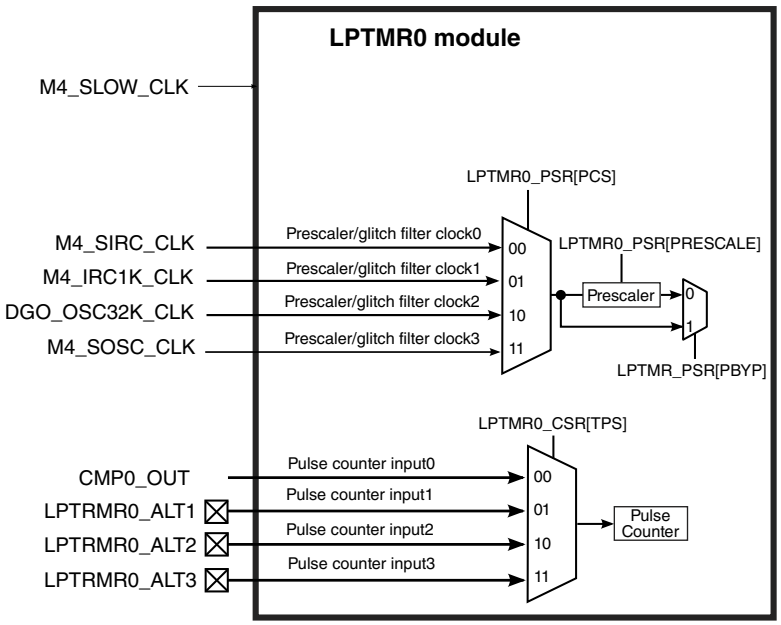


Figure 24-55. LPTMR0 clock diagram

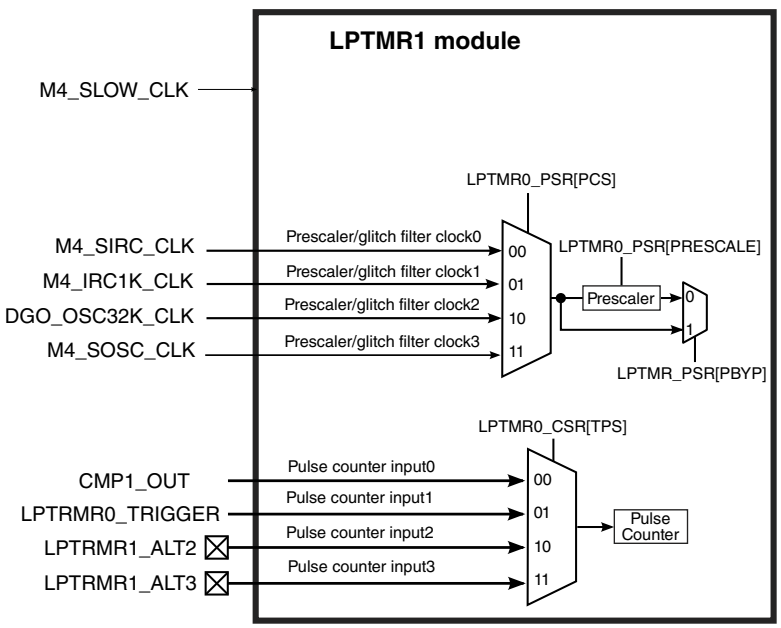


Figure 24-56. LPTMR1 clock diagram

24.8.3.1.3 Analog

24.8.3.1.3.1 CMP clocking

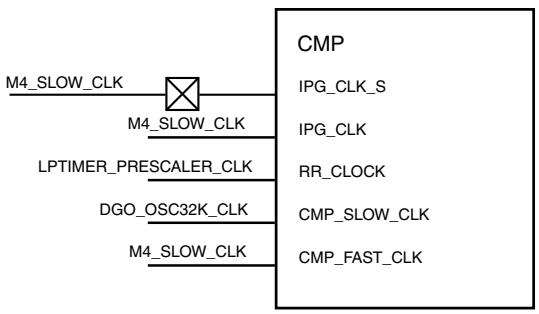


Figure 24-57. CMP clock diagram

24.8.3.1.4 Power control

24.8.3.1.4.1 MSMC clocking

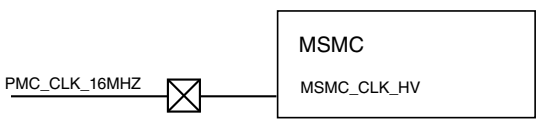


Figure 24-58. MSMC clock diagram

24.8.3.1.4.2 PMC clocking

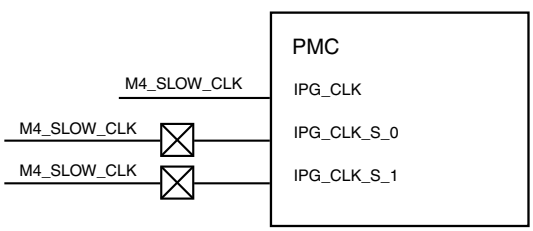


Figure 24-59. PMC clock diagram

24.8.3.1.5 Reset

24.8.3.1.5.1 RMC clocking

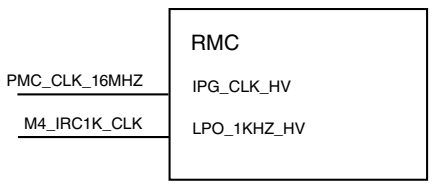


Figure 24-60. RMC clock diagram

24.9 Audio tunable clock

For audio applications where the data stream is coming from a remote source, the device has to locally tune a clock signal to match the remote system clock. The Auxiliary PLL, which provides the clock for master audio, has synchronization logic to support on-the-fly configuration changes. This allows the device to generate a tunable clock for audio stream. The clock from one of the Auxiliary PLLs (PLL1) can be divided by the post-dividers in analog and also the dividers in SCG module.

See the i.MX 7ULP Data Sheet, to see the requirement specifications for the divided tunable clock generated.

24.10 CoreSight debug clocks

As shown in the following table, the CoreSight debug clocks operate using IP Bus clock frequencies.

Table 24-7. CoreSight debug clocks

Clock	Used By	Sources
ATCLK	ATB interface	M4_PLAT_CLK
HCLK	DAP	M4_PLAT_CLK
PCLKDBG	APB interface, DAP, Trace Funnel, ETM, TPIU	M4_PLAT_CLK
DAPCLK		M4_PLAT_CLK
TPIUCLK	TPIU/SWO	TPIU/SWO Clock generation

24.11 Clock restrictions/guidelines

24.11.1 Limitation of NIC/GPU/DDR clock configuration for HSRUN and VLPR mode

There are limitations of NIC/GPU/DDR clock configuration for HSRUN and VLPR modes, because there are no separate registers for those clocks. CA7 Core clock and CM4 system clocks have three registers for each power mode (SCG_RCCR, SCG_VCCR, and SCG_HCCR).

For NIC/GPU/DDR clocks, the user must be careful with the following frequency configurations.

- For transition from RUN to HSRUN mode: NIC/GPU/DDR clocks need to be programmed at HSRUN maximum frequency after A7 domain entered the HSRUN mode. On the other hand, frequencies need to be reduced (RUN frequencies) before HSRUN mode exit.
- For transition from RUN to VLPR mode: NIC clocks need to be programmed before entering the VLPR mode. After that, the user needs to exit the VLPR mode, and then increase the frequencies to RUN mode values.

24.12 Clock dividers

The SCG and PCCs utilize clock dividers to divide the input source clocks and generate required functional clocks for various modules.

- The dividers implemented in SCG and PCCs are integer dividers and can provide any integer division value starting from divide-by-1.
- Division value of a divider is generally controlled by programming a register field.
- The maximum amount of time required to change the divided output clock frequency after programming a register bit is 1 clock cycle of input undivided clock + 1 cycle of output clock.

24.13 Default setting and clock gating

Extensive clock gating is used on this device. Clock gating is implemented at clock tree branches in the SCG for system clocks and in the PCCs for peripheral clocks. The clock gating is controlled by register bit programming and the low power mode of the respective domain.

After the system reset of this device, the following clocks will be enabled by default:

M4 Domain
- Core clock (FIRC)
- Platform clock (FIRC)
- Bus clock(FIRC)
- Slow clock (FIRC)

A7 Domain
- Core clock (FIRC)
- NIC0 clock (FIRC)
- NIC1 clock (FIRC)
- GPU clock (FIRC)
- NIC1 divbus (FIRC)
- NIC1 divext (FIRC)

The remainder of the clocks will be gated and can be enabled by the software or boot code.

24.14 Clock monitoring

The System oscillator clock and the RTC oscillator clock are monitored for their presence by clock monitoring logics in the SCG. Loss of crystal reference clock could generate system reset or an interrupt.

Chapter 25
Peripheral Clock Control

25.1 Chip-specific PCC information

Table 25-1. Reference links to related information

Topic	Related module	Reference
Full description	PCC	PCC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

25.1.1 Peripheral Clock Control

The Peripheral Clock Control (PCC) is responsible for clock selection, optional division and clock gating mode for peripherals in their respected power domain.

Table 25-2. PCC configuration

Parameter	Description
Name	PCC
Instances	4
Configurable features	Device specific
Interface speed	NA
External I/O pins	NA

25.1.2 PCC_WDOG register reset values

The reset values of PCC_WDOG1, and PCC_WDOG2 are different, depending on which core is reading the PCC_WDOG1 and PCC_WDOG2 registers, as depicted in the following table.

Table 25-3. PCC_WDOG register reset values

Register name	Reset value when accessed by CM4	Reset value when accessed by CA7
PCC_WDOG1	C0000000h	A0000000h
PCC_WDOG2	C0000000h	A0000000h

25.1.3 PCC reserved registers

PCC0 and PCC1 contains several registers which are reserved for internal use only and are not documented in the PCC memory maps. The offsets and reset values of these registers are given in the table below.

Table 25-4. PCC0 Reserved registers

PCC0 Register Offset	Reset Value
0x41026084	E0000000h
0x41026090	E0000000h
0x41026098	E0000000h
0x4102609C	E0000000h
0x410260F4	E0000000h
0x41026120	E0000000h

Table 25-5. PCC1 Reserved registers

PCC1 Register Offset	Reset Value
0x410B2040	E0000000h
0x410B2080	E0000000h
0x410B2084	E0000000h
0x410B208C	E0000000h
0x410B2090	E0000000h
0x410B2098	E0000000h
0x410B209C	E0000000h
0x410B20C8	E0000000h

25.1.4 Clock source selection through PCS field

The PCS bitfields in all the PCC registers is used for peripherals that support various clock selections. PCS can support upto 8 different clock selections, depending on the value of this field from 000 to 111. On this device, there are 4 instances of PCC: PCC0, PCC1, PCC2, and PCC3.

See Table 24-2 for clock sources mapped to different PCC0/PCC1 clock selects PCS and see Table 24-3 for clock sources mapped to different PCC2/PCC3 clock select PCS.

25.1.5 PCC register implementation on this device

- PCC_TPIU register: In i.MX 7ULP, PCC1_TPIU register is also used for the clock configuration of SWO.
- PCC_USB register: In this device, the HSIC clock programming is done via PCC_USB0[PCS]. PCC_USB1[PCS] is not used.

25.1.6 Peripheral usage restrictions

As documented in each and every PCC register, the INUSE bitfield (bit 29) shows whether a given peripheral instance is currently being used or not by another CPU. Therefore, if a given CPU reads such bitfield as "set", interactions going beyond ordinary register reads between the domain such CPU belongs to and the respective peripheral should be avoided until the bitfield becomes "clear" again. Failing to observe such peripheral usage restrictions may result in unexpected results.

25.2 Introduction

The Peripheral Clock Control (PCC) module provides clock control and configuration for on-chip peripherals. Each peripheral has its own clock control and configuration register.

25.3 Features

The PCC module enables software to configure the following clocking options for each peripheral:

- Interface clock gating

- Functional clock source selection
- Functional clock divide values

Below is a block diagram of the PCC module:

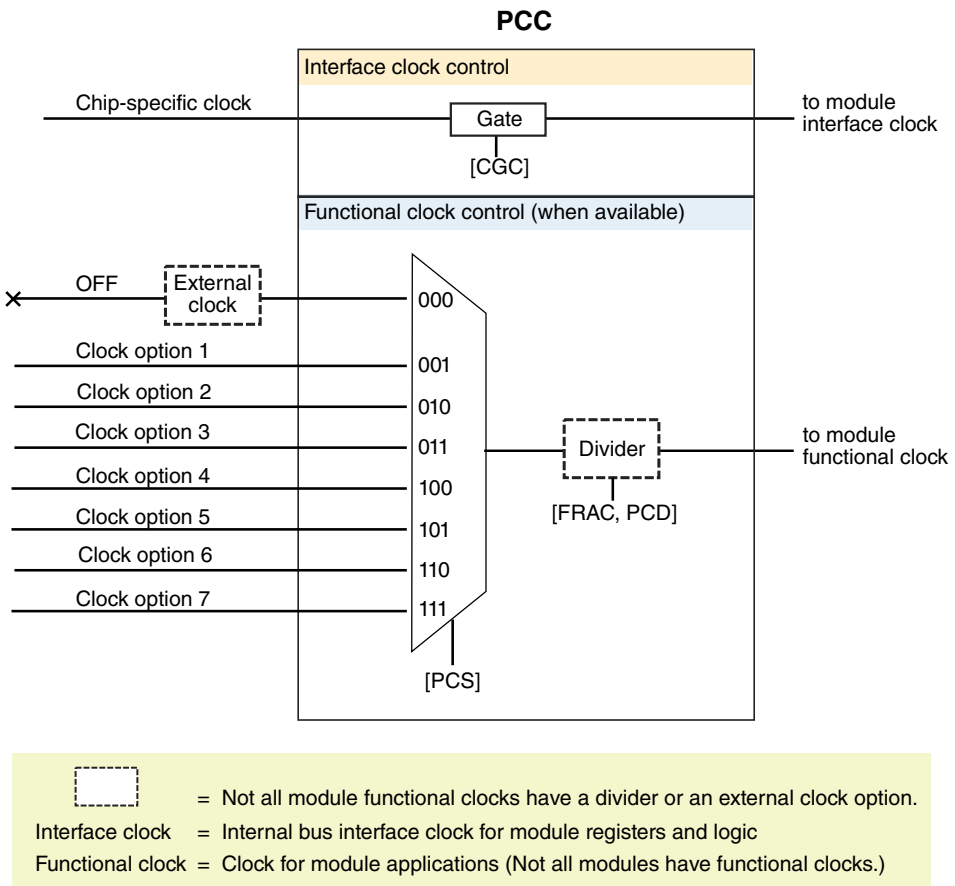


Figure 25-1. PCC Block Diagram

25.4 Functional description

The PCC module provides on-chip peripherals (modules) their own dedicated PCC registers for clock gating and configuration options. Each module's PCC register contains a clock gating control bit (CGC) for the module's interface clock. Before a module can be used, its interface clock must be enabled (CGC = 1) in the module's PCC register.

If a module has a functional clock, its PCC register may provide options for the clock source, selected by programming the Peripheral Clock Select (PCS) field. Optionally, a module may also have a clock divider, selected by programming the Peripheral Clock Divider (PCD) field along with a Fraction (FRAC) field. Before configuring a functional clock, the module's interface clock must be disabled (CGC = 0).

25.5 Memory map and register definition

Each module has its own dedicated PCC register, which controls the clock gating, clock source and divider (when applicable) for that specific module. See each module's PCC register for details.

PCC registers can be written only in supervisor mode using 32-bit accesses.

NOTE

To configure the clocking options available to a given module or to modify an existing configuration, first disable the module's interface clock by writing 0 to its CGC bit.

25.6 PCC register descriptions

25.6.1 PCC Memory map

PCC0 base address: 4102_6000h

Offset	Register	Width (In bits)	Access	Reset value
20h	PCC DMA0 Register (PCC_DMA0)	32	RW	8000_0000h
3Ch	PCC RGPIO2P0 Register (PCC_RGPIO2P0)	32	RW	8000_0000h
50h	PCC XRDC Register (PCC_XRDC)	32	RW	8000_0000h
6Ch	PCC SEMA42_0 Register (PCC_SEMA42_0)	32	RW	8000_0000h
80h	PCC DMA_MUX0 Register (PCC_DMA_MUX0)	32	RW	8000_0000h
88h	PCC MU_A Register (PCC_MU_A)	32	RW	8000_0000h
94h	PCC WDOG0 Register (PCC_WDOG0)	32	RW	C000_0000h
A4h	PCC CRC Register (PCC_CRC)	32	RW	8000_0000h
A8h	PCC LTC Register (PCC_LTC)	32	RW	8000_0000h
B0h	PCC TRNG Register (PCC_TRNG)	32	RW	8000_0000h
B4h	PCC LPIT0 Register (PCC_LPIT0)	32	RW	8000_0000h
B8h	PCC LPTIMER0 Register (PCC_LPTIMER0)	32	RW	8000_0000h
BCh	PCC LPTIMER1 Register (PCC_LPTIMER1)	32	RW	8000_0000h
C0h	PCC TPM0 Register (PCC_TPM0)	32	RW	8000_0000h
C4h	PCC TPM1 Register (PCC_TPM1)	32	RW	8000_0000h
C8h	PCC FLEXIO0 Register (PCC_FLEXIO0)	32	RW	8000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
CCh	PCC LPI2C0 Register (PCC_LPI2C0)	32	RW	8000_0000h
D0h	PCC LPI2C1 Register (PCC_LPI2C1)	32	RW	8000_0000h
D4h	PCC LPI2C2 Register (PCC_LPI2C2)	32	RW	8000_0000h
D8h	PCC LPI2C3 Register (PCC_LPI2C3)	32	RW	8000_0000h
DCh	PCC SAI0 Register (PCC_SAI0)	32	RW	8000_0000h
E0h	PCC LPSPI0 Register (PCC_LPSPI0)	32	RW	8000_0000h
E4h	PCC LPSPI1 Register (PCC_LPSPI1)	32	RW	8000_0000h
E8h	PCC LPUART0 Register (PCC_LPUART0)	32	RW	8000_0000h
ECh	PCC LPUART1 Register (PCC_LPUART1)	32	RW	8000_0000h
FCh	PCC PCTLA Register (PCC_PCTLA)	32	RW	8000_0000h
100h	PCC PCTLB Register (PCC_PCTLB)	32	RW	8000_0000h
104h	PCC ADC0 Register (PCC_ADC0)	32	RW	8000_0000h
108h	PCC CMP0 Register (PCC_CMP0)	32	RW	8000_0000h
10Ch	PCC CMP1 Register (PCC_CMP1)	32	RW	8000_0000h
110h	PCC DAC0 Register (PCC_DAC0)	32	RW	8000_0000h
114h	PCC DAC1 Register (PCC_DAC1)	32	RW	8000_0000h
1C0h	PCC SNVS Register (PCC_SNVS)	32	RW	8000_0000h

25.6.2 PCC DMA0 Register (PCC_DMA0)

25.6.2.1 Offset

Register	Offset
PCC_DMA0	20h

25.6.2.2 Function

This register is for the DMA0 module.

25.6.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.2.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.3 PCC RGPIO2P0 Register (PCC_RGPIO2P0)

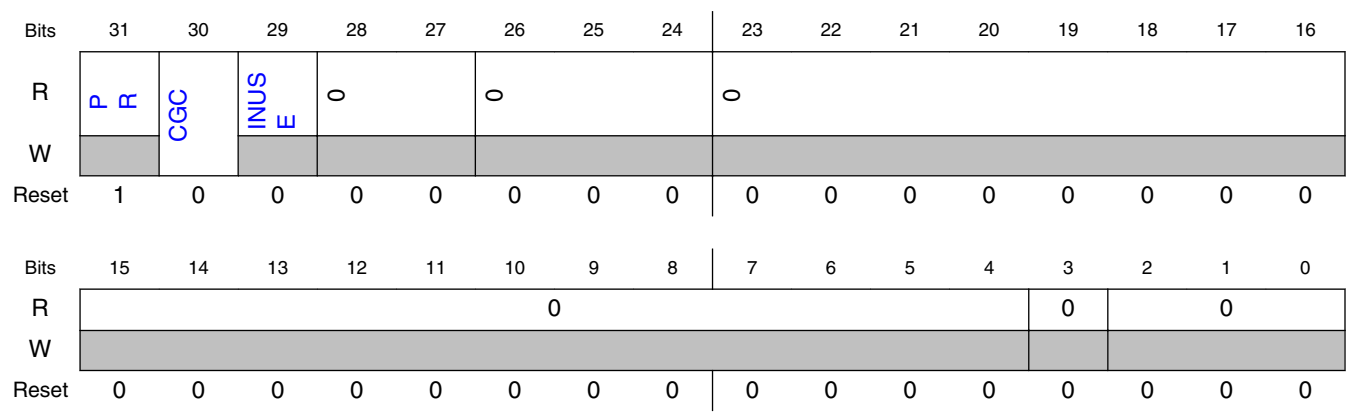
25.6.3.1 Offset

Register	Offset
PCC_RGPIO2P0	3Ch

25.6.3.2 Function

This register is for the RGPIO2P0 module.

25.6.3.3 Diagram



25.6.3.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.4 PCC XRDC Register (PCC_XRDC)

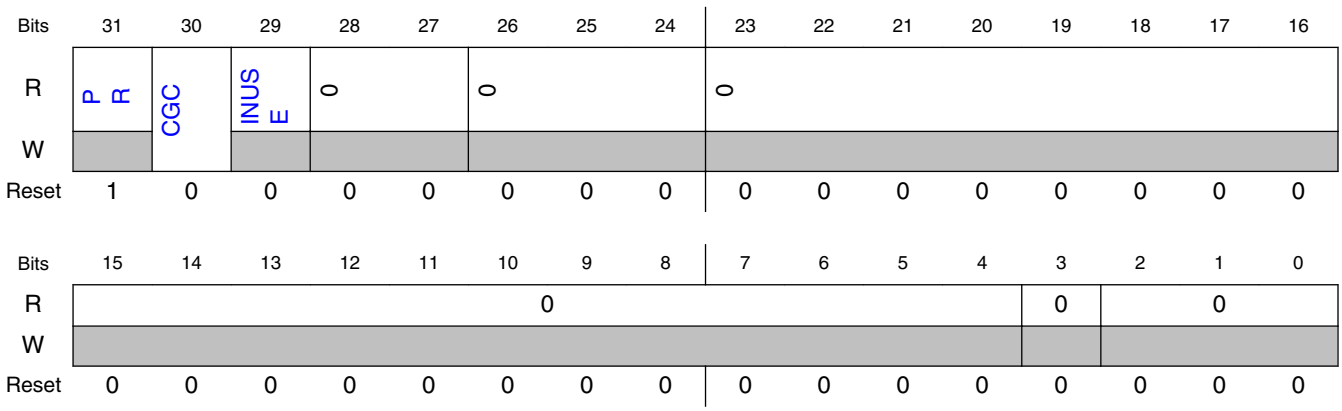
25.6.4.1 Offset

Register	Offset
PCC_XRDC	50h

25.6.4.2 Function

This register is for the XRDC module.

25.6.4.3 Diagram



25.6.4.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.5 PCC SEMA42_0 Register (PCC_SEMA42_0)

25.6.5.1 Offset

Register	Offset
PCC_SEMA42_0	6Ch

25.6.5.2 Function

This register is for the SEMA42_0 module.

25.6.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.5.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.6 PCC DMA_MUX0 Register (PCC_DMA_MUX0)

25.6.6.1 Offset

Register	Offset
PCC_DMA_MUX0	80h

25.6.6.2 Function

This register is for the DMA_MUX0 module.

25.6.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.6.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.7 PCC MU_A Register (PCC_MU_A)

25.6.7.1 Offset

Register	Offset
PCC_MU_A	88h

25.6.7.2 Function

This register is for the MU_A module.

25.6.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.7.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.8 PCC WDOG0 Register (PCC_WDOG0)

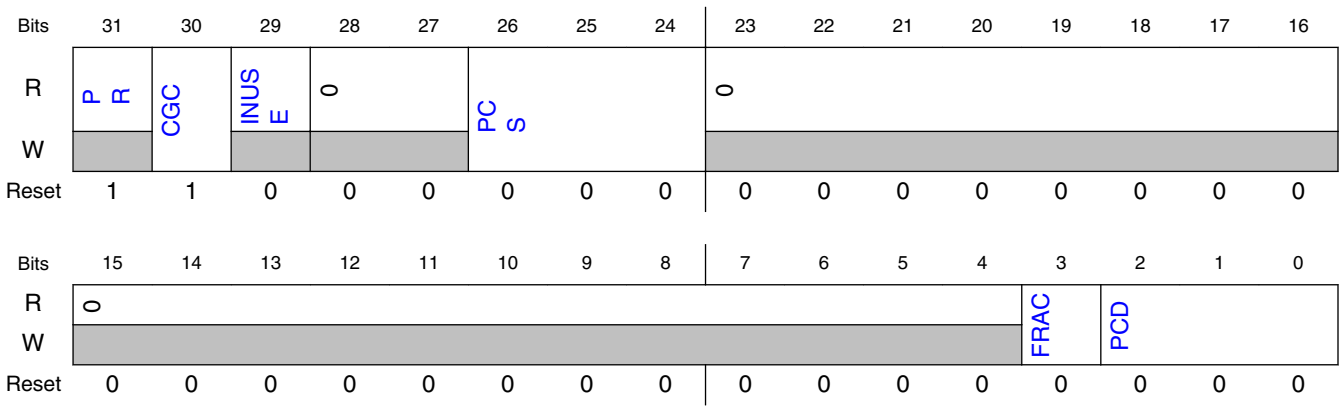
25.6.8.1 Offset

Register	Offset
PCC_WDOG0	94h

25.6.8.2 Function

This register is for the WDOG0 module.

25.6.8.3 Diagram



25.6.8.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.6.9 PCC CRC Register (PCC_CRC)

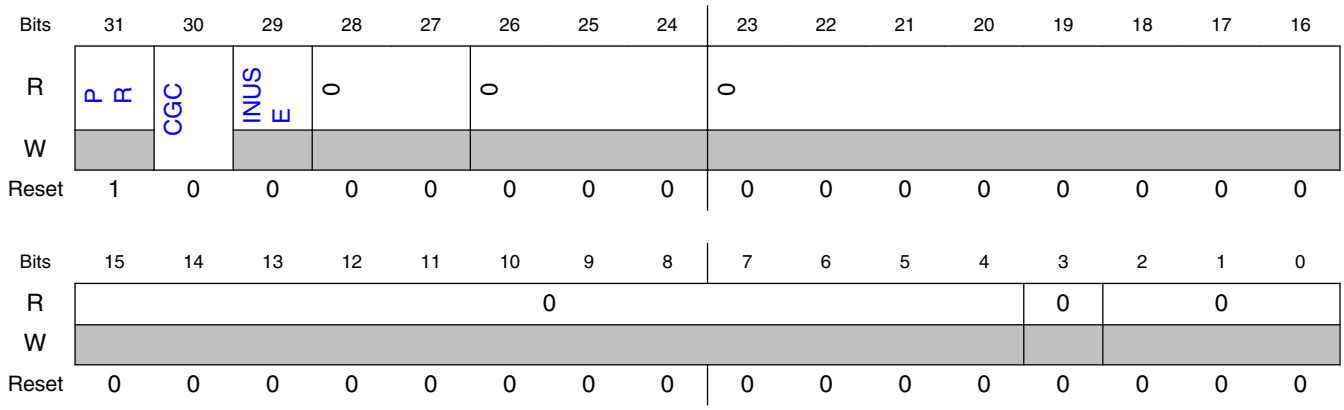
25.6.9.1 Offset

Register	Offset
PCC_CRC	A4h

25.6.9.2 Function

This register is for the CRC module.

25.6.9.3 Diagram



25.6.9.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.10 PCC LTC Register (PCC_LTC)

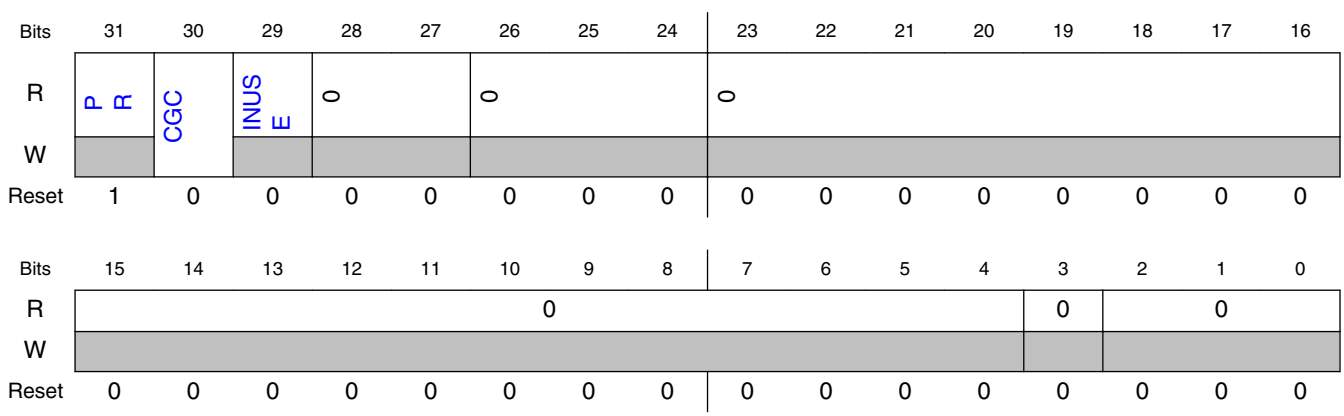
25.6.10.1 Offset

Register	Offset
PCC_LTC	A8h

25.6.10.2 Function

This register is for the LTC module.

25.6.10.3 Diagram



25.6.10.4 Fields

Field	Function
31	Present

Table continues on the next page...

Field	Function
PR	This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.11 PCC TRNG Register (PCC_TRNG)

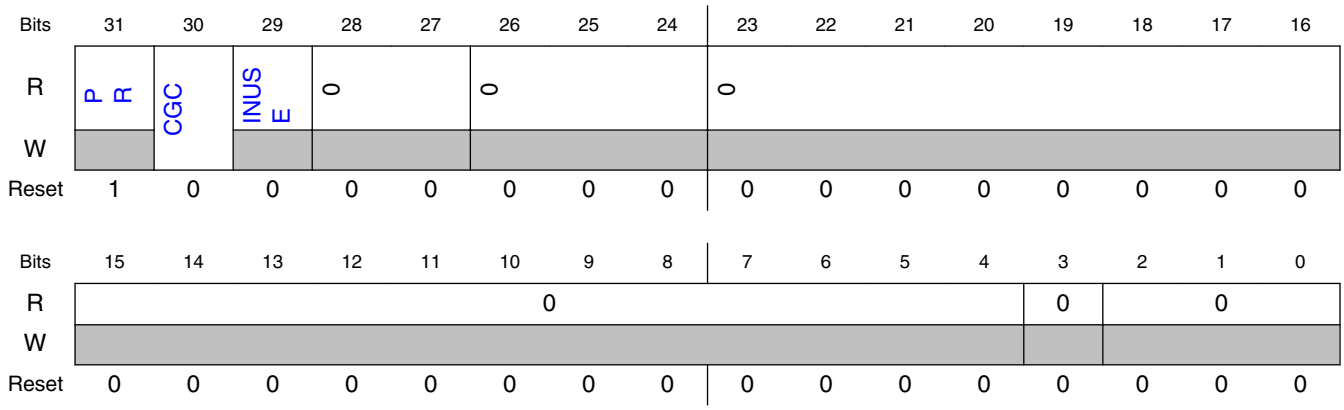
25.6.11.1 Offset

Register	Offset
PCC_TRNG	B0h

25.6.11.2 Function

This register is for the TRNG module.

25.6.11.3 Diagram



25.6.11.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.12 PCC LPIT0 Register (PCC_LPIT0)

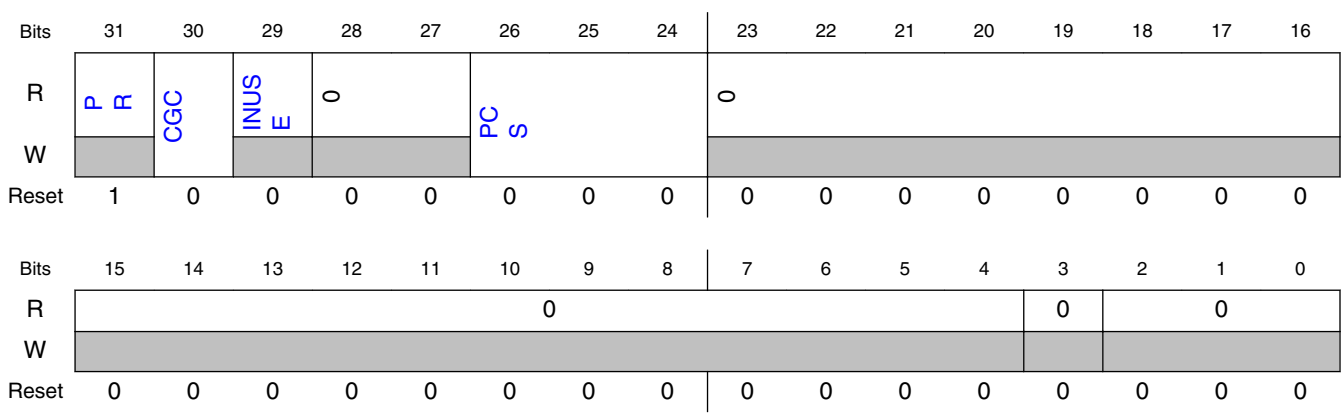
25.6.12.1 Offset

Register	Offset
PCC_LPIT0	B4h

25.6.12.2 Function

This register is for the LPIT0 module.

25.6.12.3 Diagram



25.6.12.4 Fields

Field	Function
31	Present

Table continues on the next page...

Field	Function
PR	This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.13 PCC LPTIMER0 Register (PCC_LPTIMER0)

25.6.13.1 Offset

Register	Offset
PCC_LPTIMER0	B8h

25.6.13.2 Function

This register is for the LPTIMER0 module.

25.6.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.13.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag

Table continues on the next page...

Field	Function
	This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.14 PCC LPTIMER1 Register (PCC_LPTIMER1)

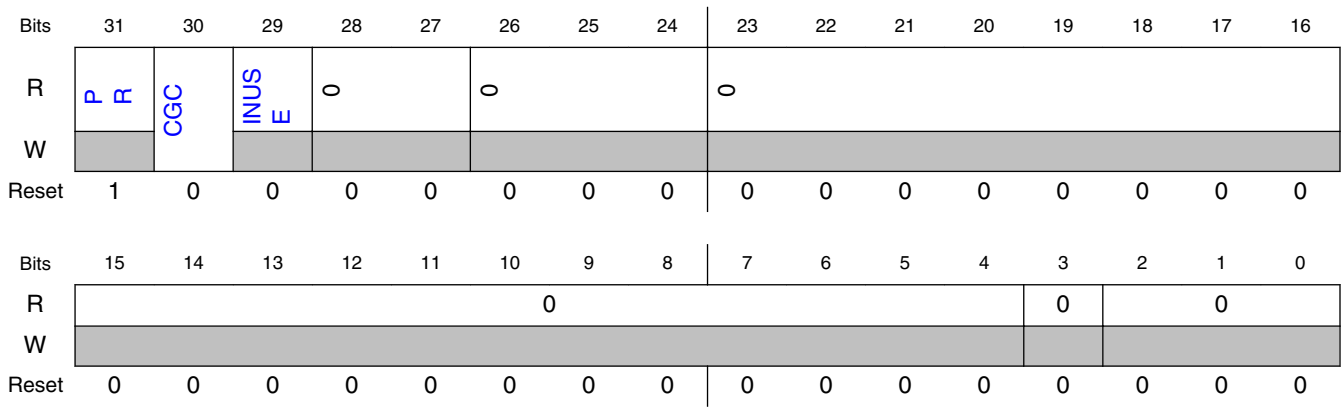
25.6.14.1 Offset

Register	Offset
PCC_LPTIMER1	BCh

25.6.14.2 Function

This register is for the LPTIMER1 module.

25.6.14.3 Diagram



25.6.14.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.15 PCC TPM0 Register (PCC_TPM0)

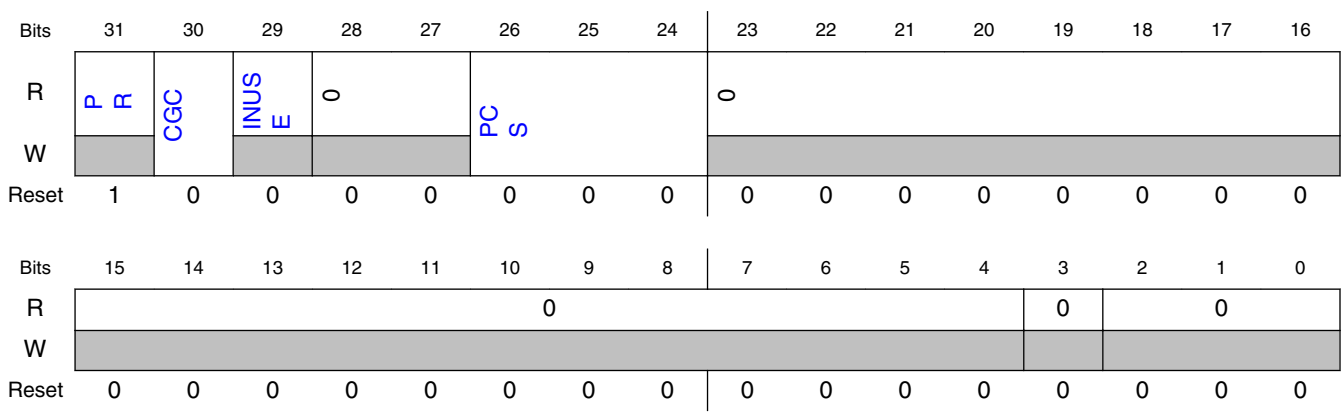
25.6.15.1 Offset

Register	Offset
PCC_TPM0	C0h

25.6.15.2 Function

This register is for the TPM0 module.

25.6.15.3 Diagram



25.6.15.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.16 PCC TPM1 Register (PCC_TPM1)

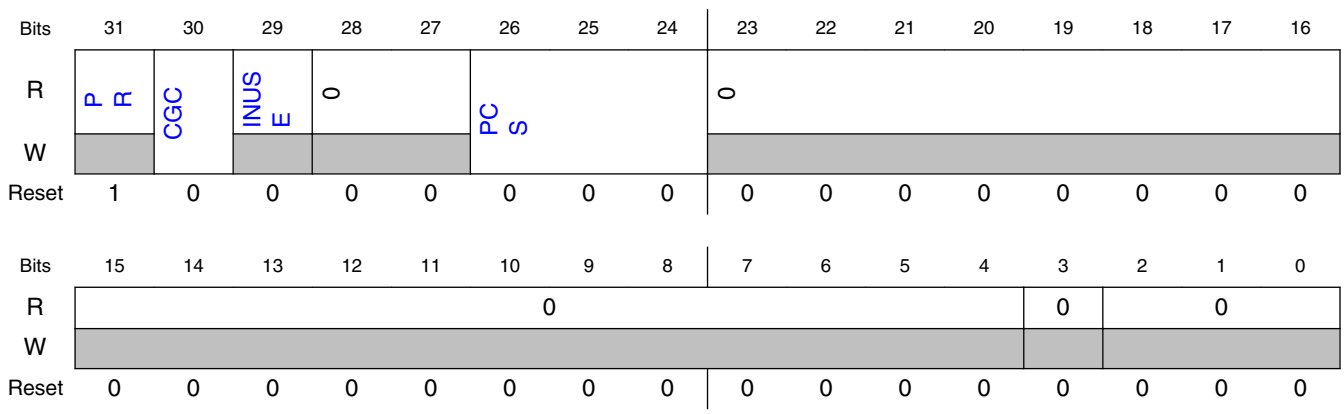
25.6.16.1 Offset

Register	Offset
PCC_TPM1	C4h

25.6.16.2 Function

This register is for the TPM1 module.

25.6.16.3 Diagram



25.6.16.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag

Table continues on the next page...

Field	Function
	This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.17 PCC FLEXIO0 Register (PCC_FLEXIO0)

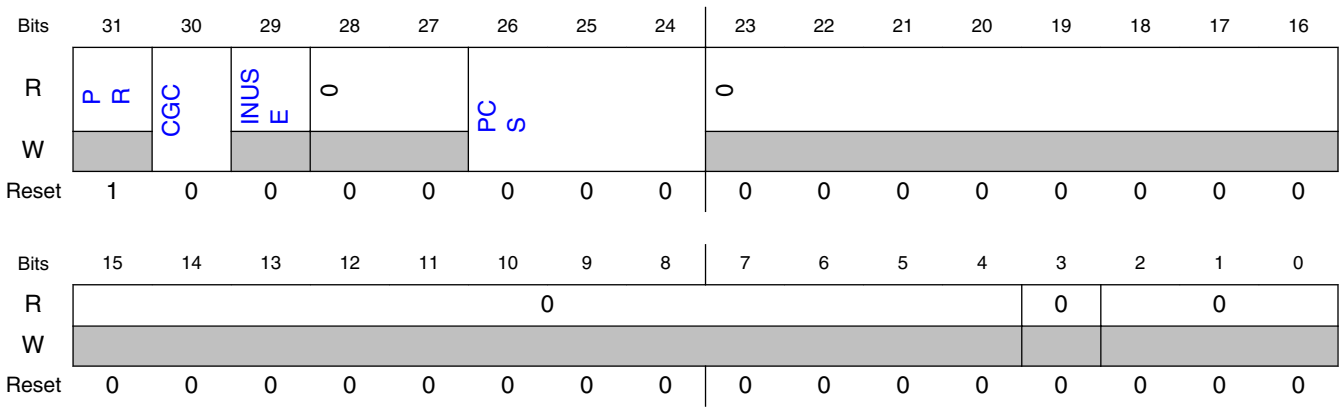
25.6.17.1 Offset

Register	Offset
PCC_FLEXIO0	C8h

25.6.17.2 Function

This register is for the FLEXIO0 module.

25.6.17.3 Diagram



25.6.17.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.18 PCC LPI2C0 Register (PCC_LPI2C0)

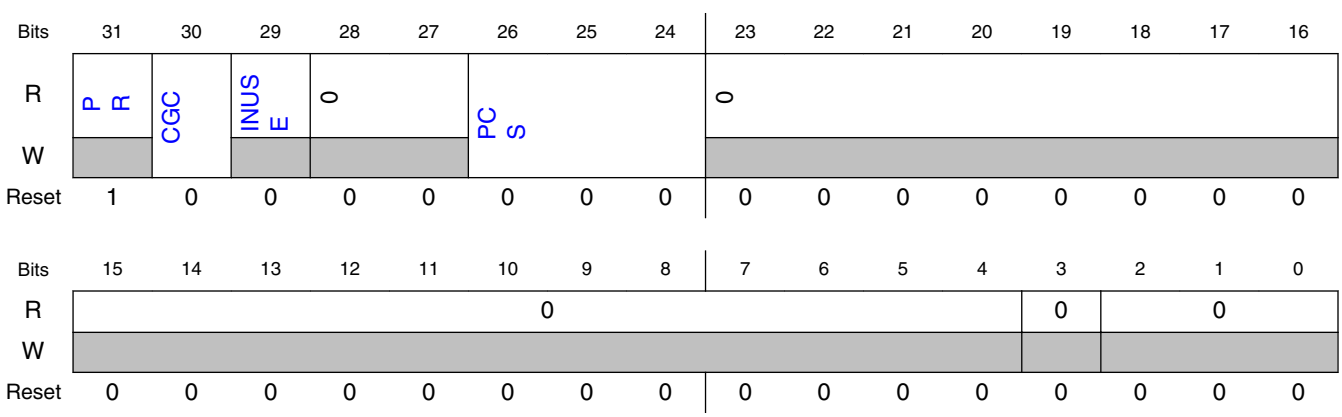
25.6.18.1 Offset

Register	Offset
PCC_LPI2C0	CCh

25.6.18.2 Function

This register is for the LPI2C0 module.

25.6.18.3 Diagram



25.6.18.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.19 PCC LPI2C1 Register (PCC_LPI2C1)

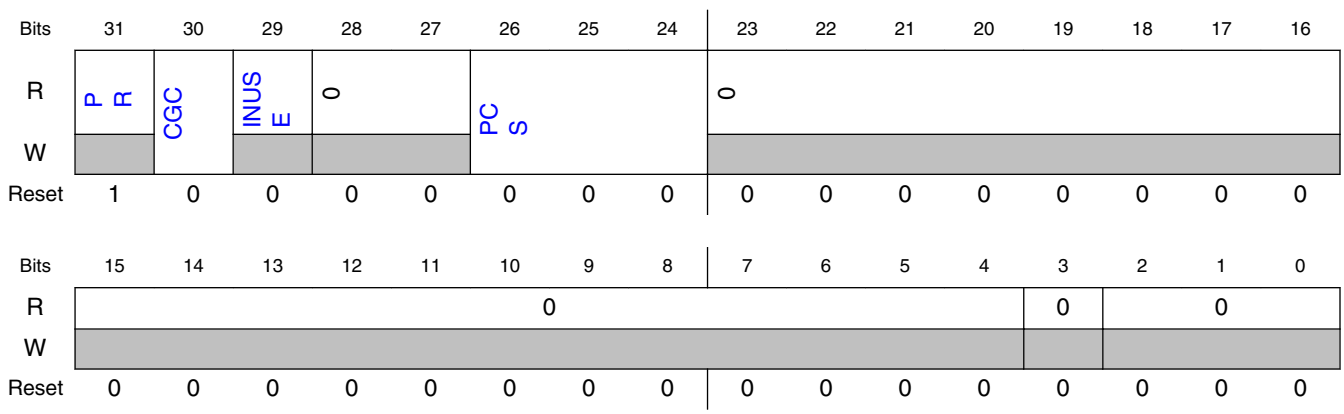
25.6.19.1 Offset

Register	Offset
PCC_LPI2C1	D0h

25.6.19.2 Function

This register is for the LPI2C1 module.

25.6.19.3 Diagram



25.6.19.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.20 PCC LPI2C2 Register (PCC_LPI2C2)

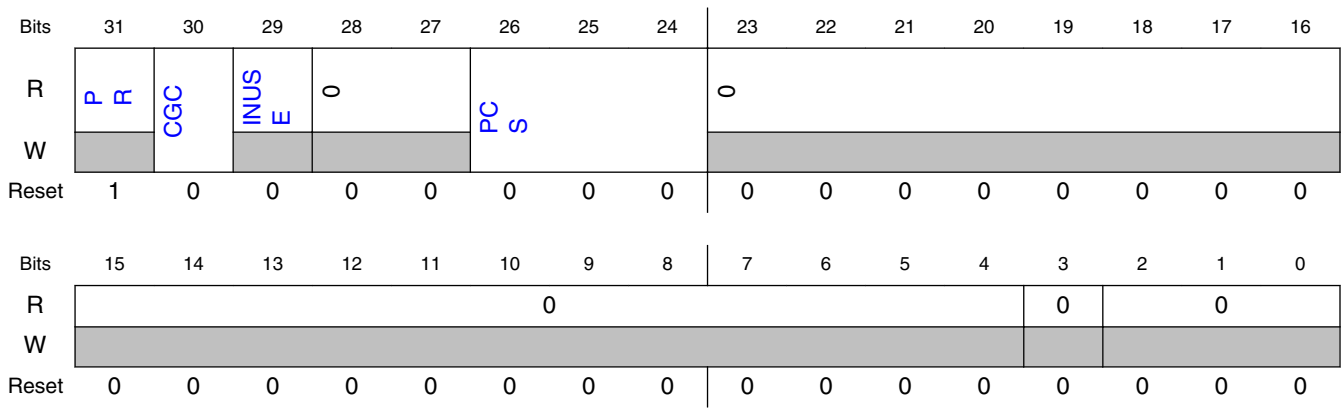
25.6.20.1 Offset

Register	Offset
PCC_LPI2C2	D4h

25.6.20.2 Function

This register is for the LPI2C2 module.

25.6.20.3 Diagram



25.6.20.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	Peripheral Clock Source Select

Table continues on the next page...

Field	Function
PCS	This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.21 PCC LPI2C3 Register (PCC_LPI2C3)

25.6.21.1 Offset

Register	Offset
PCC_LPI2C3	D8h

25.6.21.2 Function

This register is for the LPI2C3 module.

25.6.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.21.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.22 PCC SAI0 Register (PCC_SAI0)

25.6.22.1 Offset

Register	Offset
PCC_SAI0	DCh

25.6.22.2 Function

This register is for the SAI0 module.

25.6.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			FRAC							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.22.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. An external clock can be enabled for this peripheral. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-16 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 00000000b - Fractional value is 0. 00000001b - Fractional value is 1.
15-0 PCD	Peripheral Clock Divider Select

Field	Function
	This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked.

25.6.23 PCC LPSPI0 Register (PCC_LPSPI0)

25.6.23.1 Offset

Register	Offset
PCC_LPSPI0	E0h

25.6.23.2 Function

This register is for the LPSPI0 module.

25.6.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.23.4 Fields

Field	Function
31	Present

Table continues on the next page...

Field	Function
PR	This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.24 PCC LPSPI1 Register (PCC_LPSPI1)

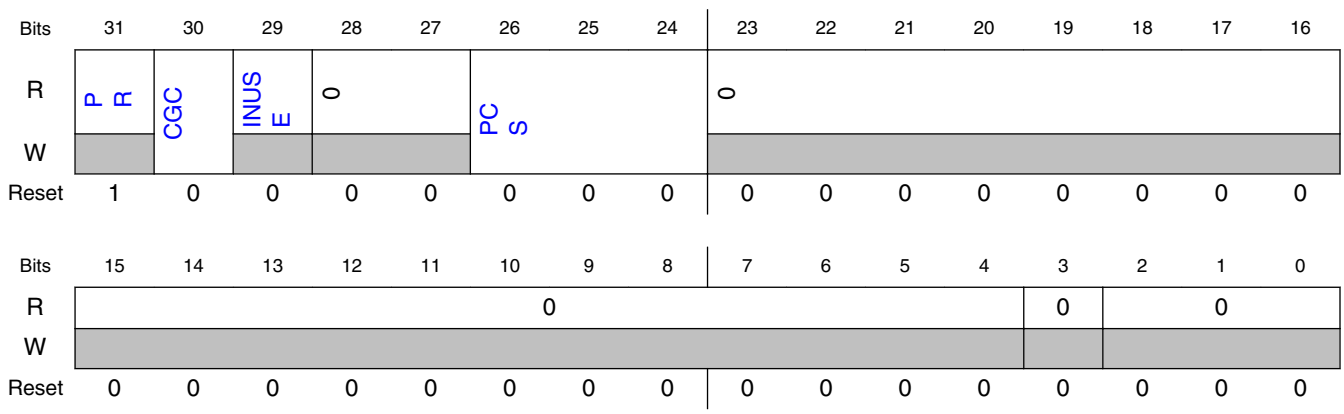
25.6.24.1 Offset

Register	Offset
PCC_LPSPI1	E4h

25.6.24.2 Function

This register is for the LPSPI1 module.

25.6.24.3 Diagram



25.6.24.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag

Table continues on the next page...

Field	Function
	This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.25 PCC LPUART0 Register (PCC_LPUART0)

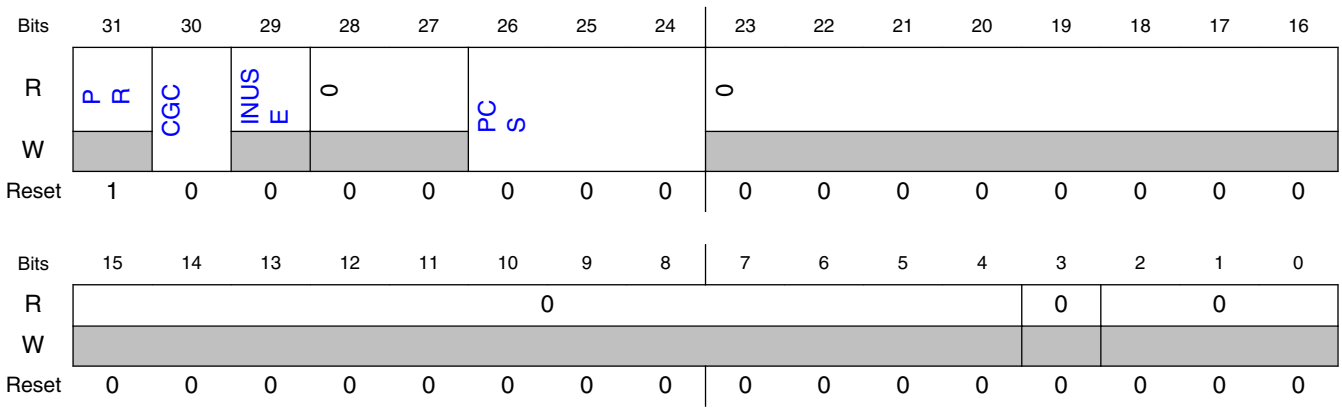
25.6.25.1 Offset

Register	Offset
PCC_LPUART0	E8h

25.6.25.2 Function

This register is for the LPUART0 module.

25.6.25.3 Diagram



25.6.25.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.26 PCC LPUART1 Register (PCC_LPUART1)

25.6.26.1 Offset

Register	Offset
PCC_LPUART1	ECh

25.6.26.2 Function

This register is for the LPUART1 module.

25.6.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.26.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.27 PCC PCTLA Register (PCC_PCTLA)

25.6.27.1 Offset

Register	Offset
PCC_PCTLA	FCh

25.6.27.2 Function

This register is for the PCTLA module.

25.6.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.27.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.28 PCC PCTLB Register (PCC_PCTLB)

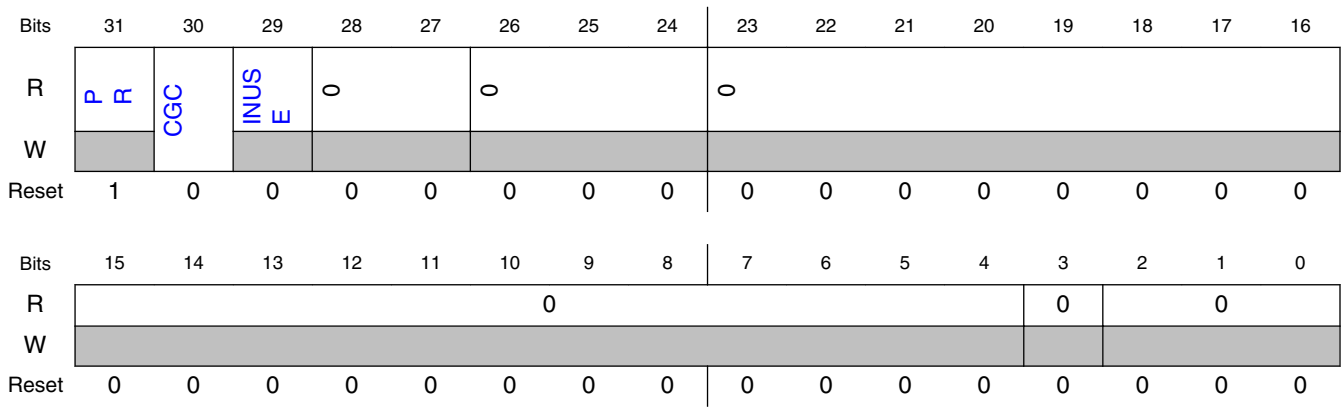
25.6.28.1 Offset

Register	Offset
PCC_PCTLB	100h

25.6.28.2 Function

This register is for the PCTLB module.

25.6.28.3 Diagram



25.6.28.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.29 PCC ADC0 Register (PCC_ADC0)

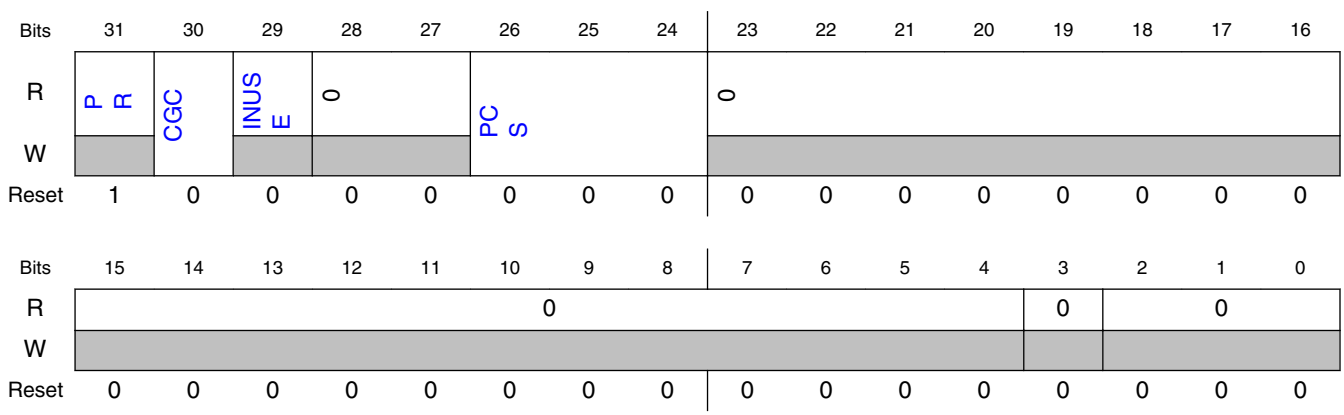
25.6.29.1 Offset

Register	Offset
PCC_ADC0	104h

25.6.29.2 Function

This register is for the ADC0 module.

25.6.29.3 Diagram



25.6.29.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.30 PCC CMP0 Register (PCC_CMP0)

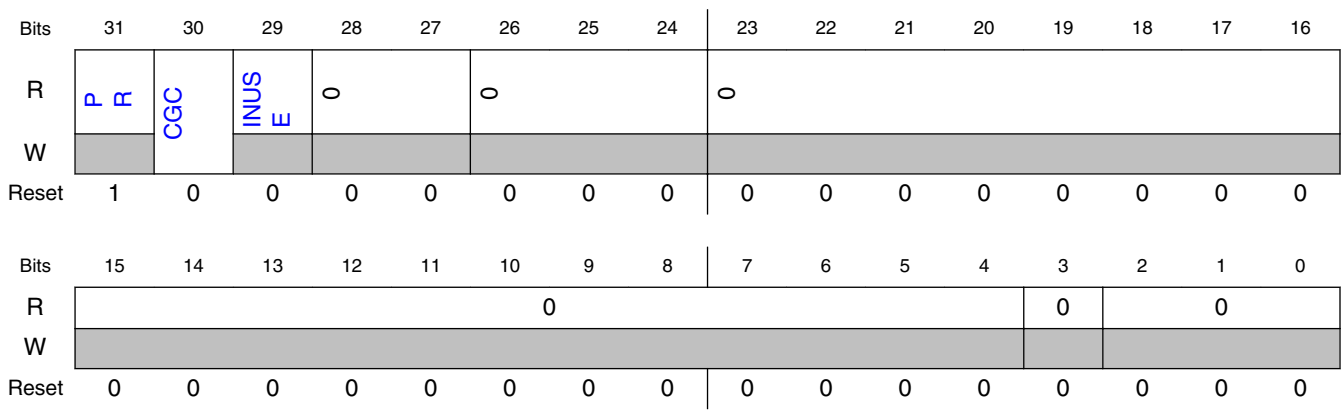
25.6.30.1 Offset

Register	Offset
PCC_CMP0	108h

25.6.30.2 Function

This register is for the CMP0 module.

25.6.30.3 Diagram



25.6.30.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag

Table continues on the next page...

Field	Function
	This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.31 PCC CMP1 Register (PCC_CMP1)

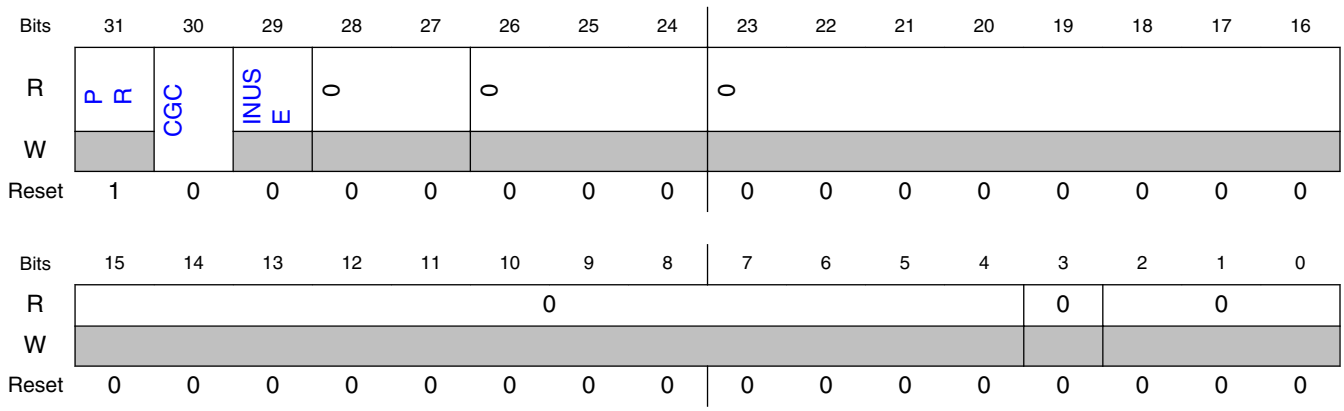
25.6.31.1 Offset

Register	Offset
PCC_CMP1	10Ch

25.6.31.2 Function

This register is for the CMP1 module.

25.6.31.3 Diagram



25.6.31.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
2-0 —	This read-only bit field is reserved and always has the value 0.

25.6.32 PCC DAC0 Register (PCC_DAC0)

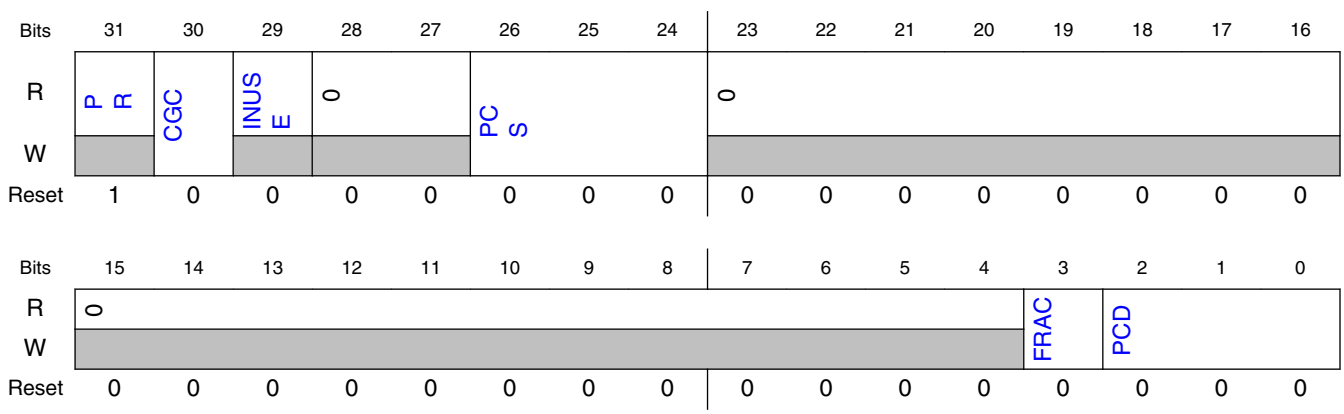
25.6.32.1 Offset

Register	Offset
PCC_DAC0	110h

25.6.32.2 Function

This register is for the DAC0 module.

25.6.32.3 Diagram



25.6.32.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device.

Table continues on the next page...

Field	Function
	0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1.

Field	Function
	001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.6.33 PCC DAC1 Register (PCC_DAC1)

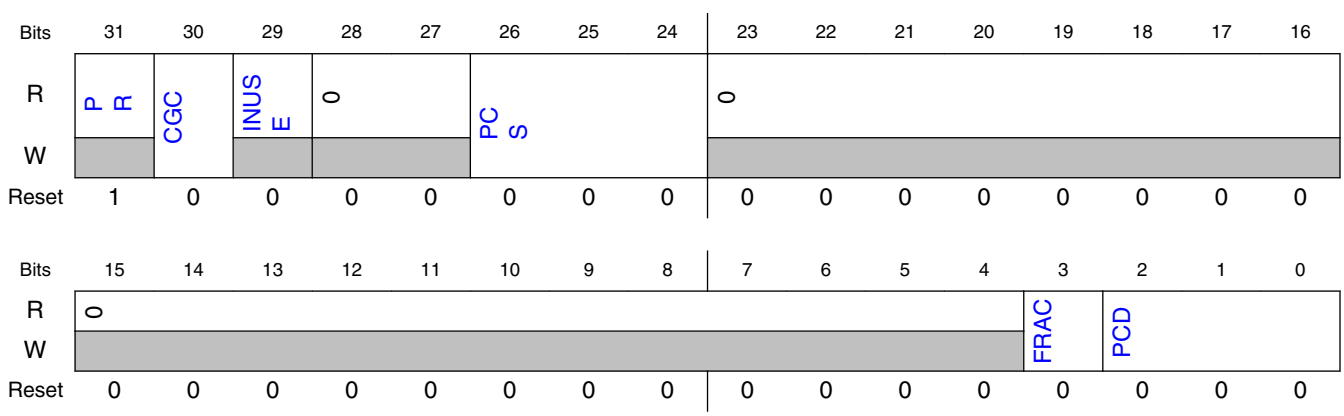
25.6.33.1 Offset

Register	Offset
PCC_DAC1	114h

25.6.33.2 Function

This register is for the DAC1 module.

25.6.33.3 Diagram



25.6.33.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0	Peripheral Clock Divider Select

Field	Function
PCD	This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.6.34 PCC SNVS Register (PCC_SNVS)

25.6.34.1 Offset

Register	Offset
PCC_SNVS	1C0h

25.6.34.2 Function

This register is for the SNVS module.

25.6.34.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.6.34.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.7 PCC register descriptions

25.7.1 PCC Memory map

PCC1 base address: 410B_2000h

Offset	Register	Width (In bits)	Access	Reset value
50h	PCC TPIU Register (PCC_TPIU)	32	RW	8000_0000h
94h	PCC QSPI_OTFAD Register (PCC_QSPI_OTFAD)	32	RW	8000_0000h
A0h	PCC TPM2 Register (PCC_TPM2)	32	RW	8000_0000h
A4h	PCC TPM3 Register (PCC_TPM3)	32	RW	8000_0000h
A8h	PCC SAI1 Register (PCC_SAI1)	32	RW	8000_0000h
ACh	PCC LPUART2 Register (PCC_LPUART2)	32	RW	8000_0000h
B0h	PCC LPUART3 Register (PCC_LPUART3)	32	RW	8000_0000h
B4h	PCC ADC1 Register (PCC_ADC1)	32	RW	8000_0000h

25.7.2 PCC TPIU Register (PCC_TPIU)

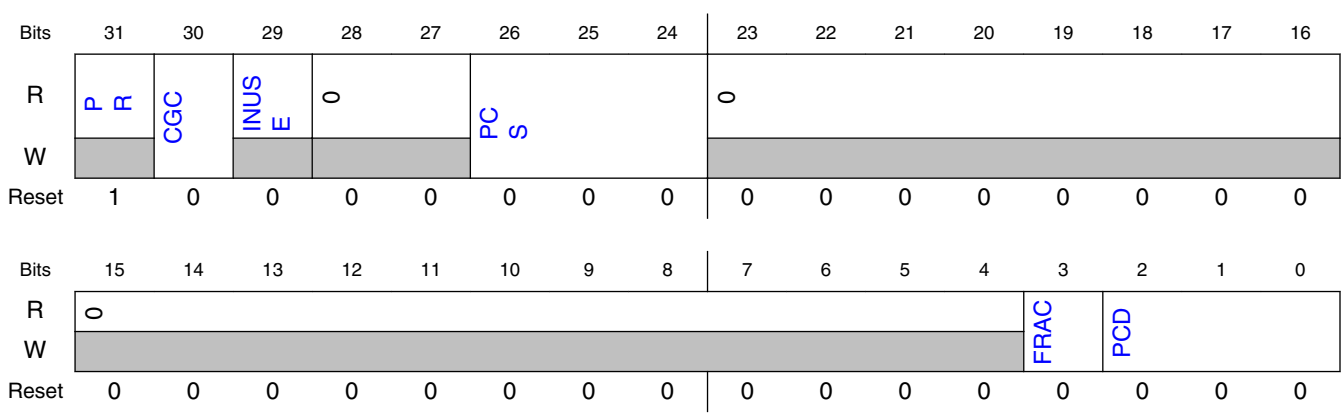
25.7.2.1 Offset

Register	Offset
PCC_TPIU	50h

25.7.2.2 Function

This register is for the TPIU module.

25.7.2.3 Diagram



25.7.2.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0	Peripheral Clock Divider Select

Field	Function
PCD	This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.7.3 PCC QSPI_OTFAD Register (PCC_QSPI_OTFAD)

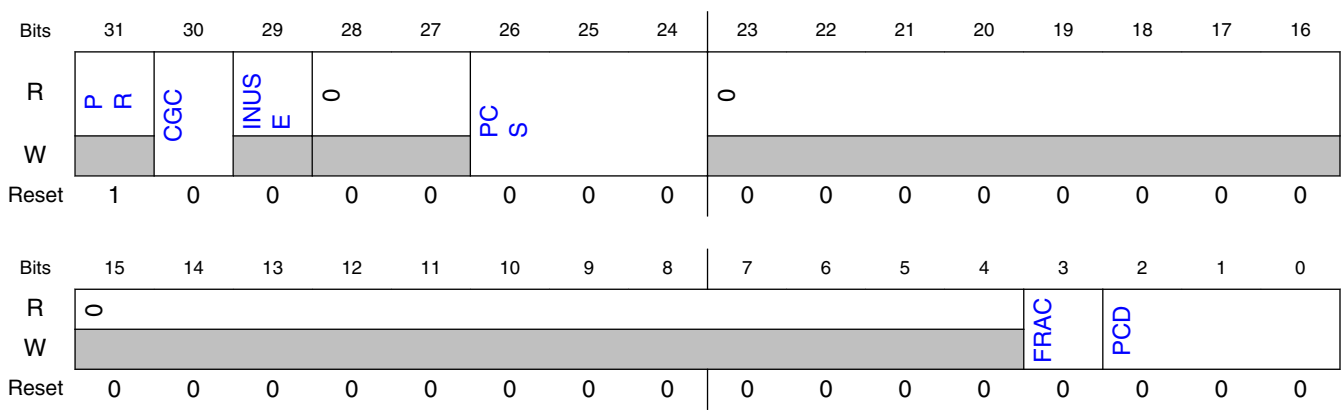
25.7.3.1 Offset

Register	Offset
PCC_QSPI_OTFAD	94h

25.7.3.2 Function

This register is for the QSPI_OTFAD module.

25.7.3.3 Diagram



25.7.3.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.

Table continues on the next page...

Field	Function
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.7.4 PCC TPM2 Register (PCC_TPM2)

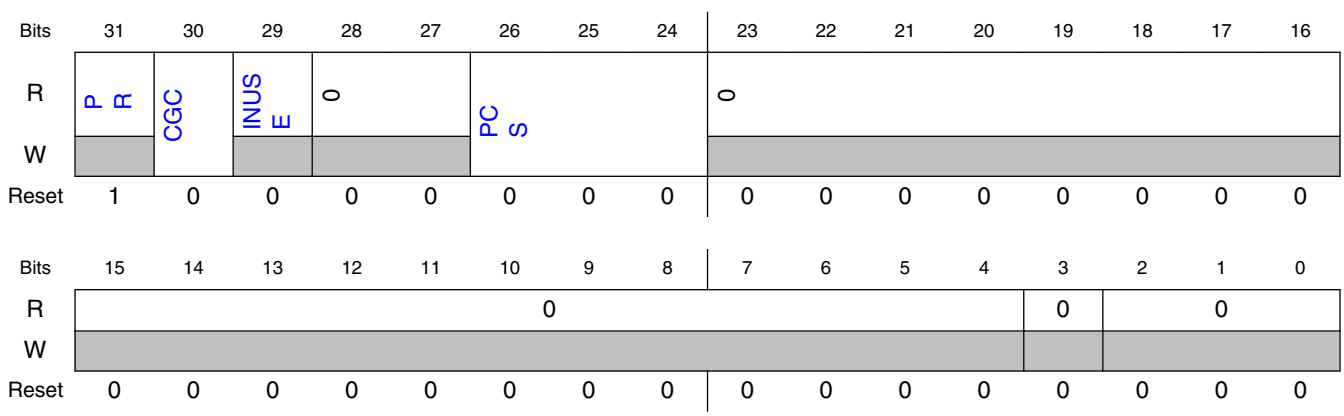
25.7.4.1 Offset

Register	Offset
PCC_TPM2	A0h

25.7.4.2 Function

This register is for the TPM2 module.

25.7.4.3 Diagram



25.7.4.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.7.5 PCC TPM3 Register (PCC_TPM3)

25.7.5.1 Offset

Register	Offset
PCC_TPM3	A4h

25.7.5.2 Function

This register is for the TPM3 module.

25.7.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.7.5.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.7.6 PCC SAI1 Register (PCC_SAI1)

25.7.6.1 Offset

Register	Offset
PCC_SAI1	A8h

25.7.6.2 Function

This register is for the SAI1 module.

25.7.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			FRAC							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.7.6.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	Peripheral Clock Source Select

Table continues on the next page...

Field	Function
PCS	This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. An external clock can be enabled for this peripheral. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-16 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 00000000b - Fractional value is 0. 00000001b - Fractional value is 1.
15-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked.

25.7.7 PCC LPUART2 Register (PCC_LPUART2)

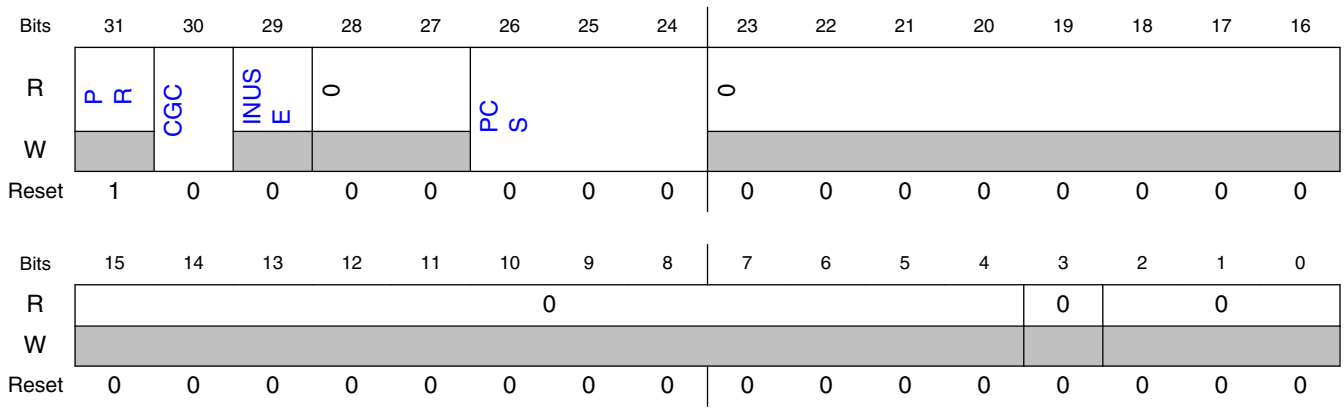
25.7.7.1 Offset

Register	Offset
PCC_LPUART2	ACh

25.7.7.2 Function

This register is for the LPUART2 module.

25.7.7.3 Diagram



25.7.7.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.7.8 PCC LPUART3 Register (PCC_LPUART3)

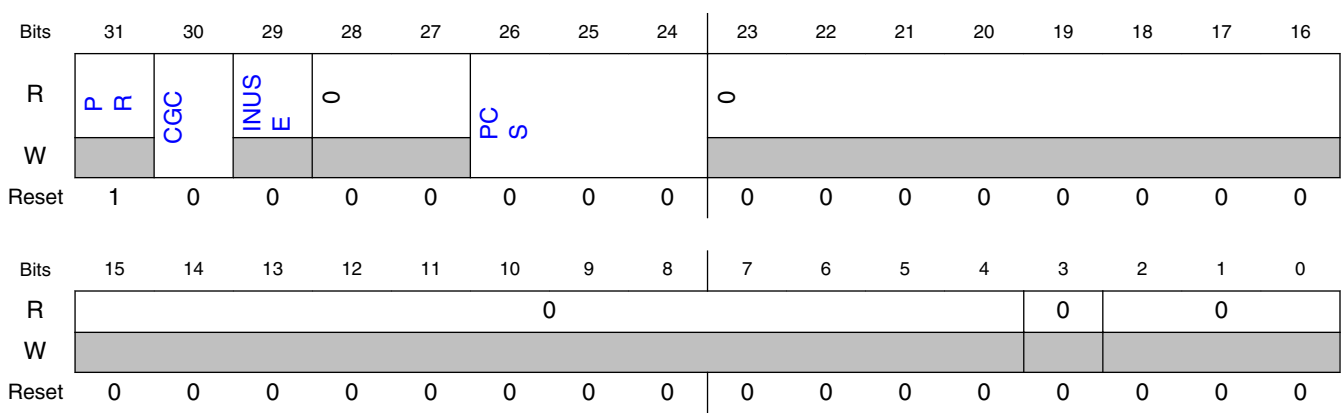
25.7.8.1 Offset

Register	Offset
PCC_LPUART3	B0h

25.7.8.2 Function

This register is for the LPUART3 module.

25.7.8.3 Diagram



25.7.8.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.7.9 PCC ADC1 Register (PCC_ADC1)

25.7.9.1 Offset

Register	Offset
PCC_ADC1	B4h

25.7.9.2 Function

This register is for the ADC1 module.

25.7.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.7.9.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8 PCC register descriptions

25.8.1 PCC Memory map

PCC2 base address: 403F_0000h

Offset	Register	Width (In bits)	Access	Reset value
20h	PCC DMA1 Register (PCC_DMA1)	32	RW	8000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
3Ch	PCC RGPIO2P1 Register (PCC_RGPIO2P1)	32	RW	8000_0000h
40h	PCC FLEXBUS Register (PCC_FLEXBUS)	32	RW	8000_0000h
6Ch	PCC SEMA42_1 Register (PCC_SEMA42_1)	32	RW	8000_0000h
84h	PCC DMA_MUX1 Register (PCC_DMA_MUX1)	32	RW	8000_0000h
90h	PCC CAAM Register (PCC_CAAM)	32	RW	8000_0000h
94h	PCC TPM4 Register (PCC_TPM4)	32	RW	8000_0000h
98h	PCC TPM5 Register (PCC_TPM5)	32	RW	8000_0000h
9Ch	PCC LPIT1 Register (PCC_LPIT1)	32	RW	8000_0000h
A4h	PCC LPSPI2 Register (PCC_LPSPI2)	32	RW	8000_0000h
A8h	PCC LPSPI3 Register (PCC_LPSPI3)	32	RW	8000_0000h
ACh	PCC LPI2C4 Register (PCC_LPI2C4)	32	RW	8000_0000h
B0h	PCC LPI2C5 Register (PCC_LPI2C5)	32	RW	8000_0000h
B4h	PCC LPUART4 Register (PCC_LPUART4)	32	RW	8000_0000h
B8h	PCC LPUART5 Register (PCC_LPUART5)	32	RW	8000_0000h
C4h	PCC FLEXIO1 Register (PCC_FLEXIO1)	32	RW	8000_0000h
CCh	PCC USB0 Register (PCC_USB0)	32	RW	8000_0000h
D0h	PCC USB1 Register (PCC_USB1)	32	ROZ	8000_0000h
D4h	PCC USB_PHY Register (PCC_USB_PHY)	32	RW	8000_0000h
D8h	PCC USB_PL301 Register (PCC_USB_PL301)	32	ROZ	8000_0000h
DCh	PCC USDHC0 Register (PCC_USDHC0)	32	RW	8000_0000h
E0h	PCC USDHC1 Register (PCC_USDHC1)	32	RW	8000_0000h
F4h	PCC WDOG1 Register (PCC_WDOG1)	32	RW	C000_0000h
10Ch	PCC WDOG2 Register (PCC_WDOG2)	32	RW	C000_0000h

25.8.2 PCC DMA1 Register (PCC_DMA1)

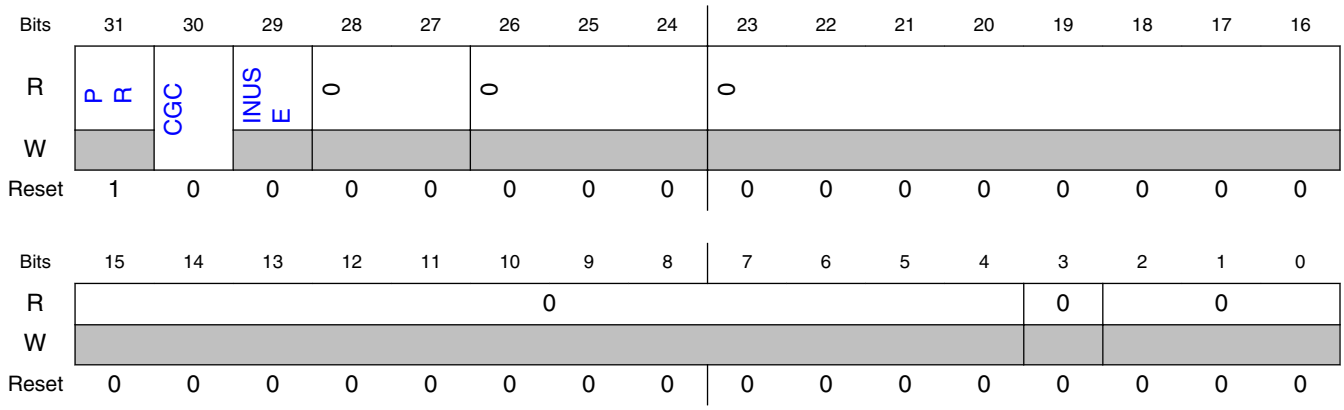
25.8.2.1 Offset

Register	Offset
PCC_DMA1	20h

25.8.2.2 Function

This register is for the DMA1 module.

25.8.2.3 Diagram



25.8.2.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.3 PCC RGPIO2P1 Register (PCC_RGPIO2P1)

25.8.3.1 Offset

Register	Offset
PCC_RGPIO2P1	3Ch

25.8.3.2 Function

This register is for the RGPIO2P1 module.

25.8.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.8.3.4 Fields

Field	Function
31	Present

Table continues on the next page...

Field	Function
PR	This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.4 PCC FLEXBUS Register (PCC_FLEXBUS)

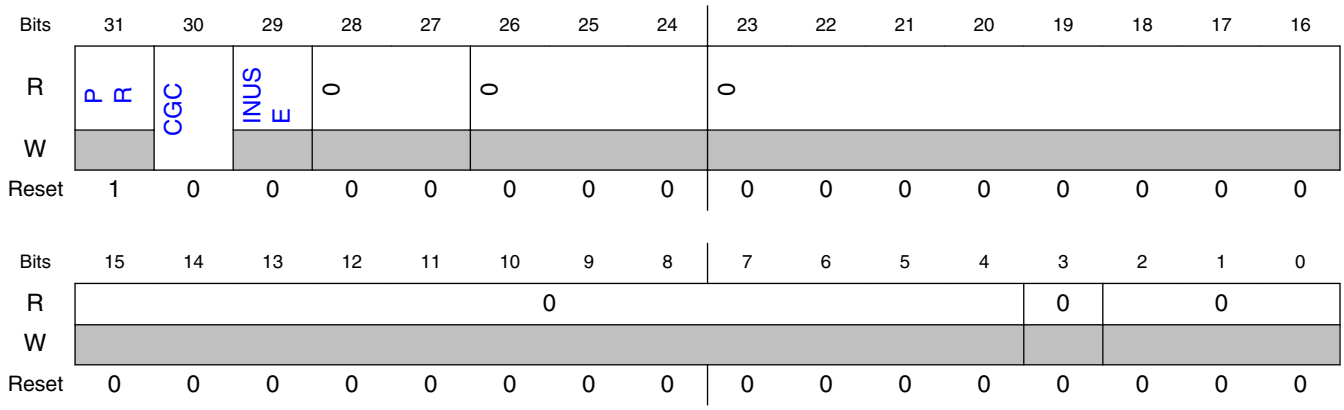
25.8.4.1 Offset

Register	Offset
PCC_FLEXBUS	40h

25.8.4.2 Function

This register is for the FLEXBUS module.

25.8.4.3 Diagram



25.8.4.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.5 PCC SEMA42_1 Register (PCC_SEMA42_1)

25.8.5.1 Offset

Register	Offset
PCC_SEMA42_1	6Ch

25.8.5.2 Function

This register is for the SEMA42_1 module.

25.8.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.8.5.4 Fields

Field	Function
31	Present

Table continues on the next page...

Field	Function
PR	This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.6 PCC DMA_MUX1 Register (PCC_DMA_MUX1)

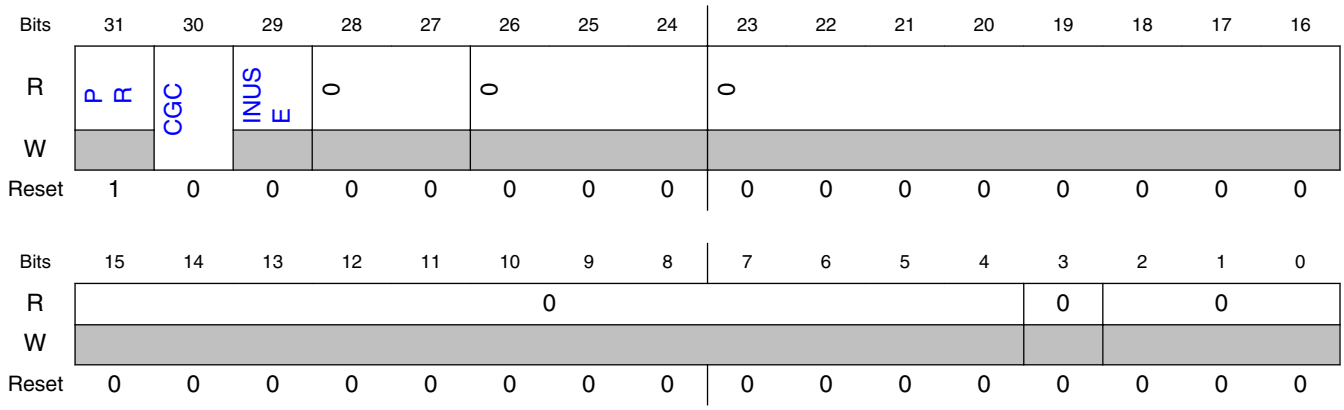
25.8.6.1 Offset

Register	Offset
PCC_DMA_MUX1	84h

25.8.6.2 Function

This register is for the DMA_MUX1 module.

25.8.6.3 Diagram



25.8.6.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.7 PCC CAAM Register (PCC_CAAM)

25.8.7.1 Offset

Register	Offset
PCC_CAAM	90h

25.8.7.2 Function

This register is for the CAAM module.

25.8.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.8.7.4 Fields

Field	Function
31	Present

Table continues on the next page...

Field	Function
PR	This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.8 PCC TPM4 Register (PCC_TPM4)

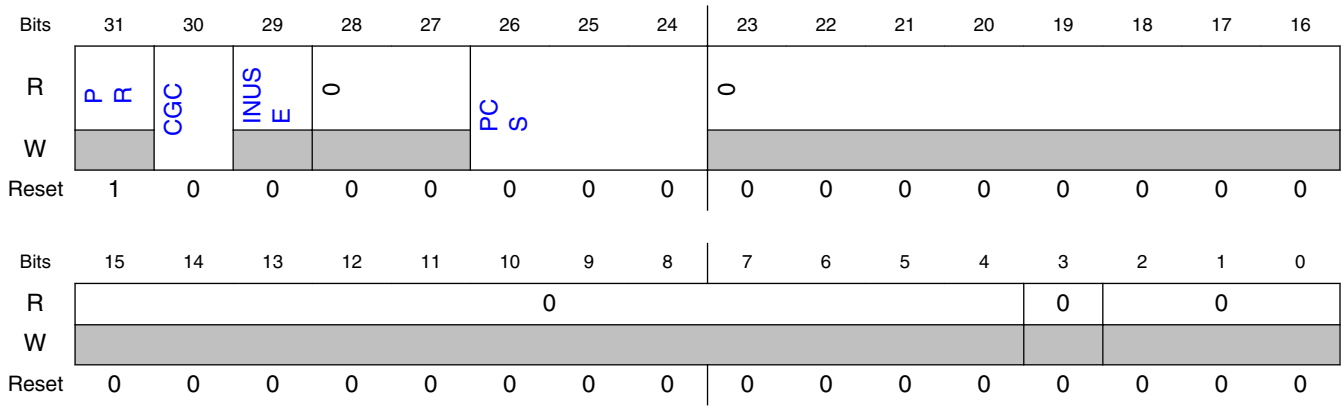
25.8.8.1 Offset

Register	Offset
PCC_TPM4	94h

25.8.8.2 Function

This register is for the TPM4 module.

25.8.8.3 Diagram



25.8.8.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off.

Table continues on the next page...

Field	Function
	001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.9 PCC TPM5 Register (PCC_TPM5)

25.8.9.1 Offset

Register	Offset
PCC_TPM5	98h

25.8.9.2 Function

This register is for the TPM5 module.

25.8.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.8.9.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.10 PCC LPIT1 Register (PCC_LPIT1)

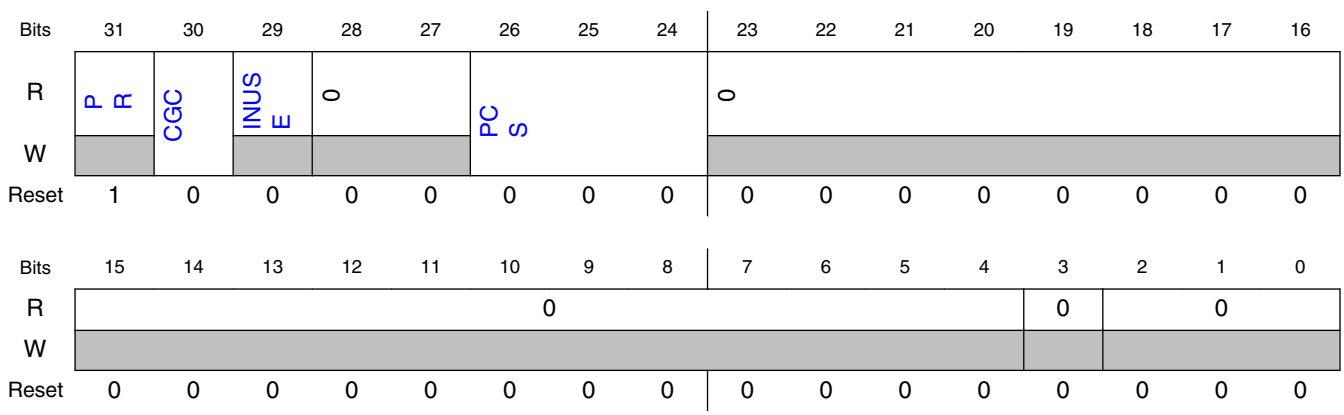
25.8.10.1 Offset

Register	Offset
PCC_LPIT1	9Ch

25.8.10.2 Function

This register is for the LPIT1 module.

25.8.10.3 Diagram



25.8.10.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified.

Table continues on the next page...

Field	Function
	0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.11 PCC LPSPI2 Register (PCC_LPSPI2)

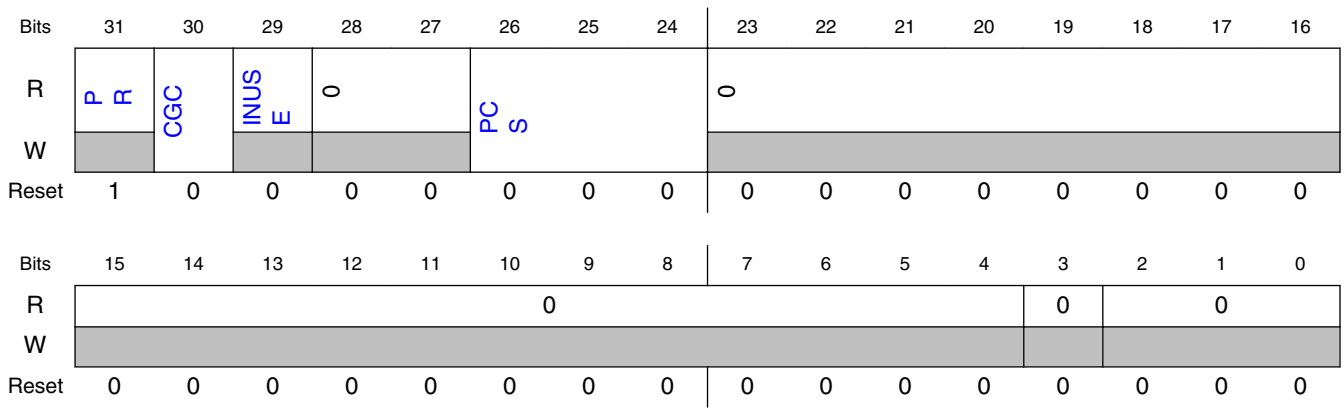
25.8.11.1 Offset

Register	Offset
PCC_LPSPI2	A4h

25.8.11.2 Function

This register is for the LPSPI2 module.

25.8.11.3 Diagram



25.8.11.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	Peripheral Clock Source Select

Table continues on the next page...

Field	Function
PCS	This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.12 PCC LPSPI3 Register (PCC_LPSPI3)

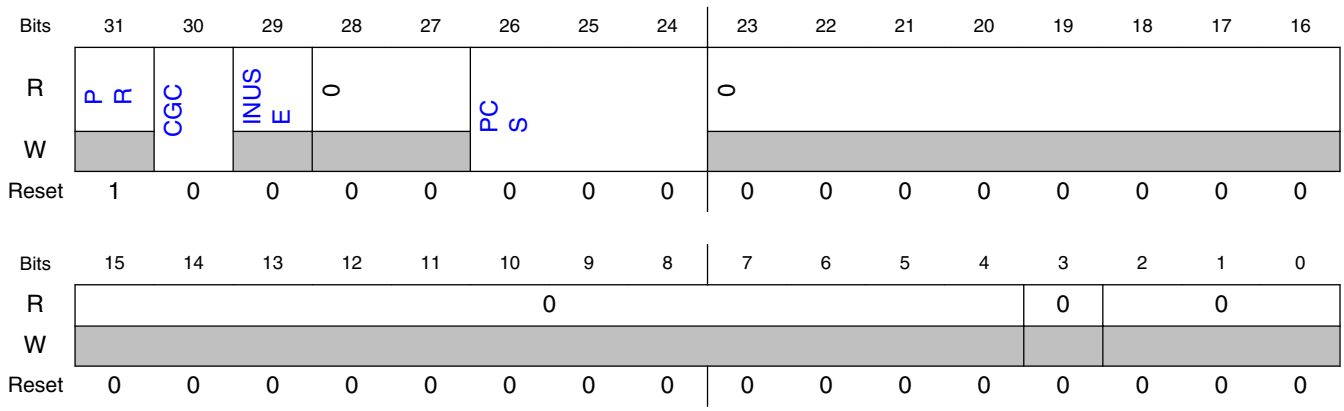
25.8.12.1 Offset

Register	Offset
PCC_LPSPI3	A8h

25.8.12.2 Function

This register is for the LPSPI3 module.

25.8.12.3 Diagram



25.8.12.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.13 PCC LPI2C4 Register (PCC_LPI2C4)

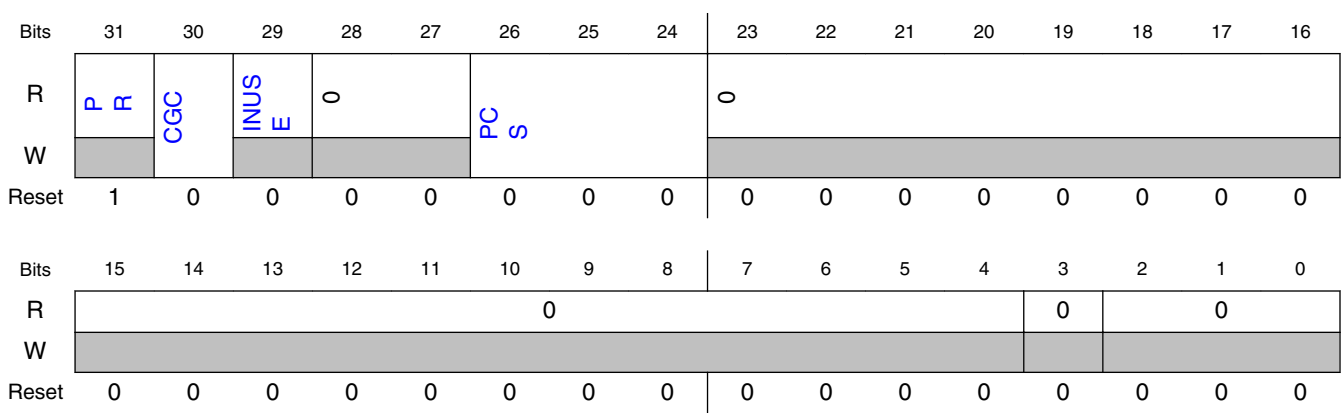
25.8.13.1 Offset

Register	Offset
PCC_LPI2C4	ACh

25.8.13.2 Function

This register is for the LPI2C4 module.

25.8.13.3 Diagram



25.8.13.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.14 PCC LPI2C5 Register (PCC_LPI2C5)

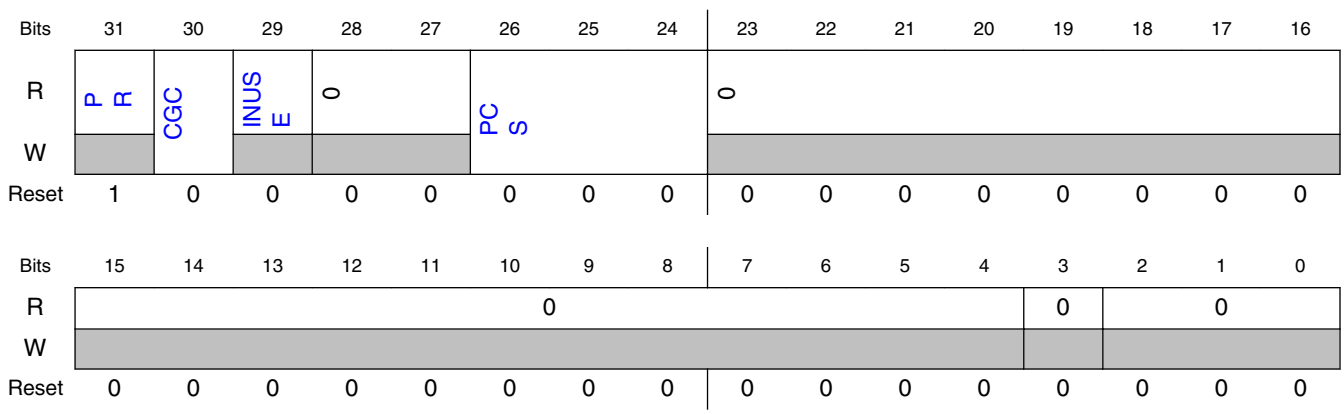
25.8.14.1 Offset

Register	Offset
PCC_LPI2C5	B0h

25.8.14.2 Function

This register is for the LPI2C5 module.

25.8.14.3 Diagram



25.8.14.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.15 PCC LPUART4 Register (PCC_LPUART4)

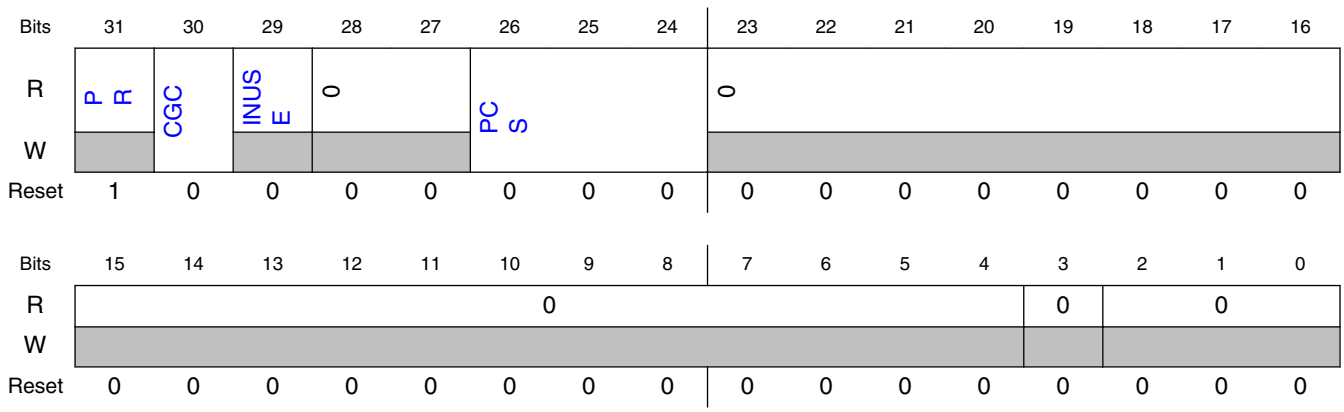
25.8.15.1 Offset

Register	Offset
PCC_LPUART4	B4h

25.8.15.2 Function

This register is for the LPUART4 module.

25.8.15.3 Diagram



25.8.15.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	Peripheral Clock Source Select

Table continues on the next page...

Field	Function
PCS	This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.16 PCC LPUART5 Register (PCC_LPUART5)

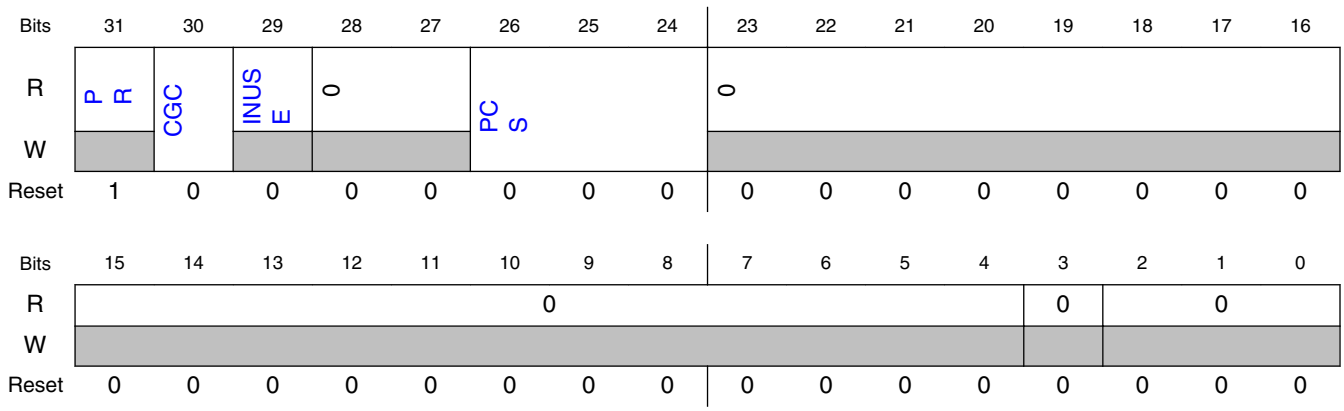
25.8.16.1 Offset

Register	Offset
PCC_LPUART5	B8h

25.8.16.2 Function

This register is for the LPUART5 module.

25.8.16.3 Diagram



25.8.16.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.17 PCC FLEXIO1 Register (PCC_FLEXIO1)

25.8.17.1 Offset

Register	Offset
PCC_FLEXIO1	C4h

25.8.17.2 Function

This register is for the FLEXIO1 module.

25.8.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.8.17.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.18 PCC USB0 Register (PCC_USB0)

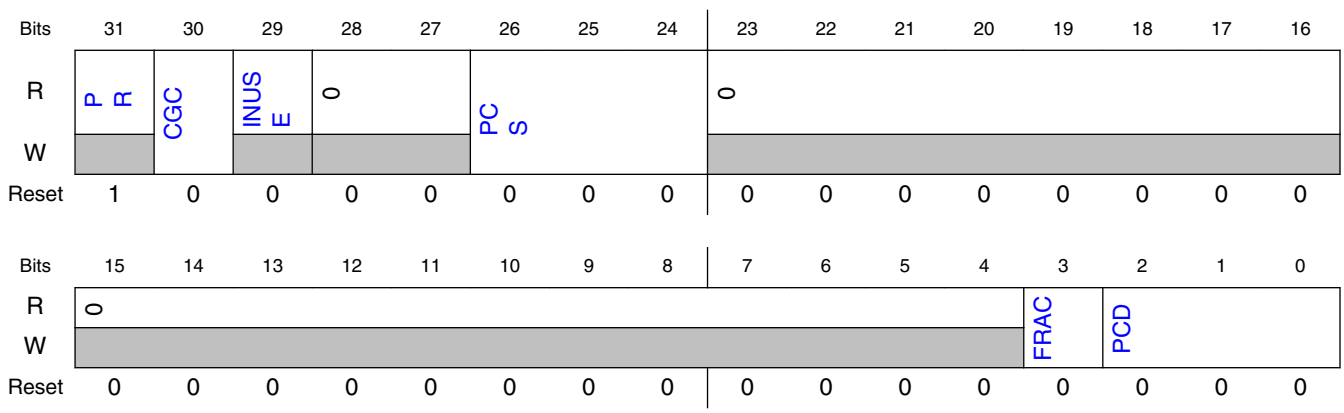
25.8.18.1 Offset

Register	Offset
PCC_USB0	CCh

25.8.18.2 Function

This register is for the USB0 module.

25.8.18.3 Diagram



25.8.18.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. An external clock can be enabled for this peripheral. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.8.19 PCC USB1 Register (PCC_USB1)

25.8.19.1 Offset

Register	Offset
PCC_USB1	D0h

25.8.19.2 Function

This register is for the USB1 module.

25.8.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.8.19.4 Fields

Field	Function
31-0 —	Reserved.

25.8.20 PCC USB_PHY Register (PCC_USB_PHY)

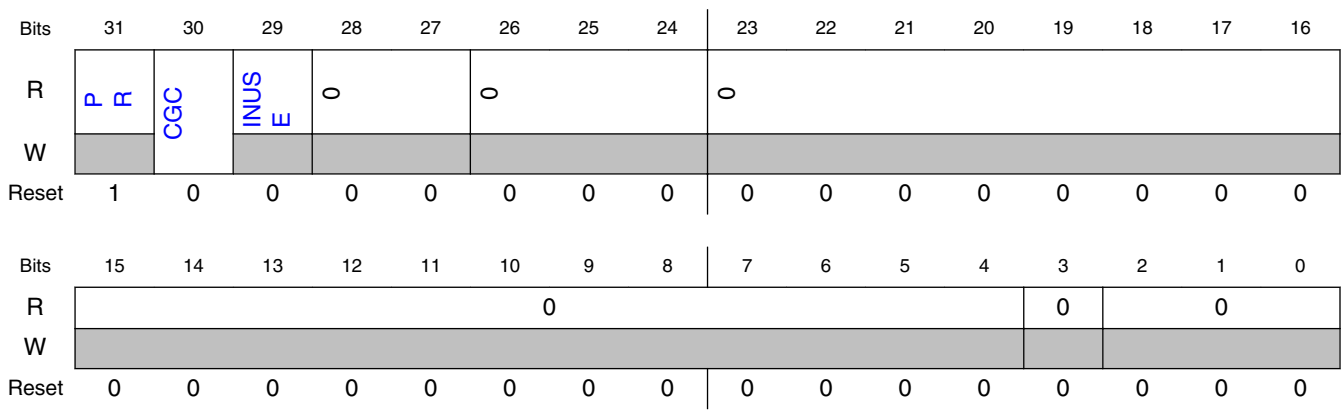
25.8.20.1 Offset

Register	Offset
PCC_USB_PHY	D4h

25.8.20.2 Function

This register is for the USB_PHY module.

25.8.20.3 Diagram



25.8.20.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag

Table continues on the next page...

Field	Function
	This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.8.21 PCC USB_PL301 Register (PCC_USB_PL301)

25.8.21.1 Offset

Register	Offset
PCC_USB_PL301	D8h

25.8.21.2 Function

This register is for the USB_PL301 module.

25.8.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.8.21.4 Fields

Field	Function
31-0 —	Reserved.

25.8.22 PCC USDHC0 Register (PCC_USDHC0)

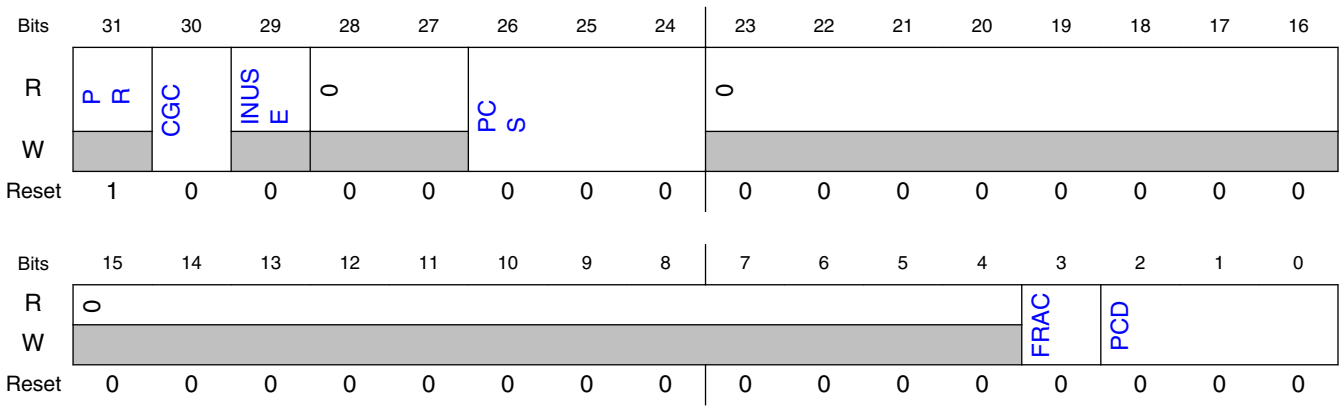
25.8.22.1 Offset

Register	Offset
PCC_USDHC0	DCh

25.8.22.2 Function

This register is for the USDHC0 module.

25.8.22.3 Diagram



25.8.22.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. An external clock can be enabled for this peripheral. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.8.23 PCC USDHC1 Register (PCC_USDHC1)

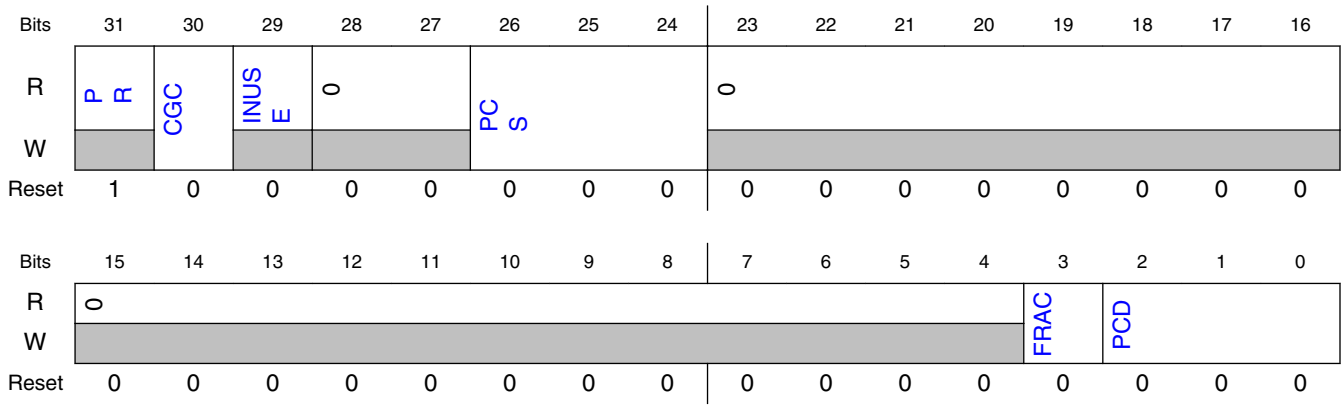
25.8.23.1 Offset

Register	Offset
PCC_USDHC1	E0h

25.8.23.2 Function

This register is for the USDHC1 module.

25.8.23.3 Diagram



25.8.23.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. An external clock can be enabled for this peripheral.

Table continues on the next page...

Field	Function
	001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.8.24 PCC WDOG1 Register (PCC_WDOG1)

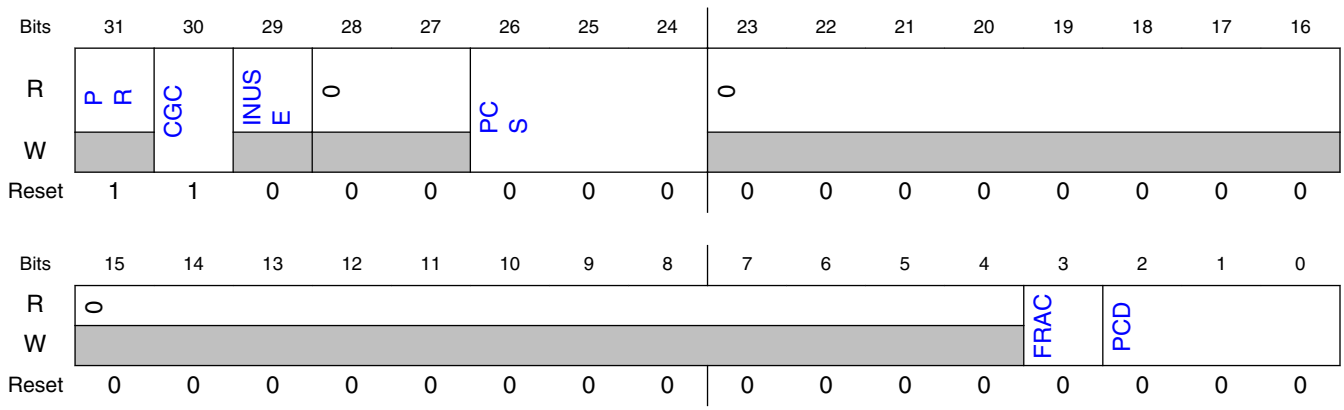
25.8.24.1 Offset

Register	Offset
PCC_WDOG1	F4h

25.8.24.2 Function

This register is for the WDOG1 module.

25.8.24.3 Diagram



25.8.24.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	Peripheral Clock Source Select

Table continues on the next page...

Field	Function
PCS	This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.8.25 PCC WDOG2 Register (PCC_WDOG2)

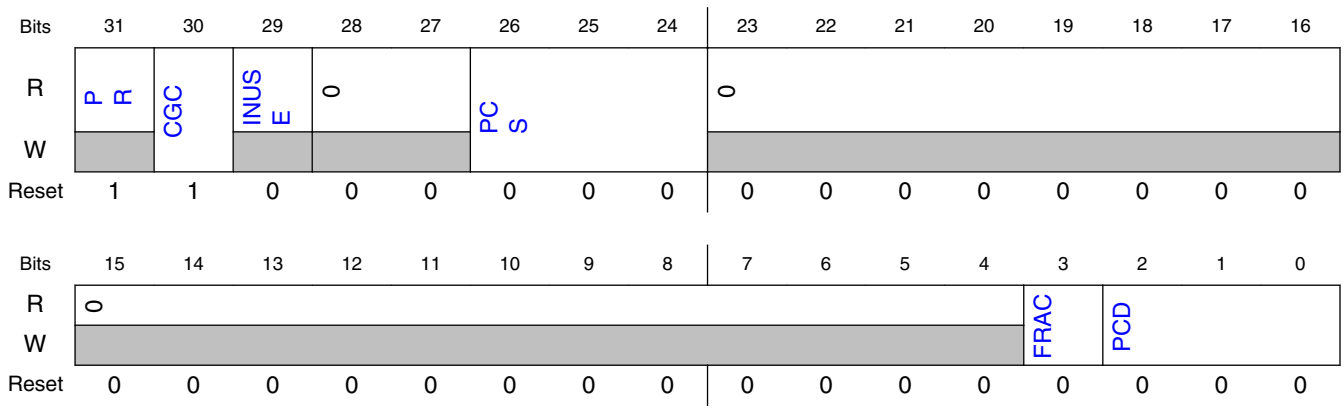
25.8.25.1 Offset

Register	Offset
PCC_WDOG2	10Ch

25.8.25.2 Function

This register is for the WDOG2 module.

25.8.25.3 Diagram



25.8.25.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.9 PCC register descriptions

25.9.1 PCC Memory map

PCC3 base address: 40B3_0000h

Offset	Register	Width (In bits)	Access	Reset value
84h	PCC TPM6 Register (PCC_TPM6)	32	RW	8000_0000h
88h	PCC TPM7 Register (PCC_TPM7)	32	RW	8000_0000h
90h	PCC LPI2C6 Register (PCC_LPI2C6)	32	RW	8000_0000h
94h	PCC LPI2C7 Register (PCC_LPI2C7)	32	RW	8000_0000h
98h	PCC LPUART6 Register (PCC_LPUART6)	32	RW	8000_0000h
9Ch	PCC LPUART7 Register (PCC_LPUART7)	32	RW	8000_0000h
A0h	PCC VIU Register (PCC_VIU)	32	RW	8000_0000h
A4h	PCC DSI Register (PCC_DSI)	32	RW	8000_0000h
A8h	PCC LCDIF Register (PCC_LCDIF)	32	RW	8000_0000h
ACh	PCC MMDC Register (PCC_MMDC)	32	RW	8000_0000h
B8h	PCC PCTLC Register (PCC_PCTLC)	32	RW	8000_0000h
BCh	PCC PCTLD Register (PCC_PCTLD)	32	RW	8000_0000h
C0h	PCC PCTLE Register (PCC_PCTLE)	32	RW	8000_0000h
C4h	PCC PCTLF Register (PCC_PCTLF)	32	RW	8000_0000h
140h	PCC GPU3D Register (PCC_GPU3D)	32	RW	8000_0000h
144h	PCC GPU2D Register (PCC_GPU2D)	32	RW	8000_0000h

25.9.2 PCC TPM6 Register (PCC_TPM6)

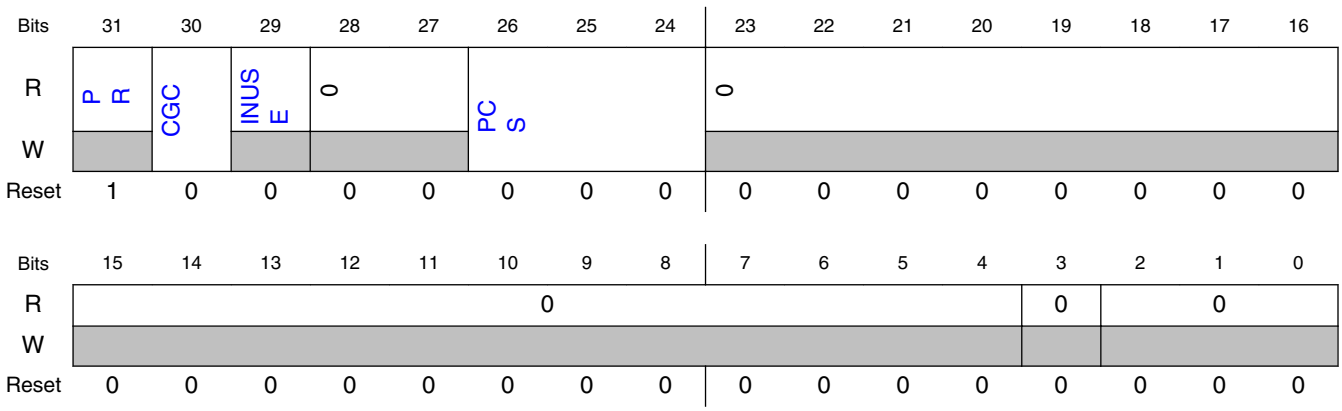
25.9.2.1 Offset

Register	Offset
PCC_TPM6	84h

25.9.2.2 Function

This register is for the TPM6 module.

25.9.2.3 Diagram



25.9.2.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.3 PCC TPM7 Register (PCC_TPM7)

25.9.3.1 Offset

Register	Offset
PCC_TPM7	88h

25.9.3.2 Function

This register is for the TPM7 module.

25.9.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.9.3.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.4 PCC LPI2C6 Register (PCC_LPI2C6)

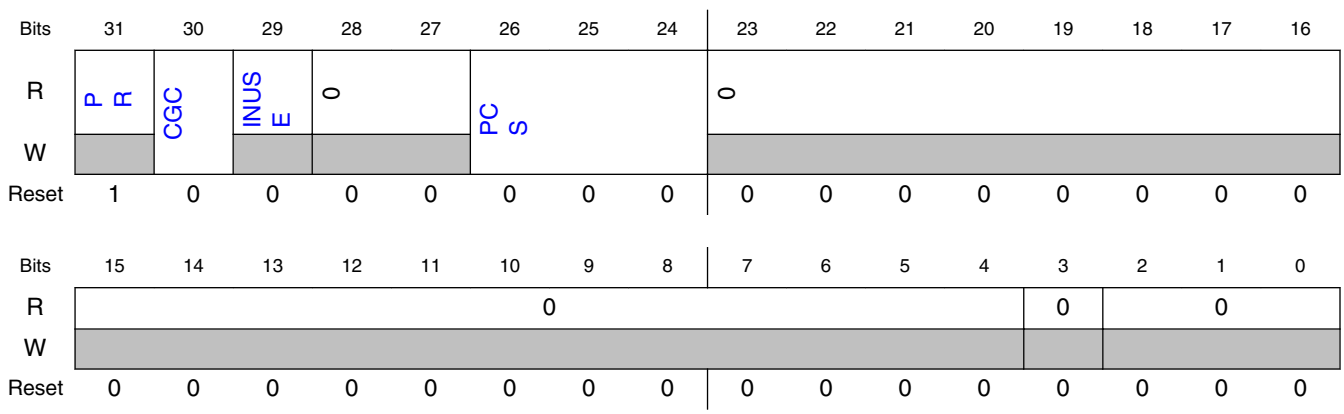
25.9.4.1 Offset

Register	Offset
PCC_LPI2C6	90h

25.9.4.2 Function

This register is for the LPI2C6 module.

25.9.4.3 Diagram



25.9.4.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.5 PCC LPI2C7 Register (PCC_LPI2C7)

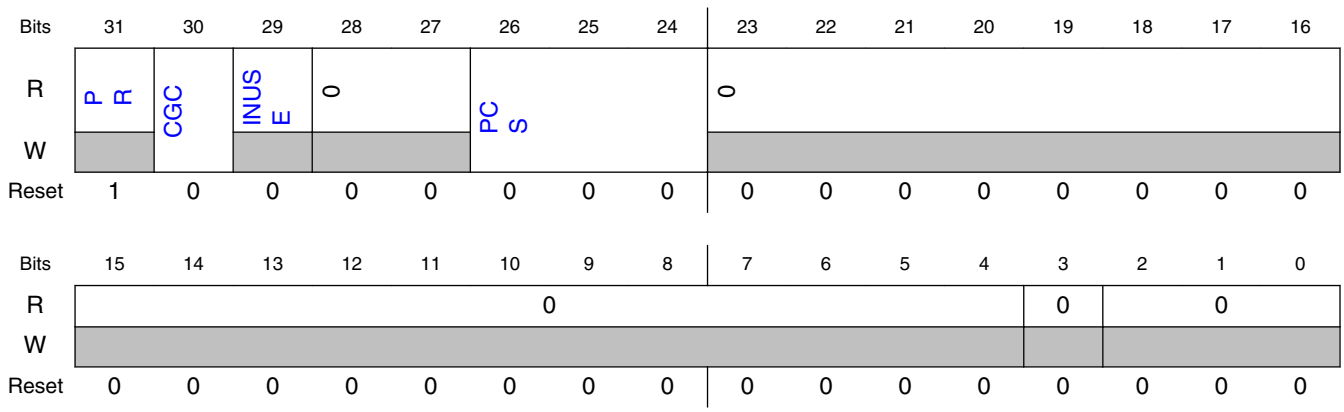
25.9.5.1 Offset

Register	Offset
PCC_LPI2C7	94h

25.9.5.2 Function

This register is for the LPI2C7 module.

25.9.5.3 Diagram



25.9.5.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	Peripheral Clock Source Select

Table continues on the next page...

Field	Function
PCS	This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.6 PCC LPUART6 Register (PCC_LPUART6)

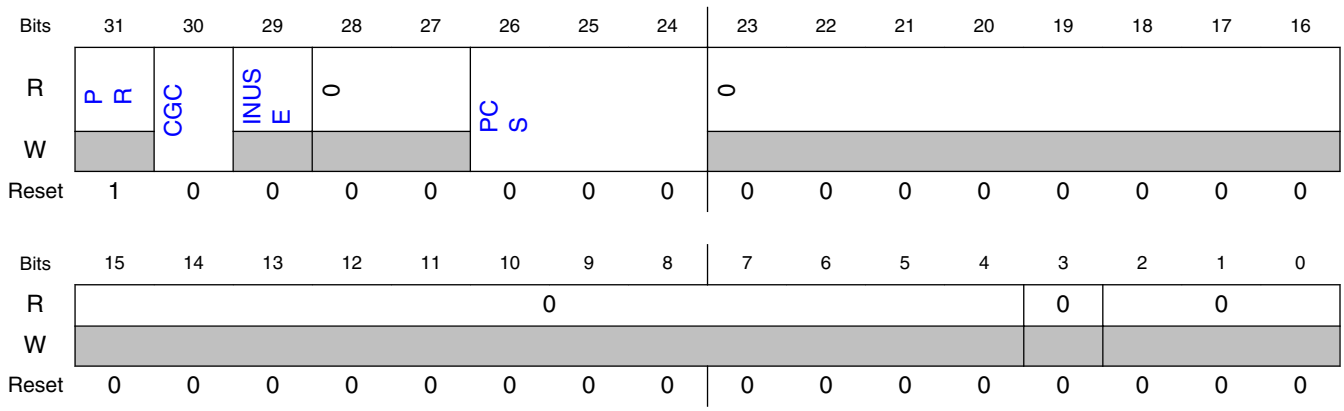
25.9.6.1 Offset

Register	Offset
PCC_LPUART6	98h

25.9.6.2 Function

This register is for the LPUART6 module.

25.9.6.3 Diagram



25.9.6.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.7 PCC LPUART7 Register (PCC_LPUART7)

25.9.7.1 Offset

Register	Offset
PCC_LPUART7	9Ch

25.9.7.2 Function

This register is for the LPUART7 module.

25.9.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.9.7.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.8 PCC VIU Register (PCC_VIU)

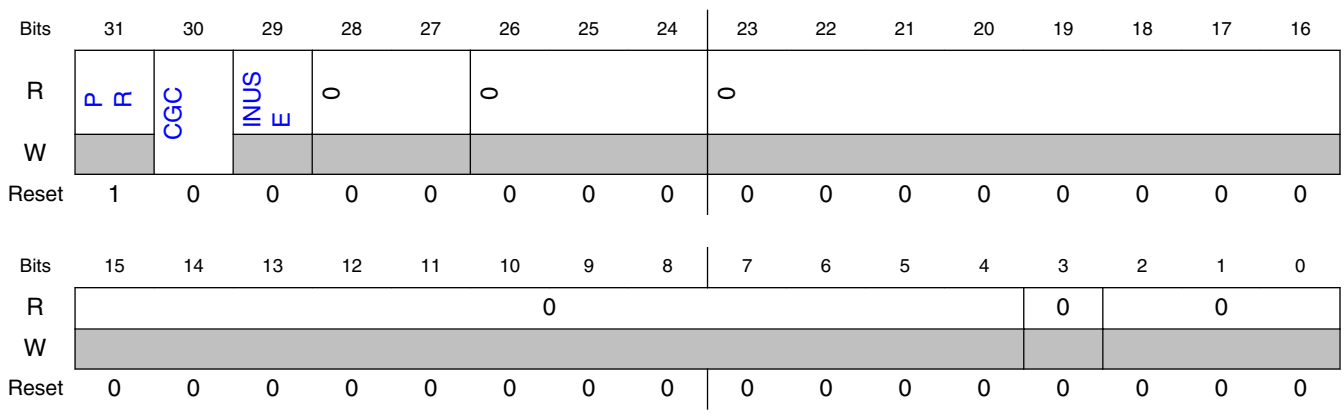
25.9.8.1 Offset

Register	Offset
PCC_VIU	A0h

25.9.8.2 Function

This register is for the VIU module.

25.9.8.3 Diagram



25.9.8.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.9 PCC DSI Register (PCC_DSI)

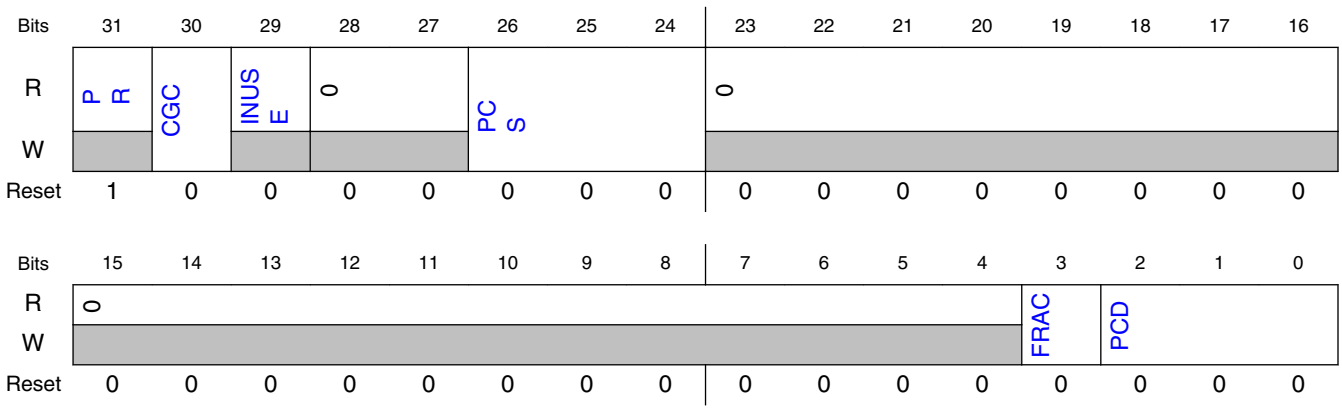
25.9.9.1 Offset

Register	Offset
PCC_DSI	A4h

25.9.9.2 Function

This register is for the DSI module.

25.9.9.3 Diagram



25.9.9.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. An external clock can be enabled for this peripheral. 001b - Clock option 1

Table continues on the next page...

Field	Function
	010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.9.10 PCC LCDIF Register (PCC_LCDIF)

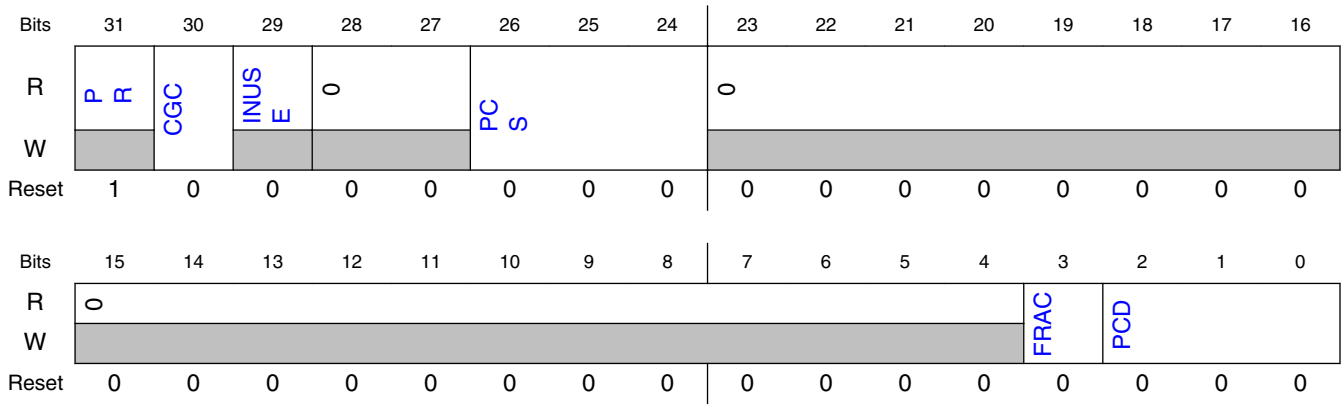
25.9.10.1 Offset

Register	Offset
PCC_LCDIF	A8h

25.9.10.2 Function

This register is for the LCDIF module.

25.9.10.3 Diagram



25.9.10.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. An external clock can be enabled for this peripheral.

Table continues on the next page...

Field	Function
	001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 FRAC	Peripheral Clock Divider Fraction This read/write bit field sets the fraction multiply value for the fractional clock divider used as a clock source. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. **NOTE:** When dividing by 1 (PCD = 000), do not set the FRAC bit; otherwise, the output clock is disabled. 0b - Fractional value is 0. 1b - Fractional value is 1.
2-0 PCD	Peripheral Clock Divider Select This read/write bit field is used for peripherals that require a clock divider. Divider output clock = Divider input clock x [(FRAC+1)/(PCD+1)]. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Divide by 1. 001b - Divide by 2. 010b - Divide by 3. 011b - Divide by 4. 100b - Divide by 5. 101b - Divide by 6. 110b - Divide by 7. 111b - Divide by 8.

25.9.11 PCC MMDC Register (PCC_MMDC)

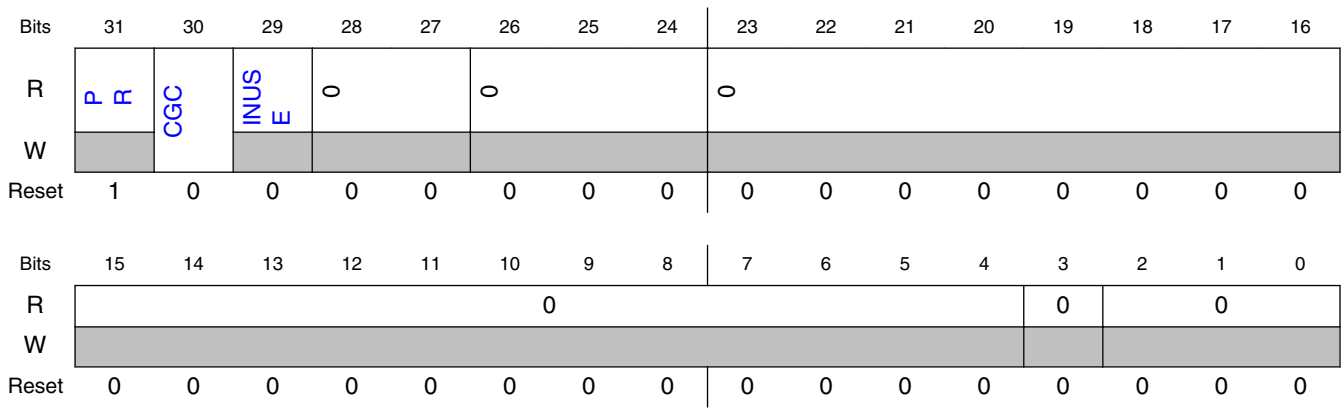
25.9.11.1 Offset

Register	Offset
PCC_MMDC	ACh

25.9.11.2 Function

This register is for the MMDC module.

25.9.11.3 Diagram



25.9.11.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.12 PCC PCTLC Register (PCC_PCTLC)

25.9.12.1 Offset

Register	Offset
PCC_PCTLC	B8h

25.9.12.2 Function

This register is for the PCTLC module.

25.9.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.9.12.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.13 PCC PCTLD Register (PCC_PCTLD)

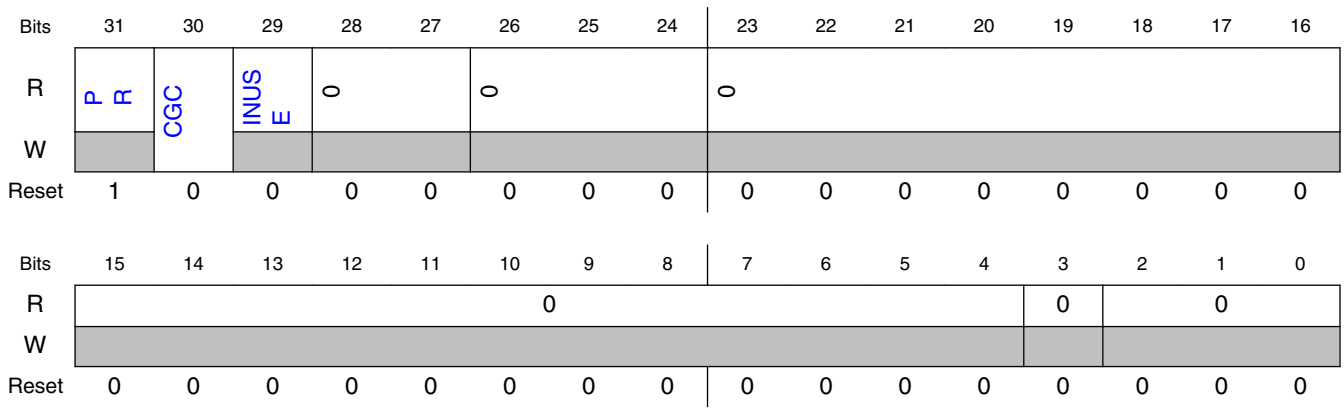
25.9.13.1 Offset

Register	Offset
PCC_PCTLD	BCh

25.9.13.2 Function

This register is for the PCTLD module.

25.9.13.3 Diagram



25.9.13.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.14 PCC PCTLE Register (PCC_PCTLE)

25.9.14.1 Offset

Register	Offset
PCC_PCTLE	C0h

25.9.14.2 Function

This register is for the PCTLE module.

25.9.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		0			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.9.14.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.15 PCC PCTLF Register (PCC_PCTLF)

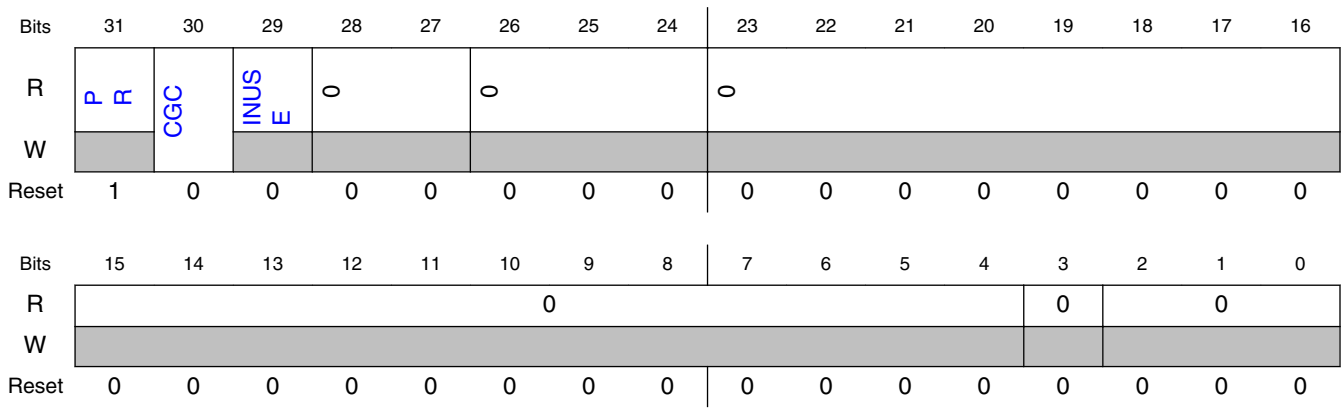
25.9.15.1 Offset

Register	Offset
PCC_PCTLF	C4h

25.9.15.2 Function

This register is for the PCTLF module.

25.9.15.3 Diagram



25.9.15.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
26-24 —	This read-only bit field is reserved and always has the value 0.
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.16 PCC GPU3D Register (PCC_GPU3D)

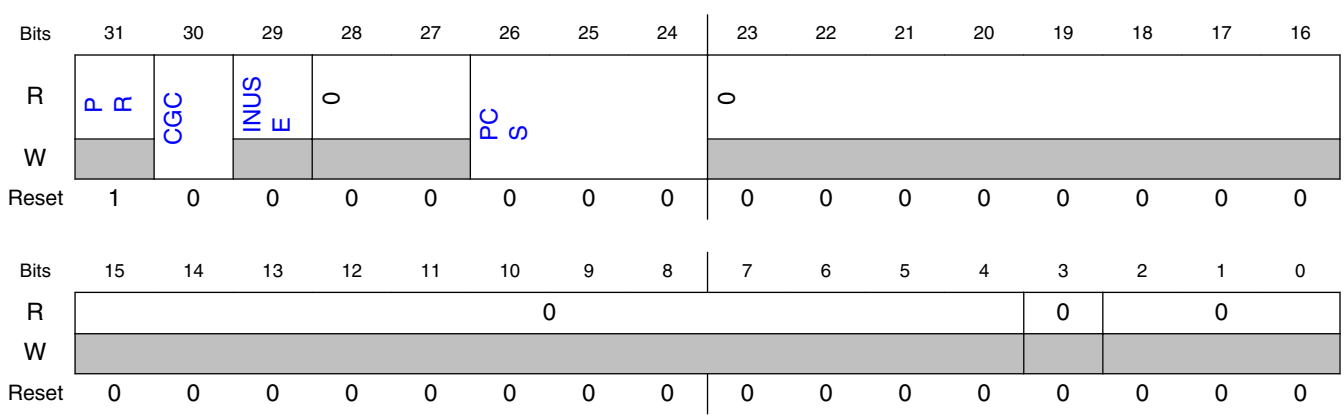
25.9.16.1 Offset

Register	Offset
PCC_GPU3D	140h

25.9.16.2 Function

This register is for the GPU3D module.

25.9.16.3 Diagram



25.9.16.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified. 1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

25.9.17 PCC GPU2D Register (PCC_GPU2D)

25.9.17.1 Offset

Register	Offset
PCC_GPU2D	144h

25.9.17.2 Function

This register is for the GPU2D module.

25.9.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PR	CGC	INUSE	0		PCS			0							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

25.9.17.4 Fields

Field	Function
31 PR	Present This bit shows whether the peripheral is present on this device. 0b - Peripheral is not present. 1b - Peripheral is present.
30 CGC	Clock Gate Control This read/write bit enables the interface clock for the peripheral, allowing access to the module's registers. It also controls whether the clock selection and divider options can be modified. 0b - Clock disabled. The current clock selection and divider options are not locked and can be modified.

Table continues on the next page...

Field	Function
	1b - Clock enabled. The current clock selection and divider options are locked and cannot be modified.
29 INUSE	In use flag This read-only bit shows that this peripheral is being used by another core. That is, software on another core has already configured the clocking options of this peripheral. For example, if CPU0 software has set this peripheral's CGC bit first, it sees the INUSE bit as cleared. However, if software on another core has set this peripheral's CGC bit first, CPU0 software sees the INUSE bit as set. 0b - Peripheral is not being used. 1b - Peripheral is being used. Software cannot modify the existing clocking configuration.
28-27 —	This read-only bit field is reserved and always has the value 0.
26-24 PCS	Peripheral Clock Source Select This read/write bit field is used for peripherals that support various clock selections. This field can be written only when the clock is disabled (CGC = 0). Similarly, if the INUSE flag is set, this field is locked. 000b - Clock is off. 001b - Clock option 1 010b - Clock option 2 011b - Clock option 3 100b - Clock option 4 101b - Clock option 5 110b - Clock option 6 111b - Clock option 7
23-4 —	This read-only bit field is reserved and always has the value 0.
3 —	This read-only bit field is reserved and always has the value 0.
2-0 —	This read-only bit field is reserved and always has the value 0.

Chapter 26
System Clock Generator (SCG)

26.1 Chip-specific SCG information

Table 26-1. Reference links to related information

Topic	Related module	Reference
Full description	SCG	SCG
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

26.1.1 System Clock Generation

The System Clock Generation (SCG) is responsible for clock generation and distribution across this device. Functions performed by the SCG include: clock reference selection, generation of clock used to derive processor, system, peripheral bus and external memory interface clocks, source selection for peripheral clocks, and control of power saving clock gating mode.

Table 26-2. SCG configuration

Parameter	Description
Name	SCG
Instances	2
Configurable features	• SCG for dual cores, dual power domains • SCG0 in CM4 domain and SCG1 in CA7 domain
Interface speed	NA
External I/O pins	NA

26.1.2 DDR clock divider change procedure

When changing the DDR clock divider (including changes from 0 to another value), the following procedure must be followed.

1. Gate off MMDC clock (clear PCC_MMDC[CGC] in PCC3).
2. If NIC is using DDR as clock source (SCG1_NICCCR[NICCS] = 0x1), change NIC clock to a source other than DDR clock (for example, FIRC).
3. Disable SCG1 APLL (DDR clock source) and gate off SCG1 APLL PFDs (via APLLPFD register) in case a PFD output is selected (APLLCSR[PLLS] = 0x1). If CA7 core clock is using SCG1 APLL as its clock source (SCG1 CSR[SCS] = 0x5), this must be changed likewise before disabling SCG1 APLL.
4. Change DDR clock divider (via SCG1_DDRCCR[DDRDIV] register field).
5. Re-enable SCG1 APLL; wait for it to be valid (SCG1_APLLCSR[APLLVLD] = 0x1), and ungate APLL PFDs (if applicable).
6. Change NIC clock source back to DDR clock (SCG1_NICCR[NICCS] = 0x1) (if applicable).
7. Ungate MMDC clock (set PCC_MMDC[CGC] in PCC3).

26.1.3 SOSC operation in VLLS mode

SOSC remains active in M4 VLLS mode, should all SOSCCSR[SOSCEN], SOSCCSR[SOSCSTEN], and SOSCCSR[SOSCLPEN] bits are set upon entering such mode. Therefore, clearing any of these bits before an M4 VLLS entry prevents SOSC from remaining active under such power mode.

26.1.4 Instantiation information for SCG0 (M4) and SCG1 (A7)

There are two instances of SCG available on this device: SCG0 (M4 domain) and SCG1 (A7 domain). Following are some of the differences in both the instances.

For SCG0:

- The reset value of SCG_SPLLCFG register is 0x0003_0000 and width of SCG_SPLLCFG[MULT] is 3 (bits 16-18).
- PLL Output Frequency = Divided Reference Frequency * MULT

- SCG0 SPLL can generate a low jitter output clock of 480 MHz from a 16, 19.2, 24, 30, 32 MHz reference clock. It is an integer-N synthesizer. The frequency multiplier factor should be 30, 25, 20, 16, 15.
- The reset value of SCG_APLLLOCK_CNFG register is 16'h468 and the reset value of SCG_SPLLLOCK_CNFG is 16'h168. However, just the SCG_SPLLLOCK_CNFG register is overwritten to 16’h12C0 during the boot flow. These registers should be reprogramed when reconfiguring the PLLs with the result of the following equation:

  SCG_xPLLLOCK_CNFG = 150 µs / (1/PLL Ref Clock).

For SCG1:

- The reset value of SCG1 SPLLCFG register is 0x0116_0000. The SPLLCFG[MULT] is 7 bits [22:16].
- PLL Output Frequency = Divided Reference Frequency * (MULT + NUM/DENOM)
- SCG1 SPLL can generate a low jitter output clock of 528 MHz from a 24 MHz reference clock. It also generated 528, 518.4, 520, 510, and 512 MHz from a 16, 19.2, 26, 30, 32 MHz reference clock respectively. It is an Integer-N loop.
- The reset value of SCG_APLLLOCK_CNFG register is 16'h468 and the reset value of SCG_SPLLLOCK_CNFG is 16'h468. However, both the registers are overwritten to 16’h12C0 during the boot flow. These registers should be reprogrammed when reconfiguring the PLLs with the result of the following equation:

  SCG_xPLLLOCK_CNFG = 150 µs / (1/PLL Ref Clock).

  .
- SCG1_NICCSR[NIC1_DIVEXT] exists, but it is not used by the device as an internal clock. However, SCG1_NICCSR[NIC1_DIVEXT] is used as a Flex Bus clock through CLKOUT pin, controlled by SCG1_CLKOUTCNFG. Refer Figure 26-2 and Figure 26-3 for details.
- SIRCSSR[LK] bitfield is read-only and can't be written on SCG1.
- SPLLCFG [24] bit changes its reset value to 0 as soon as the protocol clock is available to the module.
- Bit 28 of PARAM[DIVPRES] is reserved with reset 1, and should be ignored.
- FIRCCSR[FIRCTREN] bitfields are read-only and can't be written for this instance (SCG1).

26.2 Introduction

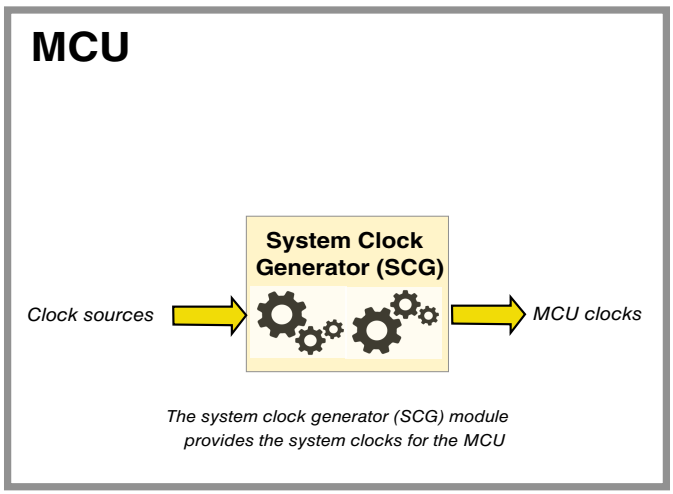


Figure 26-1. System Clock Generator (SCG) in MCU

The system clock generator (SCG) module provides the system clocks for the MCU. The SCG contains:

- an auxiliary phase-locked loop (APLL) with PLL Fractional Divide (PFD) outputs
- a system phase-locked loop (SPLL) with PLL Fractional Divide (PFD) outputs
- a slow internal reference clock (SIRC)
- a fast internal reference clock (FIRC)
- a system oscillator clock (SOSC)

The PLLs are sourced by the SOSC reference clock.

The SCG can select a source for the MCU system clocks, from one of the following: :

- the output clock of the Auxiliary PLL (APLL)
- the output clock of the System PLL (SPLL)
- a SCG reference clock (SIRC, FIRC, and SOSC)

To produce the external reference clock (which is also available as a clock source for the MCU systems clocks), the SCG can also work with crystal oscillators, using either an external crystal, a ceramic resonator, or some other external clock source.

26.2.1 Features

Key features of the SCG module are:

- Auxiliary Phase-locked loop (APLL):
  - Voltage-controlled oscillator (VCO)
  - Selectable Internal or External reference clock is used as the APLL source

    - Modulo VCO frequency divider
    - Phase/Frequency detector
    - Integrated loop filter
    - Can be selected as the clock source for the MCU system clocks
    - 3 programmable post-dividers clock outputs, which can be used as clock sources for other on-chip peripherals
    - 4 programmable fractional post-dividers clock outputs, which can be used as clock sources for system or other on-chip peripherals
- System Phase-locked loop (SPLL):
    - Voltage-controlled oscillator (VCO)
    - Selectable Internal or External reference clock is used as the PLL source
    - Modulo VCO frequency divider
    - Phase/Frequency detector
    - Integrated loop filter
    - Can be selected as the clock source for the MCU system clocks
    - 3 programmable post-dividers clock outputs, which can be used as clock sources for other on-chip peripherals
    - 4 programmable fractional post-dividers clock outputs, which can be used as clock sources for system or other on-chip peripherals
- 2 Internal reference clock (IRC) generators:
    - Slow IRC clock with programmable High and Low frequency range, with each range having a set of 8 trim bits for accuracy
    - Fast IRC clock with programmable High and Low frequency range, with 3 sets of trim bits for accuracy
    - Either the slow or the fast clock can be selected as the clock source for the MCU system clocks
    - 3 programmable post-divider clock outputs for each IRC, which can be used as clock sources for other on-chip peripherals
- System Crystal Oscillator:
    - Can be used as a source for the Auxiliary PLL

    - Can be used as a source for the System PLL
    - Can be selected as the clock source for the MCU system clocks
- External clock from the Real Time Counter (RTC):
    - Can be selected as the clock source for the MCU
- Clock monitor with reset and interrupt request capability for SOSC, ROSC clocks
- Each of the clock sources have reference dividers for clocking on-chip modules and peripherals, namely:
    - APLLDIV1_CLK / APLLDIV2_CLK / APLLDIV3_CLK
    - SPLLDIV1_CLK / SPLLDIV2_CLK / SPLLDIV3_CLK
    - FIRCDIV1_CLK / SCG_FIRCDIV2_CLK / SCG_FIRCDIV3_CLK
    - SIRCDIV1_CLK / SIRCDIV2_CLK / SIRCDIV3_CLK
    - SOSCDIV1_CLK / SOSCDIV2_CLK / SOSCDIV3_CLK

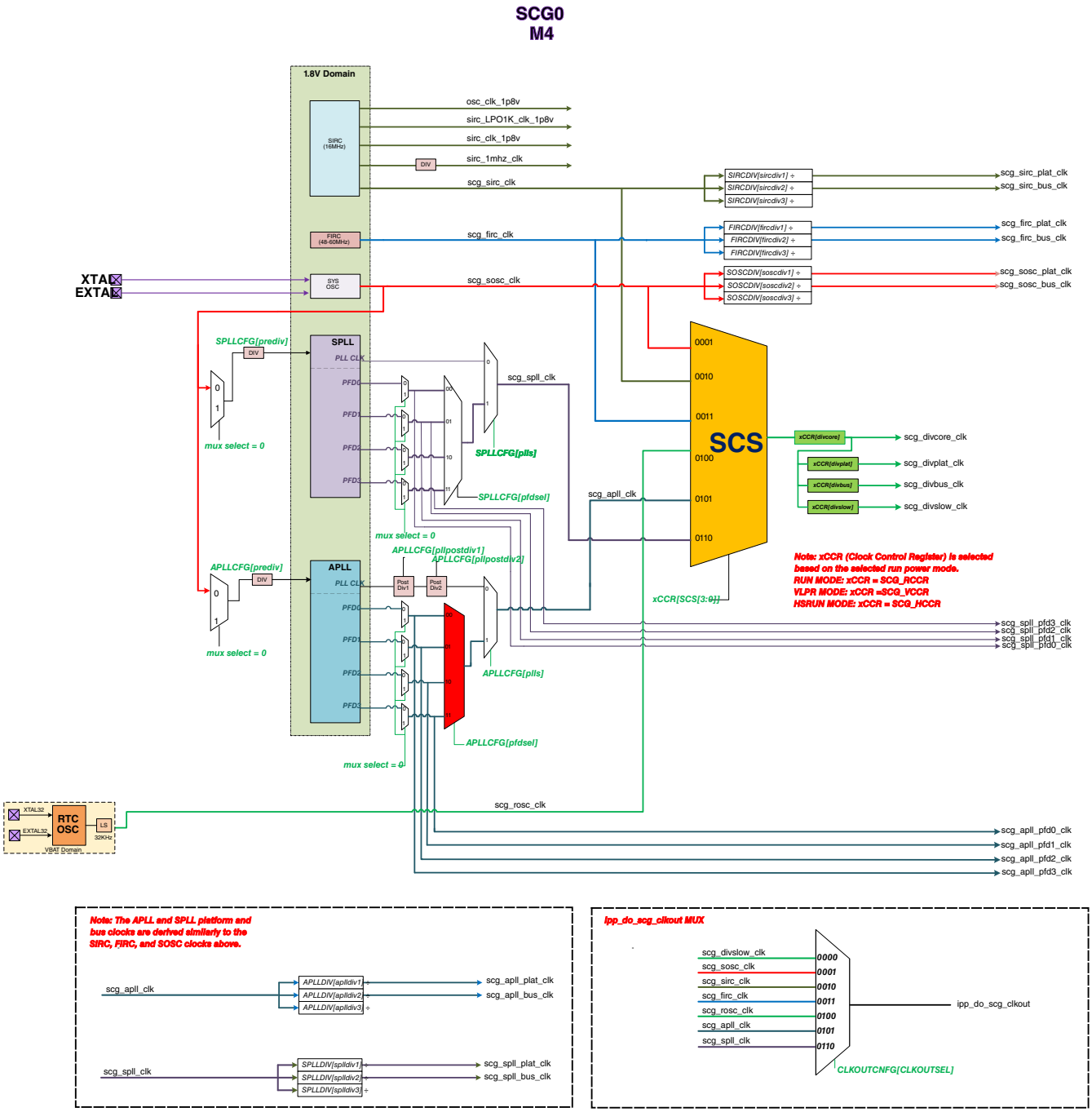


Figure 26-2. System Clock Generator block diagram for SCG0, M4 Domain

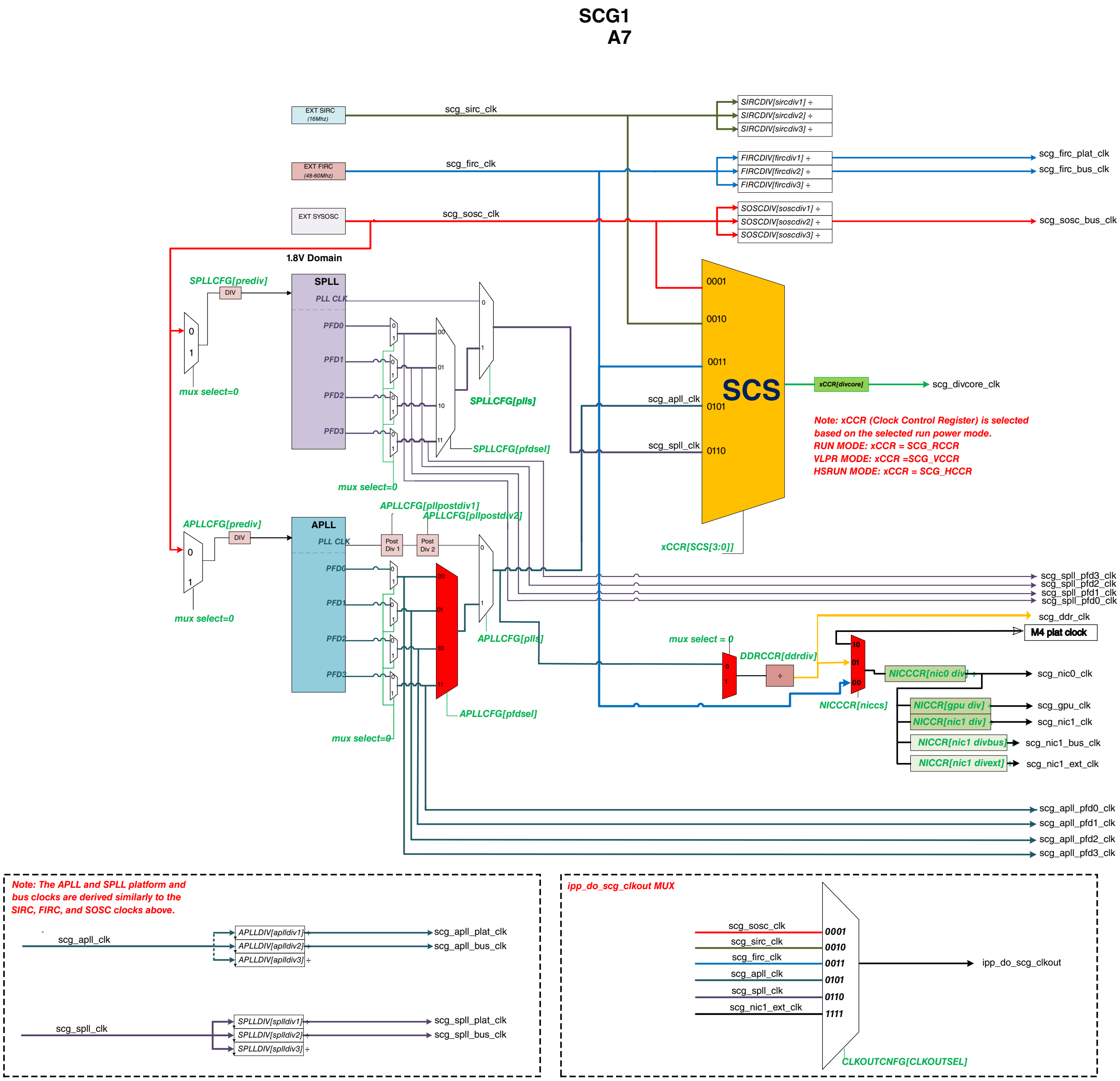


Figure 26-3. System Clock Generator block diagram for SCG1, A7 Domain

NOTE

To identify the oscillators used in your specific MCU device, see the configuration or clocking chapters.

26.3 Functional description

26.3.1 SCG Clock Mode Transitions

The following figure shows the valid clock mode transitions supported by SCG.

Slow IRC (SIRC) boot mode is not supported on this device.

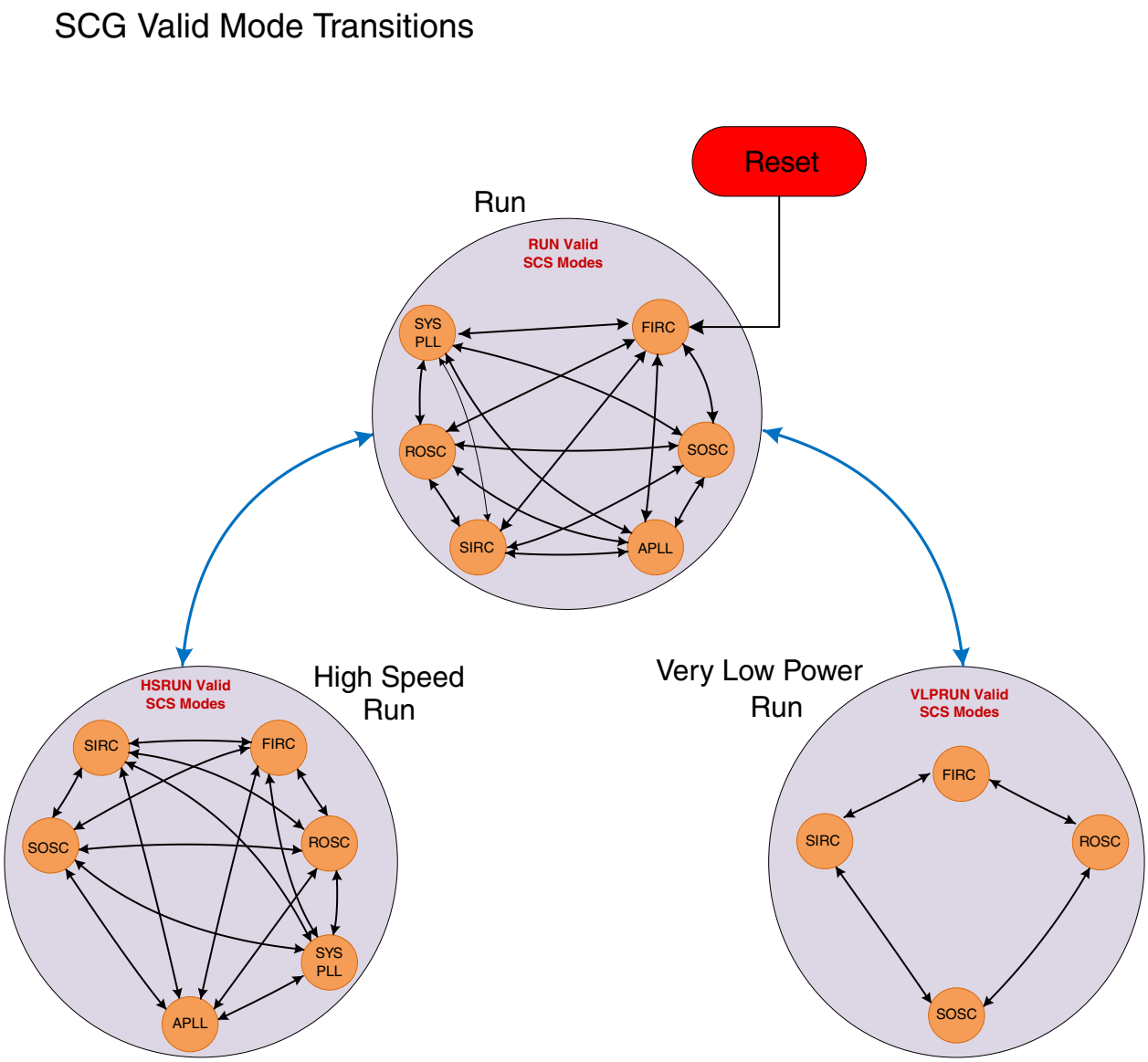


Figure 26-4. SCG Valid Mode Transitions for SCG0 (M4 domain)

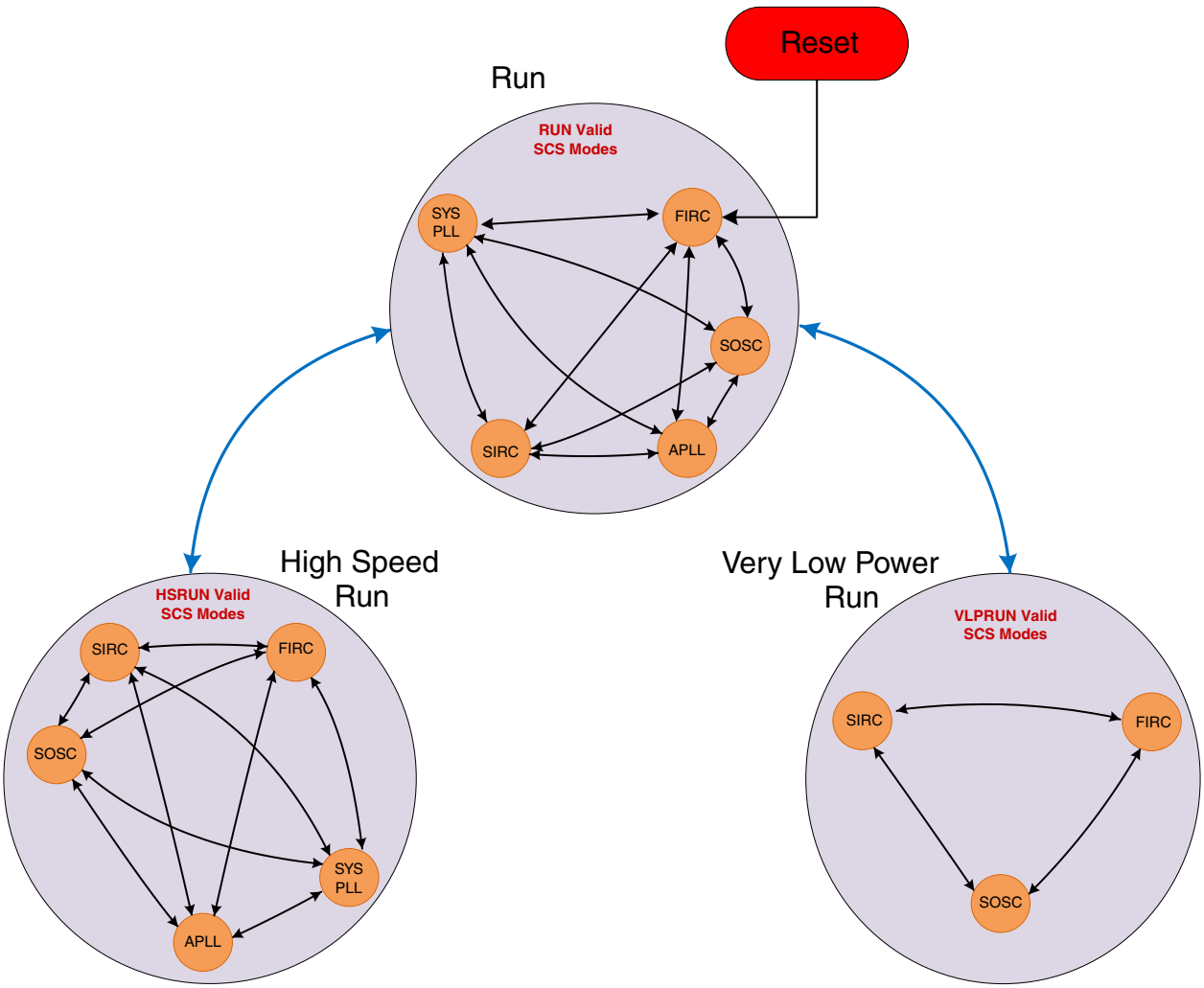


Figure 26-5. SCG Valid Mode Transitions for SCG1 (A7 domain)

The SCG will restrict programming into invalid clock modes and writes to SCS bits will be ignored. When a transition between run modes or a transition into wait mode occurs, the SCG completes the switch to the clock mode as defined in the SCG clock control register. When completed, the system switches to the selected run/wait mode.

The modes of operation listed in the following table are the valid modes for this implementation of the SCG.

Table 26-3. SCG modes of operation

Mode	Description

System Oscillator Clock (SOSC)	System Oscillator Clock (SOSC) mode is entered when all the following conditions occur:
	• RUN MODE:
	  • 0001 is written to RCCR[SCS]
	VLRUN MODE: One of the following scenarios is true
	  • 0001 is written to VCCR[SCS] and VCCR[SCS] matches RCCR[SCS]
	  • 0001 is written to VCCR[SCS], VCCR[SCS] differs from RCCR[SCS]) and 1 is written to SOSCCSR[SOSCLPEN]
	HSRUN MODE:
	  • 0001 is written to HCCR[SCS]
	• SOSCEN = 1
	• SOSCVLD = 1
	
	In SOSC mode, SCSCLKOUT and system clocks are derived from the external System Oscillator Clock (SOSC).

Slow Internal Reference Clock (SIRC)	Slow Internal Reference Clock (SIRC) mode is entered when all the following conditions occur:
	• RUN MODE:
	  • 0010 is written to RCCR[SCS]
	VLRUN MODE: One of the following scenarios is true
	  • 0010 is written to VCCR[SCS] and VCCR[SCS] matches RCCR[SCS]
	  • 0010 is written to VCCR[SCS], VCCR[SCS] differs from RCCR[SCS]) and 1 is written to SIRCCSR[SIRCLPEN]
	HSRUN MODE:
	  • 0010 is written to HCCR[SCS]
	• SIRCEN = 1
	• SIRCVLD = 1
	
	In SIRC mode, SCSCLKOUT and system clocks are derived from the slow internal reference clock. Two frequency ranges are available for SIRC clock as described in the SIRCCFG[RANGE] register definition. Changes to SIRC range settings will be ignored when SIRC clock is enabled.

Fast Internal Reference Clock (FIRC)	Fast Internal Reference Clock (FIRC) mode is the default clock mode of operation and is entered when all the following conditions occur:
	• RUN MODE:
	  • 0011 is written to RCCR[SCS]
	VLRUN MODE: One of the following scenarios is true
	  • 0011 is written to VCCR[SCS] and VCCR[SCS] matches RCCR[SCS]
	  • 0011 is written to VCCR[SCS], VCCR[SCS] differs from RCCR[SCS]) and 1 is written to FIRCCSR[FIRCLPEN]
	HSRUN MODE:
	  • 0011 is written to HCCR[SCS]
	• FIRCEN = 1
	• FIRCVLD = 1
	
	In FIRC mode, SCSCLKOUT and system clocks are derived from the fast internal reference clock. frequency range settings are available for FIRC clock as described in the FIRC[RANGE] register definition. Changes to FIRC range settings will be ignored when FIRC clock is enabled.

Real Time Clock Oscillator (RTCOSC)	Real Time Clock Oscillator (RTCOSC) mode is entered when all the following conditions occur:

Table continues on the next page...

Table 26-3. SCG modes of operation (continued)

Mode	Description

	• RUN MODE:
	  • 0100 is written to RCCR[SCS]
	VLRUN MODE:
	  • 0100 is written to VCCR[SCS]
	HSRUN MODE:
	  • 0100 is written to HCCR[SCS]
	• ROSCVLD = 1
	
	In RTCOSC mode, SCSCLKOUT and systems clocks are derived from the external RTC Oscillator Clock (RTCOSC).

Auxiliary PLL (APLL)	Auxiliary PLL (APLL) mode is entered when all the following conditions occur:
	• RUN MODE:
	  • 0101 is written to RCCR[SCS]
	VLRUN MODE:
	  • Invalid mode. Programming SCG into APLL mode will be ingored
	HSRUN MODE:
	  • 0101 is written to HCCR[SCS]
	• APLLEN = 1
	• APLLVLD = 1
	
	In APLL mode, SCSCLKOUT and system clocks are derived from the Auxiliary PLL (APLL) that is controlled by the System Oscillator (SOSC) clock. The APLL clock frequency locks to a multiplication factor, as specified by its corresponding SCG_APLLCFG[MULT], times the selected APLL reference frequency. The APLL programmable reference divider must be configured to produce a valid APLL reference clock. This divide value is defined by the SCG_APLLCFG[PREDIV] bits.

Sys PLL (SPLL)	Sys PLL (SPLL) mode is entered when all the following conditions occur:
	• RUN MODE:
	  • 0110 is written to RCCR[SCS]
	VLRUN MODE:
	  • Invalid mode. Programming SCG into SPLL mode will be ingored
	HSRUN MODE:
	  • 0110 is written to HCCR[SCS]
	• SPLLEN = 1
	• SPLLVLD = 1
	
	In SPLL mode, the SCSCLKOUT and system clocks are derived from the output of PLL which is controlled by the System Oscillator (SOSC) clock. The selected PLL clock frequency locks to a multiplication factor, as specified by its corresponding SCG_SPLLCFG[MULT], times the selected PLL reference frequency. The PLL's programmable reference divider must be configured to produce a valid PLL reference clock. This divide value is defined by the SCG_SPLLCFG[PREDIV] bits.

Stop	Entered whenever the MCU enters a Stop state. The power modes are chip-specific. For power mode assignments, see the chapter that describes how modules are configured and SCG behaviour during Stop recovery. Entering Stop mode, all SCG clock signals are static, including SCG system clocks (DIVCORE,DIVPLAT,,DIVBUS,DIVSLOW). There are exceptions; the following clocks can continue to run and stay enabled in the following cases:

Table 26-3. SCG modes of operation

Mode	Description
	SIRC is available in Normal Stop and VLPS modes when all the following conditions become true: • SIRCCSR[SIRCEN] = 1 • SIRCCSR[SIRCSTEN] = 1 • SIRCCSR[SIRCLPEN] = 1 in VLPS FIRC is available in Normal Stop and VLPS modes when all the following conditions become true: • FIRCCSR[FIRCEN] = 1 • FIRCCSR[FIRCSTEN] = 1 • FIRCCSR[FIRCLPEN] = 1 in VLPS SOSC is available in following low power stop modes (Normal Stop, VLPS, LLS) when all the below conditions are true. In VLLS stop mode, SOSCLK is disabled. • SOSCCSR[SOSCEN] = 1 • SOSCCSR[SOSCSTEN] = 1 • SOSCCSR[SOSCLPEN] = 1 (required only for Low Power Stop modes (VLPS and LLS) APLL is available in Normal Stop mode when all the following conditions become true: • APLLCSR[APLLEN] = 1 • APLLCSR[APLLSTEN] = 1 • APLLSTEN control bit has no effect in LLS or VLPS Power mode. SPLL is available in Normal Stop mode when all the following conditions are true: • SPLLCSR[SPLLEN] = 1 • SPLLCSR[SPLLSTEN] = 1 • SPLLSTEN control bit has no affect in LLS or VLPS Power mode.

26.4 Memory Map/Register Definition

This section includes the memory map and register definition.

The SCG registers can only be written when in supervisor mode. Write accesses when in user mode will result in a bus transfer error. Read accesses may be performed in both supervisor and user mode.

26.4.1 SCG register descriptions

- For any writeable SCG registers, only 32-bit writes are allowed. 8-bit or 16-bit writes will result in transfer errors.
- For PLL programming, enable the PLL first, and then the PFD can be programmed.

26.4.1.1 SCG Memory map

SCG1 base address: 403E_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_0000h
4h	Parameter Register (PARAM)	32	RO	8000_006Eh
10h	Clock Status Register (CSR)	32	RO	See description.
14h	Run Clock Control Register (RCCR)	32	RW	See description.
18h	VLPR Clock Control Register (VCCR)	32	RW	See description.
1Ch	HSRUN Clock Control Register (HCCR)	32	RW	0300_0000h
20h	SCG CLKOUT Configuration Register (CLKOUTCNFG)	32	RW	0300_0000h
30h	DDR Clock Control Register (DDRCCR)	32	RW	0000_0000h
40h	NIC Clock Control Register (NICCCR)	32	RW	0000_0000h
44h	NIC Clock Status Register (NICCSR)	32	RO	0000_0000h
100h	System OSC Control Status Register (SOSCCSR)	32	RW	See description.
104h	System OSC Divide Register (SOSCDIV)	32	RW	0000_0000h
200h	Slow IRC Control Status Register (SIRCCSR)	32	RW	0100_0000h
204h	Slow IRC Divide Register (SIRCDIV)	32	RW	0000_0000h
300h	Fast IRC Control Status Register (FIRCCSR)	32	RW	0300_2000h
304h	Fast IRC Divide Register (FIRCDIV)	32	RW	0000_0000h
500h	Auxiliary PLL Control Status Register (APLLCSR)	32	RW	0000_0000h
504h	Auxiliary PLL Divide Register (APLLDIV)	32	RW	0000_0000h
508h	Auxiliary PLL Configuration Register (APLLCFG)	32	RW	0116_0000h
50Ch	Auxiliary PLL PFD Register (APLLPFD)	32	RW	8080_8080h
510h	Auxiliary PLL Numerator Register (APLLNUM)	32	RW	04DD_2F15h
514h	Auxiliary PLL Denominator Register (APLLDENOM)	32	RW	1FFF_FFDBh
518h	Auxiliary PLL Spread Spectrum Register (APLLSS)	32	RW	0000_0000h
5F8h	Auxiliary PLL LOCK Configuration Register (APLLLOCK_CNFG)	32	RW	0000_0468h
600h	System PLL Control Status Register (SPLLCSR)	32	RW	0000_0000h
604h	System PLL Divide Register (SPLLDIV)	32	RW	0000_0000h
608h	System PLL Configuration Register (SPLLCFG)	32	RW	0116_0000h
60Ch	System PLL PFD Register (SPLLPFD)	32	RW	8080_8080h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
610h	System PLL Numerator Register (SPLLNUM)	32	RW	04DD_2F15h
614h	System PLL Denominator Register (SPLLDENOM)	32	RW	1FFF_FFDBh
618h	System PLL Spread Spectrum Register (SPLLSS)	32	RW	0000_0000h
6F8h	System PLL LOCK Configuration Register (SPLLLOCK_CNFG)	32	RW	0000_0468h

26.4.1.2 Version ID Register (VERID)

26.4.1.2.1 Offset

Register	Offset
VERID	0h

26.4.1.2.2 Function

Note: Writing to this register will result in a transfer error.

26.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VERSION															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VERSION															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.1.2.4 Fields

Field	Function
31-0 VERSION	SCG Version Number

26.4.1.3 Parameter Register (PARAM)

26.4.1.3.1 Offset

Register	Offset
PARAM	4h

26.4.1.3.2 Function

Note: Writing to this register will result in a transfer error.

26.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DIVPRES					0										
W																
Reset	1[1]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CLKPRES							
W																
Reset	0	0	0	0	0	0	0	0	0[2]	1	1	0	1	1	1	0

1. The reset value is controlled by which SCG System Dividers are used by the SoC.
2. The reset value is controlled by which SCG Clock Sources are used by the SoC. Please reference the Reference manual clocking chapter.

26.4.1.3.4 Fields

Field	Function
31-27 DIVPRES	Divider Present Indicates which system clock dividers are present in this instance of SCG. 1xxxxb - System DIVCORE is present.
26-16 —	Reserved
15-8 —	Reserved
7-0 CLKPRES	Clock Present Indicates which clock sources are present in this instance of SCG. Any bits not defined in this bit field are Reserved and always has the value 0 when read.

Field	Function
	00000000-00000001b - Reserved. x1xxxxxxb - System PLL (SPLL) is present. xx1xxxxxb - Auxiliary PLL (APLL) is present. xxxx1xxxb - Fast IRC (FIRC) is present. xxxxx1xxb - Slow IRC (SIRC) is present. xxxxxx1xb - System OSC (SOSC) is present.

26.4.1.4 Clock Status Register (CSR)

26.4.1.4.1 Offset

Register	Offset
CSR	10h

26.4.1.4.2 Function

The Clock Status Register (CSR) returns the currently configured system clock source and the system clock dividers for the core (DIVCORE). The SCG_CSR reports the configuration set by one of the clock control registers SCG_RCCR, SCG_VCCR, SCG_HCCR.

Note: Writing to the Clock Status Register (CSR) will result in a transfer error.

26.4.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	u[1]	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-1 or div-by-2 when resetting into RUN mode or div-by-4 or div-by-8 when resetting into VLPR mode

26.4.1.4.4 Fields

Field	Function
31-28 —	Reserved
27-24 SCS	System Clock Source Returns the currently configured clock source generating the system clock. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - Reserved 0101b - Auxiliary PLL (APLL_CLK) 0110b - System PLL (SPLL_CLK) 0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

26.4.1.5 Run Clock Control Register (RCCR)

26.4.1.5.1 Offset

Register	Offset
RCCR	14h

26.4.1.5.2 Function

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in Run mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in RUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in RUN, new system clock divide ratios will not take affect until a new clock source is valid.

NOTE

Switching to new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

26.4.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					SCS			0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	u[1]	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-1 and div-by-2

26.4.1.5.4 Fields

Field	Function
31-27 —	Reserved
26-24	System Clock Source

Table continues on the next page...

Field	Function
SCS	Selects the clock source generating the system clock in Run mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in Run mode requires that clock source to be enabled first and be valid before system clocks are allowed to switch to that clock source. **NOTE:** Programming SCS to a different value should only be done after the previous SCS clock switch has finished. 000b - Reserved 001b - System OSC (SOSC_CLK) 010b - Slow IRC (SIRC_CLK) 011b - Fast IRC (FIRC_CLK) 100b - Reserved 101b - Auxiliary PLL (APLL_CLK) 110b - System PLL (SPLL_CLK) 111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

26.4.1.6 VLPR Clock Control Register (VCCR)

26.4.1.6.1 Offset

Register	Offset
VCCR	18h

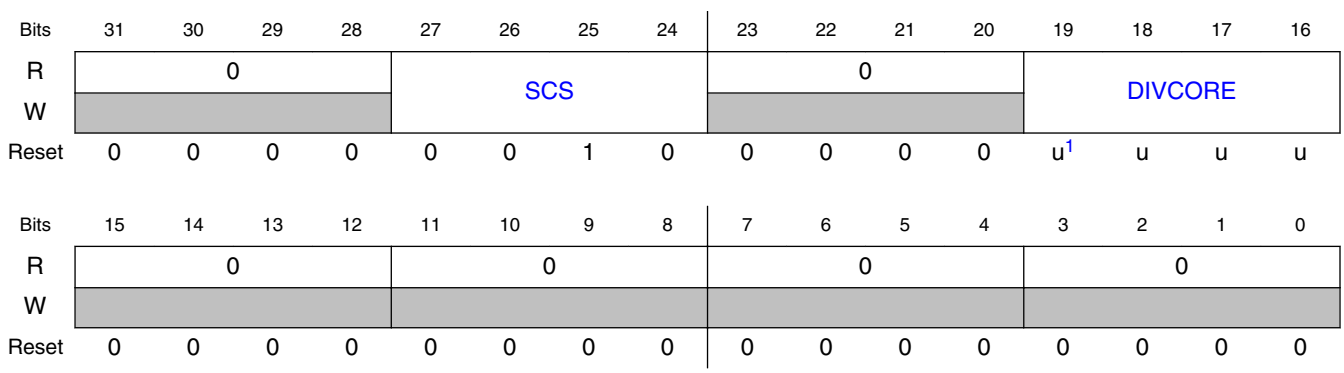
26.4.1.6.2 Function

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in VLPR mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in VLPR requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in VLPR, new system clock divide ratios will not take affect until new clock source is valid.

NOTE

Switching to a new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

26.4.1.6.3 Diagram



1. The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-4 and div-by-8.

26.4.1.6.4 Fields

Field	Function
31-28 —	Reserved
27-24	System Clock Source

Table continues on the next page...

Field	Function
SCS	Selects the clock source generating the system clock in VLPR mode. Attempting to select a clock that is not valid will be ignored. Selects the clock source generating the system clock. Selecting a different clock source when in VLPR mode requires that clock source to be enabled first and be valid before system clocks switch to that clock source. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - Reserved 0101b - Reserved 0110b - Reserved 0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

26.4.1.7 HSRUN Clock Control Register (HCCR)

26.4.1.7.1 Offset

Register	Offset
HCCR	1Ch

26.4.1.7.2 Function

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in HSRUN mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in HSRUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in HSRUN, new system clock divide ratios will not take affect until new clock source is valid.

NOTE

Switching to a new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

26.4.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.1.7.4 Fields

Field	Function
31-28 —	Reserved
27-24 SCS	System Clock Source Selects the clock source generating the system clock in HSRUN mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in HSRUN mode will enable that clock source and switch to that clock mode when it is valid. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - Reserved 0101b - Auxiliary PLL (APLL_CLK)

Table continues on the next page...

Field	Function
	0110b - System PLL (SPLL_CLK) 0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

26.4.1.8 SCG CLKOUT Configuration Register (CLKOUTCNFG)

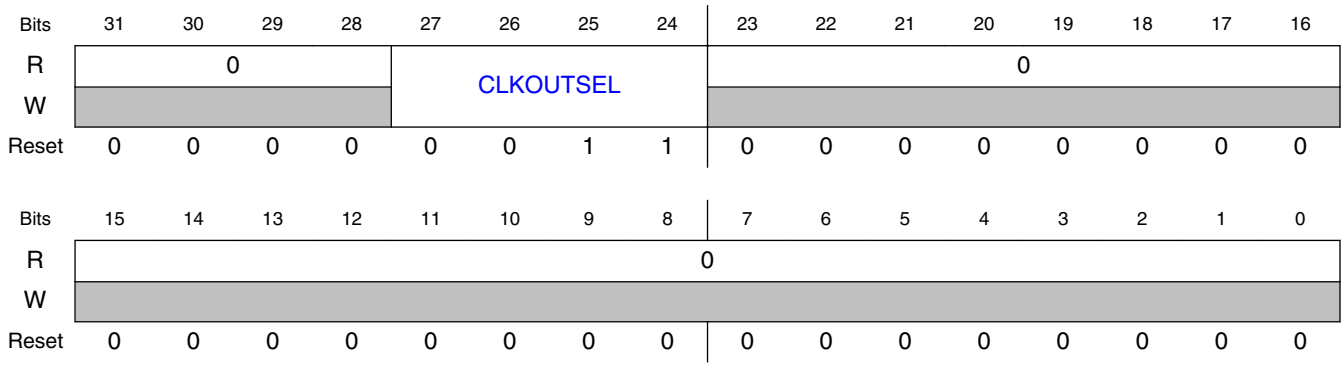
26.4.1.8.1 Offset

Register	Offset
CLKOUTCNFG	20h

26.4.1.8.2 Function

This register controls which SCG clock source is selected to be ported out to the CLKOUT pin.

26.4.1.8.3 Diagram



26.4.1.8.4 Fields

Field	Function
31-28 —	Reserved
27-24 CLKOUTSEL	SCG Clkout Select Selects the SCG system clock. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - Reserved 0101b - Auxiliary PLL (APLL_CLK) 0110b - System PLL (SPLL_CLK) 0111b - Reserved 1111b - NIC1 EXT CLK
23-0 —	Reserved

26.4.1.9 DDR Clock Control Register (DDRCCR)

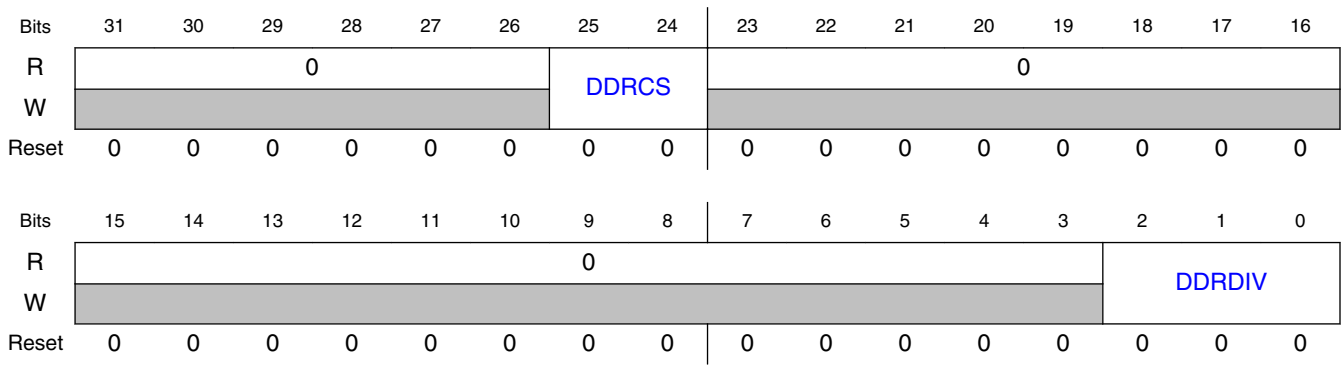
26.4.1.9.1 Offset

Register	Offset
DDRCCR	30h

26.4.1.9.2 Function

This register controls the DDR clock source and the DDR clock dividers. This register can only be written using a 32-bit write. Selecting a different clock source requires that clock source to be enabled first and be valid and previous clock source to remain enabled until switching to new clock is complete. If DDR clock divide ratios also change when selecting a different clock mode, new DDR clock divide ratios will not take affect until new clock source is valid.

26.4.1.9.3 Diagram



26.4.1.9.4 Fields

Field	Function
31-26 —	Reserved
25-24 DDRCS	DDR Clock Source Selects the clock source generating the DDR clock. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source requires that clock source to be enabled first and be valid and the old clock to stay on and enabled before DDR clock is allowed to switch to that clock source. 00b - APLL PLLS Clock 01b - RESERVED. Software should write 0 to this bit to maintain compatibility. 10b - RESERVED 11b - RESERVED
23-3 —	Reserved
2-0 DDRDIV	DDR Divider DDRDIV can only be updated when the DDR clock is disabled. If the DDRDIV is changed on the fly when the DDR clock is active, glitches can occur. Before enabling DDR, a non zero value must be written to DDRDIV. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8

Field	Function
	101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.1.10 NIC Clock Control Register (NICCCR)

26.4.1.10.1 Offset

Register	Offset
NICCCR	40h

26.4.1.10.2 Function

This register controls the NIC clock source and the NIC clock dividers. This register can only be written using a 32-bit write. Selecting a different clock source requires that clock source to be enabled first and be valid before NIC clocks are allowed to switch over to that clock source.

The NICCS bitfield cannot be changed at the same time as the NIC Divider bitfields.

26.4.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		NICCS		NIC0_DIV				GPU_DIV				NIC1_DIV			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				NIC1_DIVEXT				NIC1_DIVBUS				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.1.10.4 Fields

Field	Function
31-30	Reserved

Table continues on the next page...

Field	Function
—	
29-28 NICCS	NIC Clock Source Selects the clock source generating the NIC clocks. 00b - Fast IRC 01b - DDR Clock 10b - NIC External PLL 11b - RESERVED
27-24 NIC0_DIV	NIC0 Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
23-20 GPU_DIV	GPU Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
19-16 NIC1_DIV	NIC1 Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13

Table continues on the next page...

Field	Function
	1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 NIC1_DIVEXT	NIC1 External Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
7-4 NIC1_DIVBUS	NIC1 Bus Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
3-0 —	Reserved

26.4.1.11 NIC Clock Status Register (NICCSR)

26.4.1.11.1 Offset

Register	Offset
NICCSR	44h

26.4.1.11.2 Function

This register returns the currently configured NIC clock source and the NIC clock dividers. The SCG_NICCSR reflects the configuration set by the clock control registers SCG_NICCCR.

26.4.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		NICCS		NIC0_DIV				GPU_DIV				NIC1_DIV			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				NIC1_DIVEXT				NIC1_DIVBUS				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.1.11.4 Fields

Field	Function
31-30 —	Reserved
29-28 NICCS	NIC Clock Source Selects the clock source generating the NIC clocks. 00b - Fast IRC 01b - DDR Clock 10b - NIC External PLL 11b - Reserved
27-24 NIC0_DIV	NIC0 Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15

Table continues on the next page...

Field	Function
	1111b - Divide-by-16
23-20 GPU_DIV	GPU Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
19-16 NIC1_DIV	NIC1 Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 —	Reserved
11-8 NIC1_DIVEXT	NIC1 External Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
7-4	NIC1 Bus Clock Divide Ratio

Table continues on the next page...

Field	Function
NIC1_DIVBUS	0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
3-0 —	Reserved

26.4.1.12 System OSC Control Status Register (SOSCCSR)

26.4.1.12.1 Offset

Register	Offset
SOSCCSR	100h

26.4.1.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					SOSCERR	SOSCSEL	SOSCVLD	LK	0					SOSCCMRE	SOSCCM
W						W1C										
Reset	0	0	0	0	0	u[1]	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													0		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1. This flag is reset on Chip POR only

26.4.1.12.3 Fields

Field	Function
31-27 —	Reserved
26 SOSCERR	System OSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one. 0b - System OSC Clock Monitor is disabled or has not detected an error 1b - System OSC Clock Monitor is enabled and detected an error
25 SOSCSEL	System OSC Selected 0b - System OSC is not the system clock source 1b - System OSC is the system clock source
24 SOSCVLD	System OSC Valid 0b - System OSC is not enabled or clock is not valid 1b - System OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - This Control Status Register can be written. 1b - This Control Status Register cannot be written.
22-18 —	Reserved
17 SOSCCMRE	System OSC Clock Monitor Reset Enable 0b - Clock Monitor generates interrupt when error detected 1b - Clock Monitor generates reset when error detected
16 SOSCCM	System OSC Clock Monitor Enables the clock monitor when SOSCVLD is set. **NOTE:** SOSC clock monitor remains enabled in VLPR and VLPS if SOSCCM is enabled. The clock monitor is always disabled in LLS/VLLS modes. When the clock monitor is disabled in a low power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. **NOTE:** The reference clock used to monitor the SOSC is the SIRC. This clock must be programmed to be enabled in order to monitor the SOSC. SIRC is automatically disabled in LLS and VLLS power modes and clock monitor of SOSC will be disabled. 0b - System OSC Clock Monitor is disabled 1b - System OSC Clock Monitor is enabled
15-3 —	Reserved
2-1 —	Reserved
0 —	Reserved

26.4.1.13 System OSC Divide Register (SOSCDIV)

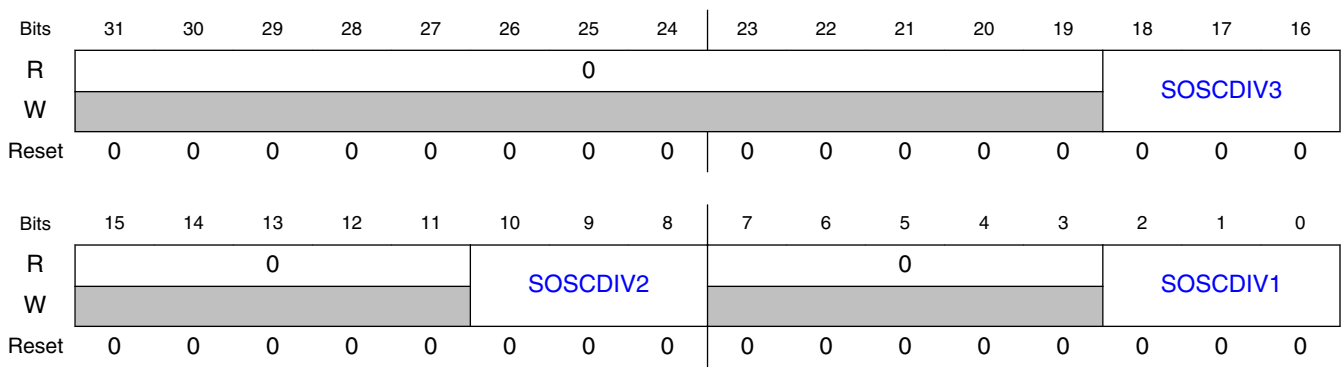
26.4.1.13.1 Offset

Register	Offset
SOSCDIV	104h

26.4.1.13.2 Function

The SCG_SOSCDIV register provides the control of 3 clock trees which can be used to provide optional peripheral functional clocks, or alternative module clocks. Each clock tree has optional dividers of the input SOSC clock. Changes to SOSCDIV should be done when System OSC is disabled to prevent glitches to output divided clock.

26.4.1.13.3 Diagram



26.4.1.13.4 Fields

Field	Function
31-19 —	Reserved
18-16 SOSCDIV3	System OSC Clock Divide 3 Clock divider 3 for System OSC. Used by slow clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

Table continues on the next page...

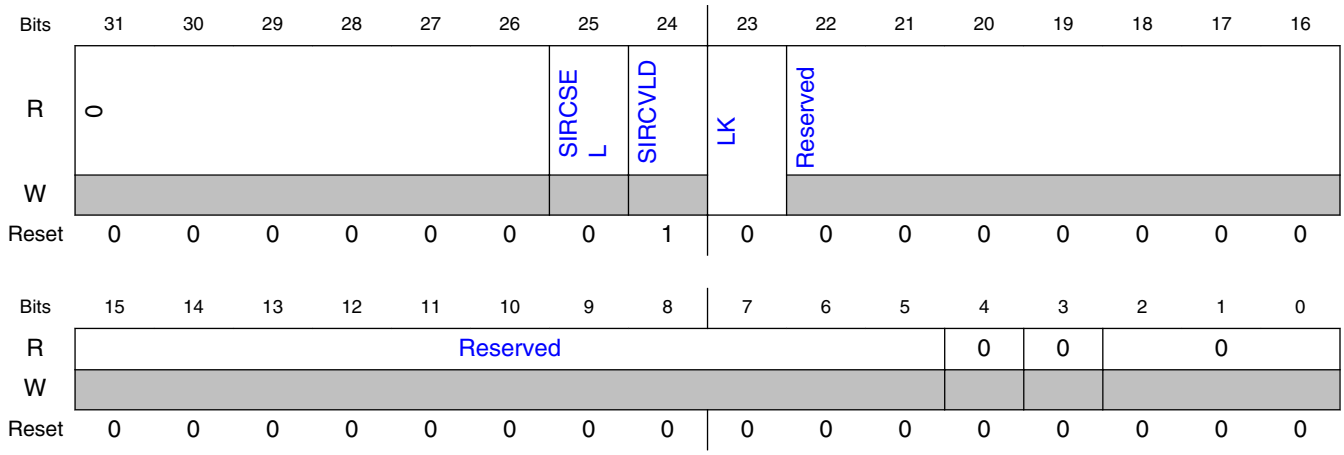
Field	Function
15-11 —	Reserved
10-8 SOSCDIV2	System OSC Clock Divide 2 Clock divider 2 for System OSC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 SOSCDIV1	System OSC Clock Divide 1 Clock divider 1 for System OSC. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.1.14 Slow IRC Control Status Register (SIRCCSR)

26.4.1.14.1 Offset

Register	Offset
SIRCCSR	200h

26.4.1.14.2 Diagram



26.4.1.14.3 Fields

Field	Function
31-26 —	Reserved
25 SIRCSEL	Slow IRC Selected 0b - Slow IRC is not the system clock source 1b - Slow IRC is the system clock source
24 SIRCVLD	Slow IRC Valid 0b - Slow IRC is not enabled or clock is not valid 1b - Slow IRC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. The purpose for this bitfield is to prevent runaway code from changing SIRC clock configurations. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-5 —	This field is reserved and is always has the value 0
4 —	Reserved
3 —	This field is reserved and is always has the value 0
2-0 —	Reserved

26.4.1.15 Slow IRC Divide Register (SIRCDIV)

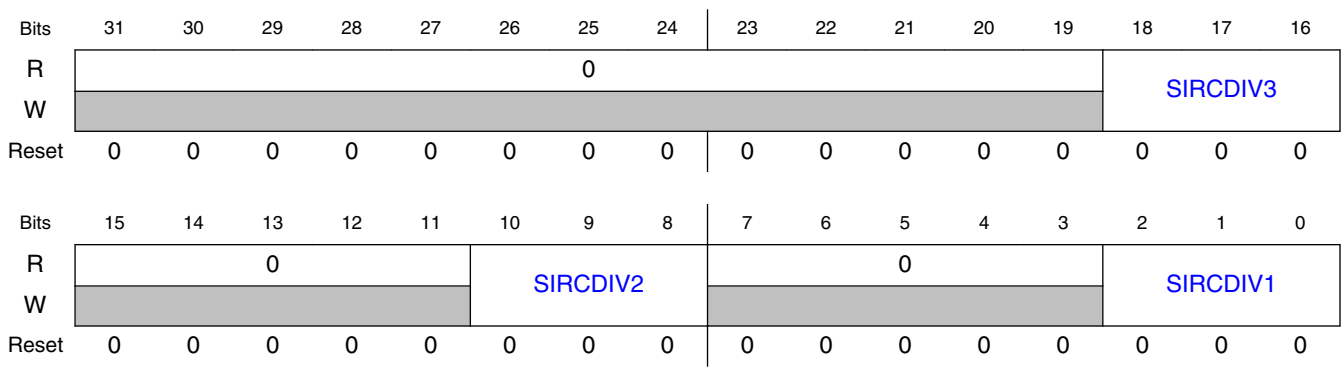
26.4.1.15.1 Offset

Register	Offset
SIRCDIV	204h

26.4.1.15.2 Function

To prevent glitches to the output divided clock, change SIRDIV when the Slow IRC is disabled.

26.4.1.15.3 Diagram



26.4.1.15.4 Fields

Field	Function
31-19 —	Reserved
18-16 SIRCDIV3	Slow IRC Clock Divider 3 Clock divider 3 for Slow IRC. Used by slow clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved

Table continues on the next page...

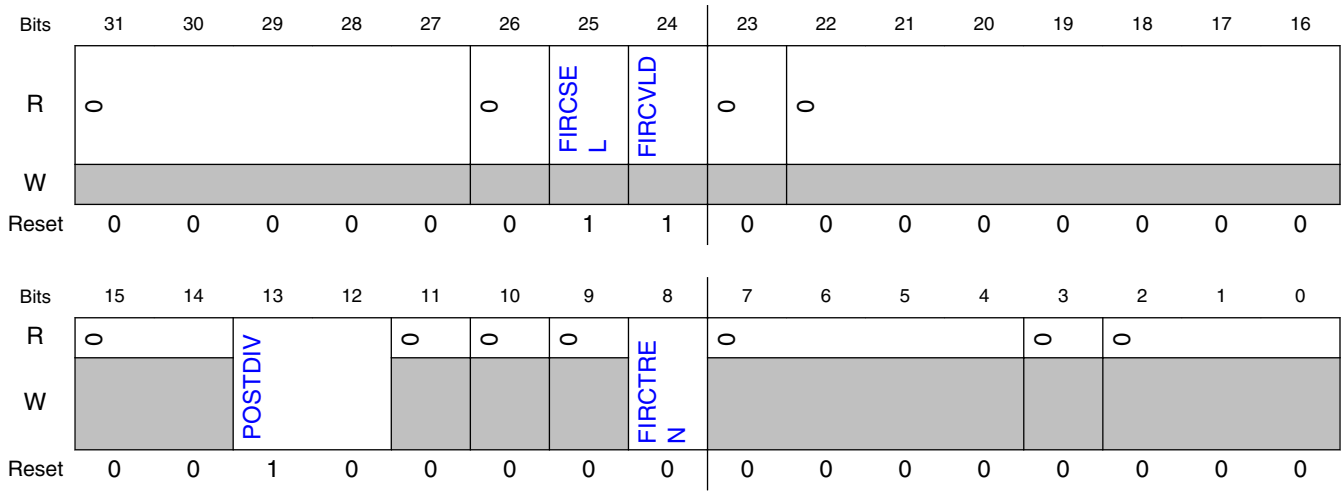
Field	Function
10-8 SIRCDIV2	Slow IRC Clock Divide 2 Clock divider 2 for Slow IRC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 SIRCDIV1	Slow IRC Clock Divide 1 Clock divider 1 for Slow IRC. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.1.16 Fast IRC Control Status Register (FIRCCSR)

26.4.1.16.1 Offset

Register	Offset
FIRCCSR	300h

26.4.1.16.2 Diagram



26.4.1.16.3 Fields

Field	Function
31-27 —	Reserved
26 —	Reserved
25 FIRCSEL	Fast IRC Selected status 0b - Fast IRC is not the system clock source 1b - Fast IRC is the system clock source
24 FIRCVLD	Fast IRC Valid status 0b - Fast IRC is not enabled or clock is not valid. 1b - Fast IRC is enabled and output clock is valid. The clock is valid after there is an output clock from the FIRC analog.
23 —	Reserved
22-14 —	Reserved
13-12 POSTDIV	Fast IRC Divider Divides the FIRC output clock selected by the RANGE bit. Divide values can be changed while FIRC is enabled 00b - Divide by 1 01b - Divide by 2 10b - Divide by 3 11b - Divide by 4
11 —	Reserved
10	Reserved

Table continues on the next page...

Field	Function
—	
9 —	Reserved
8 FIRCTREN	Fast IRC Trim Enable 0b - Disable trimming Fast IRC to an external clock source 1b - Enable trimming Fast IRC to an external clock source
7-4 —	Reserved
3 —	Reserved
2-0 —	Reserved

26.4.1.17 Fast IRC Divide Register (FIRCDIV)

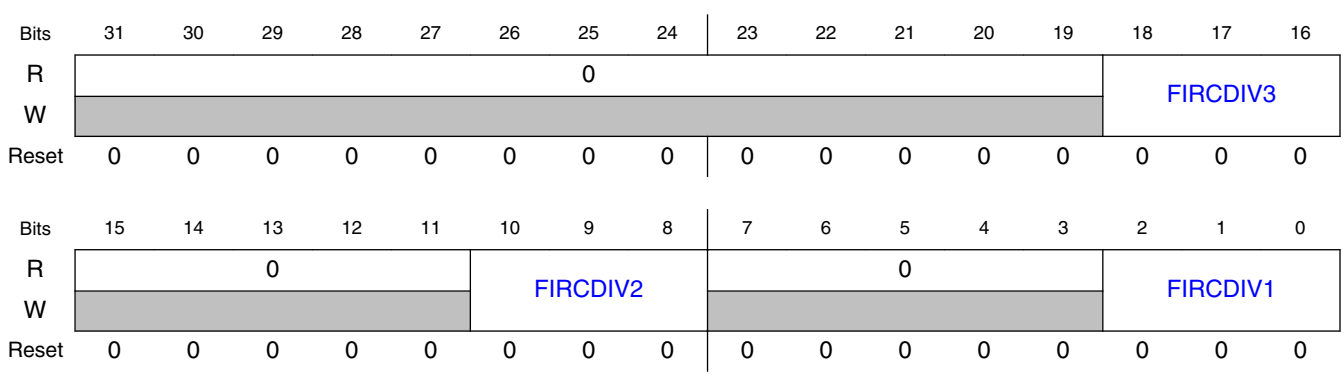
26.4.1.17.1 Offset

Register	Offset
FIRCDIV	304h

26.4.1.17.2 Function

Changes to FIRCDIV should be done when FAST IRC is disabled to prevent glitches to output divided clock.

26.4.1.17.3 Diagram



26.4.1.17.4 Fields

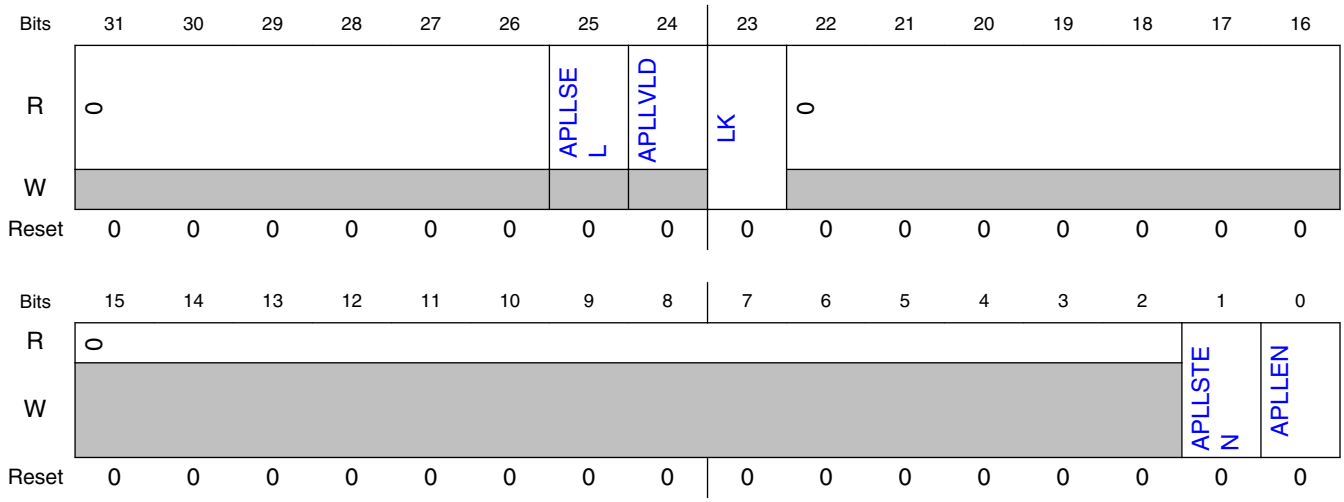
Field	Function
31-19 —	Reserved
18-16 FIRCDIV3	Fast IRC Clock Divider 3 Clock divider 3 for the Fast IRC. Used by slow clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved
10-8 FIRCDIV2	Fast IRC Clock Divide 2 Clock divider 2 for the Fast IRC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 FIRCDIV1	Fast IRC Clock Divide 1 Clock divider 1 for Fast IRC. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.1.18 Auxiliary PLL Control Status Register (APLLCSR)

26.4.1.18.1 Offset

Register	Offset
APLLCSR	500h

26.4.1.18.2 Diagram



26.4.1.18.3 Fields

Field	Function
31-26 —	Reserved
25 APLLSEL	APLL Selected 0b - APLL is not the system clock source 1b - APLL is the system clock source
24 APLLVLD	APLL Valid 0b - APLL is not enabled or clock is not valid 1b - APLL is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-2 —	Reserved
1 APLLSTEN	APLL Stop Enable 0b - APLL is disabled in Stop modes 1b - APLL is enabled in Stop modes
0 APLLEN	Auxiliary PLL (APLL) Enable **When configuring the APLL, APLLEN should be set/cleared first. Then the rest of the configuration process can be completed (disable PFDs, etc).** 0b - APLL is disabled 1b - APLL is enabled

26.4.1.19 Auxiliary PLL Divide Register (APLLDIV)

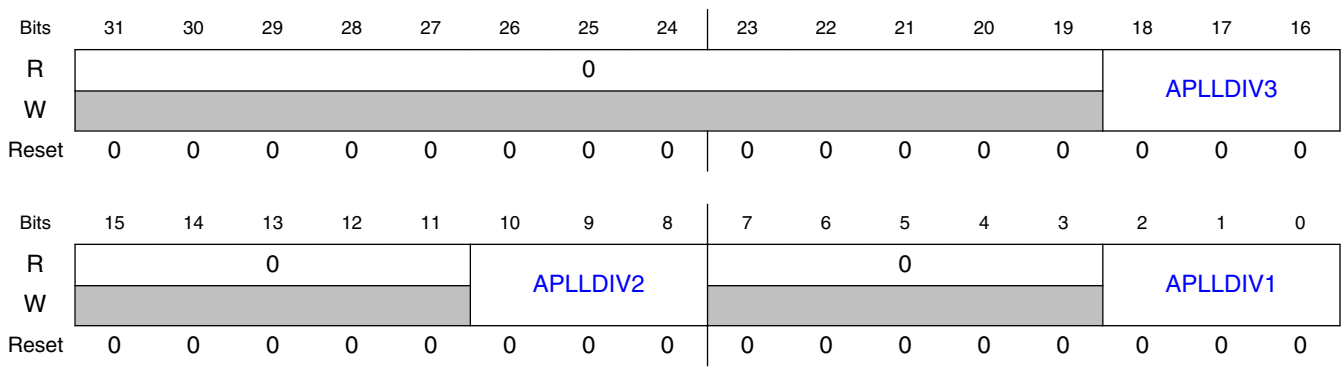
26.4.1.19.1 Offset

Register	Offset
APLLDIV	504h

26.4.1.19.2 Function

Changes to APLLDIV should be done when Auxiliary PLL is disabled to prevent glitches to output divided clock.

26.4.1.19.3 Diagram



26.4.1.19.4 Fields

Field	Function
31-19 —	Reserved
18-16 APLLDIV3	Auxiliary PLL Clock Divide 3 Clock divider 3 for Auxiliary PLL. Used by slow clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11	Reserved

Table continues on the next page...

Field	Function
—	
10-8 APLLDIV2	Auxiliary PLL Clock Divide 2 Clock divider 2 for Auxiliary PLL. Used by bus clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 APLLDIV1	Auxiliary PLL Clock Divide 1 Clock divider 1 for Auxiliary PLL. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.1.20 Auxiliary PLL Configuration Register (APLLCFG)

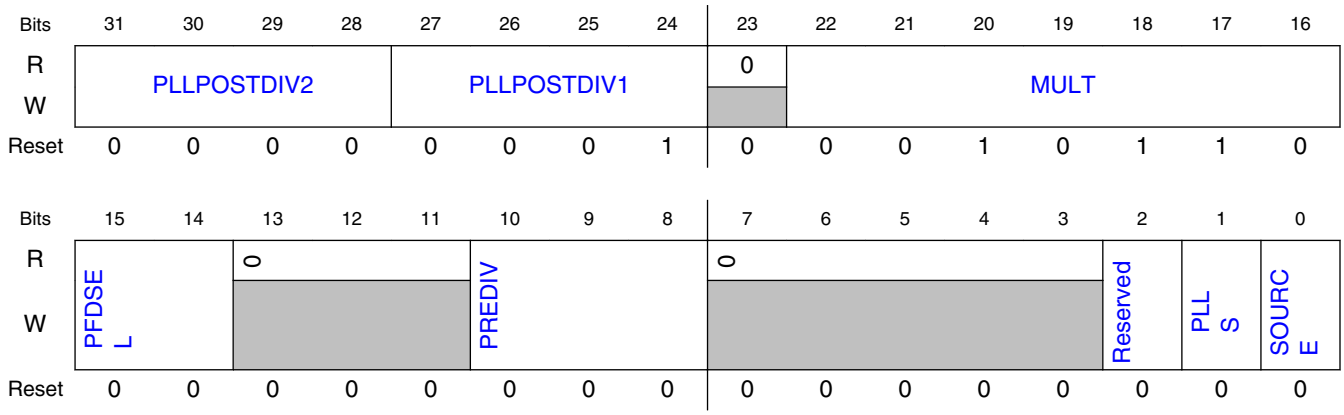
26.4.1.20.1 Offset

Register	Offset
APLLCFG	508h

26.4.1.20.2 Function

The APLLCFG register cannot be changed when the Auxiliary PLL is enabled. When the Auxiliary PLL is enabled, writes to this register are ignored, and there is no transfer error.

26.4.1.20.3 Diagram



26.4.1.20.4 Fields

Field	Function
31-28 PLLPOSTDIV2	Auxiliary PLL Post Clock Divide2 Ratio Auxiliary PLL 2nd stage post divider. Post clock dividers apply to APLL clock (non-PFD) only. NOTE: Correct divider ratio must be set prior to enabling of PLL. Changes to divider will be ignored if Auxiliary PLL is enabled. 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
27-24 PLLPOSTDIV1	Auxiliary PLL Post Clock Divide1 Ratio Auxiliary PLL 1st stage post divider. Post clock dividers apply to APLL clock (non-PFD) only. NOTE: Correct divider ratio must be set prior to enabling of PLL. Changes to divider will be ignored if Auxiliary PLL is enabled. 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9

Table continues on the next page...

Field	Function
	1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
23 —	Reserved
22-16 MULT	Auxiliary PLL Multiplier Multiplier for the Auxiliary PLL. The MULT bits establish the multiplication factor applied to the PLL reference clock frequency. • PLL Output Frequency = Divided Reference Frequency * MULT • Valid Div Input Reference frequency are (in MHz): 24 (default), 32, 30, 26, 19.2 and 16 • Valid MULT values are 33, 27, 22, 20, 17, 16
15-14 PFDSEL	PFD Clock Select Selects which PFD output clock will be used as a SCG system reference clock or as a SCG DDR reference clock. 00b - PFD0 output clock selected. 01b - PFD1 output clock selected. 10b - PFD2 output clock selected. 11b - PFD3 output clock selected.
13-11 —	Reserved
10-8 PREDIV	PLL Reference Clock Divider Selects the amount to divide down the reference clock for the Auxiliary PLL. The resulting frequency must be 24 MHz(default),32 MHz, 30 MHz, 26 MHz, 19.2 MHz and 16 MHz.
7-3 —	Reserved
2 —	Reserved This field is reserved. Software should write 0 to this bit to maintain compatibility.
1 PLLS	PLL Select

Table 26-4. Auxiliary PLL Reference Divide Factor

PREDIV	Divide Factor
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

Table continues on the next page...

Field	Function
	Selects the PFD or APLL as the PLL output clock used to generate system clocks. The PFD output clock used if PLLS=1 is determined by the PFDSEL bits. 0b - APLL clock selected. 1b - APLL PFD output clock selected
0 SOURCE	Clock Source Configures the input clock source for the Auxiliary PLL. Valid reference frequency must be 24 MHz, 32 MHz, 30 MHz,26 MHz, 19.2 MHz and 16 MHz. 0b - System OSC 1b - Reserved

26.4.1.21 Auxiliary PLL PFD Register (APLLPFD)

26.4.1.21.1 Offset

Register	Offset
APLLPFD	50Ch

26.4.1.21.2 Function

This register defines the control bits for the PFD3-PFD0 clocks derived from the Auxiliary PLL.

When changing PFD values, the following is recommended:

1. The PFDx clock is gated first by writing a value of 1 to PFDx_CLKGATE register
2. Program the new PFD value
3. Write a value of 0 to PFDx_CLKGATE to ungate the PFDx
4. Follow by polling the PFDx_VALID flag to set and allow the PFDx clock to run.

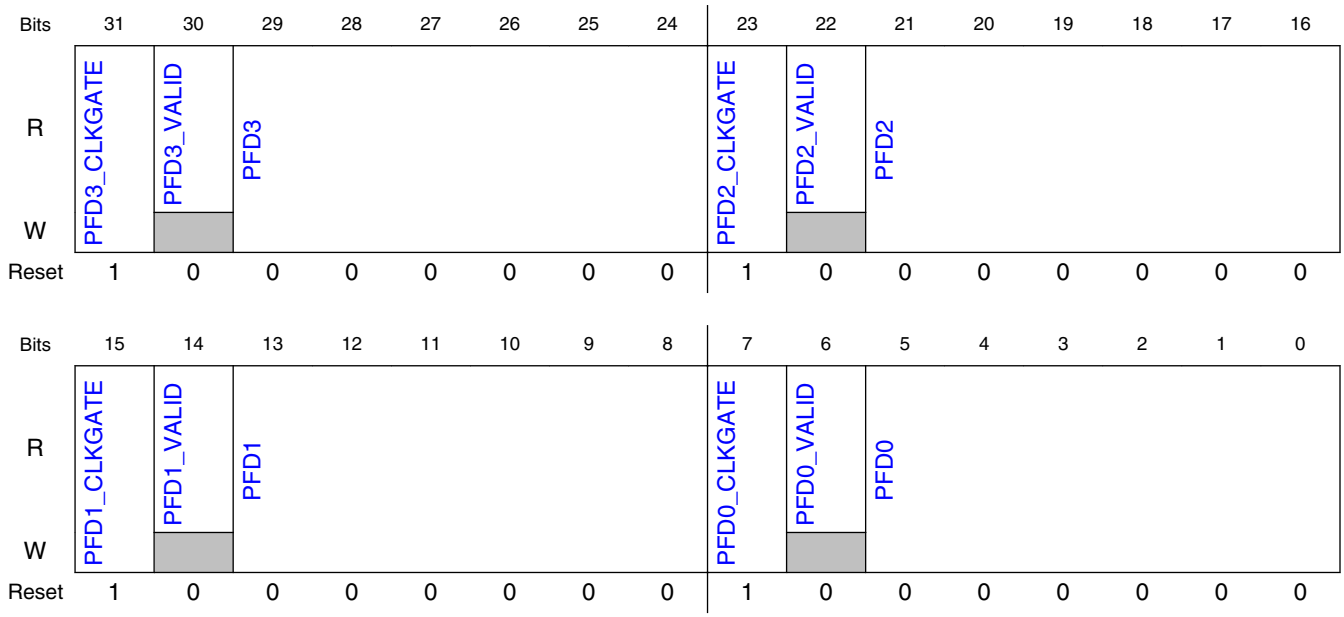
NOTE

It is recommended that PFD settings are kept between 12-35 for all PFDs.

APLL Frequency = Fref * (MULT + NUM/DENOM)

PFD Clock Frequency = PLL output frequency * 18/frac value

26.4.1.21.3 Diagram



26.4.1.21.4 Fields

Field	Function
31 PFD3_CLKGATE	PFD3_CLKGATE PFD3 Clock Gate. 0b - PFD3 clock is not gated. 1b - PFD3 clock is gated.
30 PFD3_VALID	PFD3_VALID Indicates when PFD3 clock is valid. Will clear when PFD3 is written with a new value and will set after PFD3 clock becomes stable.
29-24 PFD3	PLL Fractional Divider 3 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
23 PFD2_CLKGATE	PFD2_CLKGATE PFD2 Clock Gate. 0b - PFD2 clock is not gated. 1b - PFD2 clock is gated.
22 PFD2_VALID	PFD2_VALID Indicates when PFD2 clock is valid. Will clear when PFD2 is written with a new value and will set after PFD2 clock becomes stable.
21-16 PFD2	PLL Fractional Divider 2 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
15	PFD1_CLKGATE PFD1 Clock Gate.

Table continues on the next page...

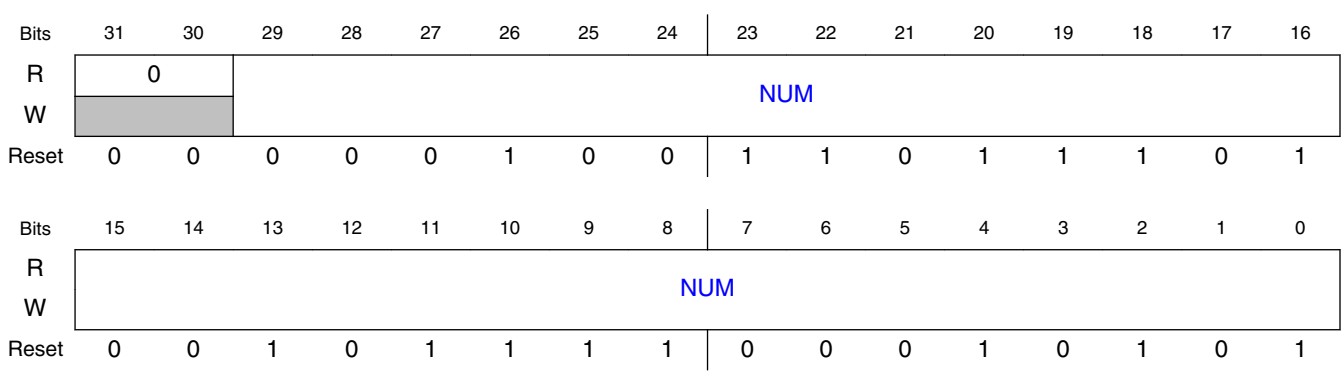
Field	Function
PFD1_CLKGATE	0b - PFD1 clock is not gated. 1b - PFD1 clock is gated.
14 PFD1_VALID	PFD1_VALID Indicates when PFD1 clock is valid. Will clear when PFD1 is written with a new value and will set after PFD1 clock becomes stable.
13-8 PFD1	PLL Fractional Divider 1 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
7 PFD0_CLKGATE	PFD0_CLKGATE PFD0 Clock Gate. 0b - PFD0 clock is not gated. 1b - PFD0 clock is gated.
6 PFD0_VALID	PFD0_VALID Indicates when PFD0 clock is valid. Will clear when PFD0 is written with a new value and will set after PFD0 clock becomes stable.
5-0 PFD0	PLL Fractional Divider 0 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac

26.4.1.22 Auxiliary PLL Numerator Register (APLLNUM)

26.4.1.22.1 Offset

Register	Offset
APLLNUM	510h

26.4.1.22.2 Diagram



26.4.1.22.3 Fields

Field	Function
31-30 —	Reserved
29-0 NUM	30-bit numerator of the Auxiliary PLL Fractional-Loop divider NOTE: The value of numerator must always be configured to be less than the value of the denominator.

26.4.1.23 Auxiliary PLL Denominator Register (APLLDENOM)

26.4.1.23.1 Offset

Register	Offset
APLLDENOM	514h

26.4.1.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DENOM													
W																
Reset	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DENOM															
W																
Reset	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1

26.4.1.23.3 Fields

Field	Function
31-30 —	Reserved
29-0 DENOM	30-bit denominator of the Auxiliary PLL Fractional-Loop divider The value of numerator must always be configured to be less than the value of the denominator. **NOTE:** The DENOM value must not be zero, because that would cause division by zero, which is undefined in mathematics.

26.4.1.24 Auxiliary PLL Spread Spectrum Register (APLLSS)

26.4.1.24.1 Offset

Register	Offset
APLLSS	518h

26.4.1.24.2 Function

This register controls the APLL spread spectrum modulation. Spread spectrum modulation is used to reduce EMI.

26.4.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STOP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ENABLE	STEP														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.1.24.4 Fields

Field	Function
31-16 STOP	STOP and STEP together control the modulation depth (maximum frequency change) and modulation depth. Modulation Depth = (STOP/DENOM)*Fref where DENOM is the DENOM field value in DENOM register. Modulation Frequency = (STEP/(2*STOP))*Fref, where Fref = 24Mhz.
15 ENABLE	Enables the spread spectrum modulation. 0b - Spectrum modulation is disabled 1b - Spectrum modulation is enabled
14-0 STEP	STOP and STEP together control the modulation depth (maximum frequency change) and modulation frequency. Modulation Depth = (STOP/DENOM)*Fref where DENOM is the DENOM field value in DENOM register. Modulation Frequency = (STEP/(2*STOP))*Fref, where Fref = 24Mhz.

26.4.1.25 Auxiliary PLL LOCK Configuration Register (APLLLOCK_CNFG)

26.4.1.25.1 Offset

Register	Offset
APLLLOCK_CNFG	5F8h

26.4.1.25.2 Function

NOTE

The reset value for this register is chip-specific. See chip-specific section for more information.

26.4.1.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCK_TIME															
W																
Reset	0	0	0	0	0	1	0	0	0	1	1	0	1	0	0	0

26.4.1.25.4 Fields

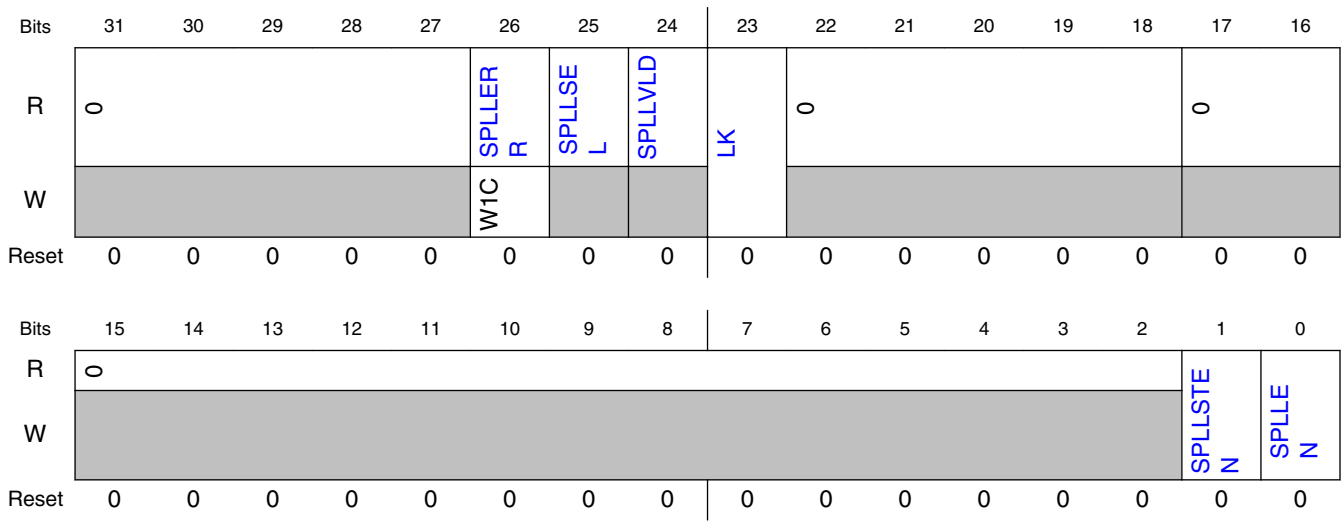
Field	Function
31-16 —	Reserved
15-0 LOCK_TIME	Configures the number of reference clocks to count before APLL is considered locked and valid.

26.4.1.26 System PLL Control Status Register (SPLLCSR)

26.4.1.26.1 Offset

Register	Offset
SPLLCSR	600h

26.4.1.26.2 Diagram



26.4.1.26.3 Fields

Field	Function
31-27 —	Reserved
26 SPLLERR	System PLL Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0b - System PLL Clock Monitor is disabled or has not detected an error 1b - System PLL Clock Monitor is enabled and detected an error. System PLL Clock Error flag will not set when System OSC is selected as its source and SOSCERR has set.
25 SPLLSEL	System PLL Selected 0b - System PLL is not the system clock source 1b - System PLL is the system clock source
24 SPLLVLD	System PLL Valid Indicates when the SPLL clock is valid. Disabling the SPLL or a SOSC error when selected as the reference clock to the SPLL will cause the SPLLVLD to clear without setting SPLLERROR. **NOTE:** The System PLL Valid bit (SPLLVLD) should only be used to verify that the SPLL is locked after initialization. 0b - System PLL is not enabled or clock is not valid 1b - System PLL is enabled and output clock is valid
23	Lock Register

Table continues on the next page...

Field	Function
LK	This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-18 —	Reserved
17-16 —	Reserved
15-2 —	Reserved
1 SPLLSTEN	System PLL Stop Enable 0b - System PLL is disabled in Stop modes 1b - System PLL is enabled in Stop modes
0 SPLLEN	System PLL Enable **When configuring the SPLL, SPLLEN should be set/cleared first. Then the rest of the configuration process can be completed (disable PFDs, etc).** 0b - System PLL is disabled 1b - System PLL is enabled

26.4.1.27 System PLL Divide Register (SPLLDIV)

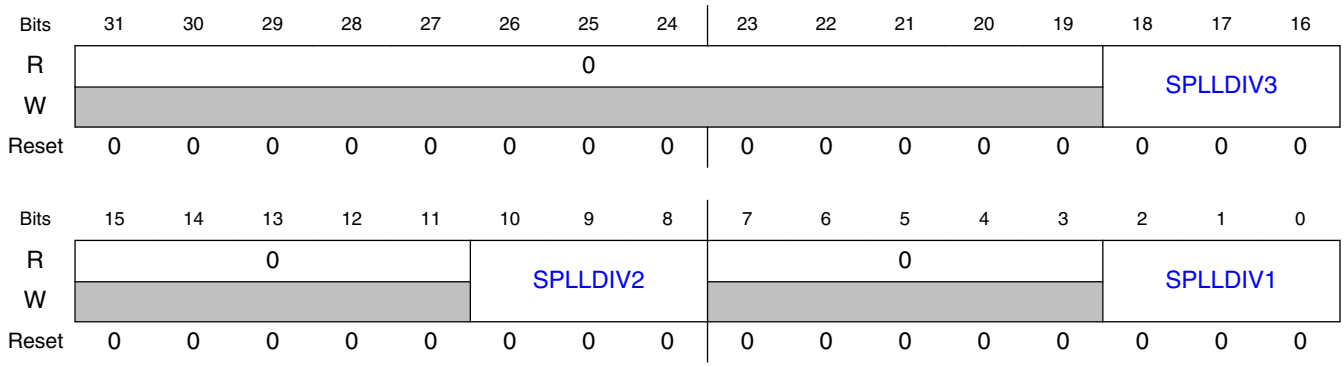
26.4.1.27.1 Offset

Register	Offset
SPLLDIV	604h

26.4.1.27.2 Function

Changes to SPLLDIV should be done when System PLL is disabled to prevent glitches to output divided clock.

26.4.1.27.3 Diagram



26.4.1.27.4 Fields

Field	Function
31-19 —	Reserved
18-16 SPLLDIV3	System PLL Clock Divide 3 Clock divider 3 for System PLL. Used by slow clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved
10-8 SPLLDIV2	System PLL Clock Divide 2 Clock divider 2 for System PLL. Used by bus clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 SPLLDIV1	System PLL Clock Divide 1 Clock divider 1 for System PLL. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1

Field	Function
	010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.1.28 System PLL Configuration Register (SPLLCFG)

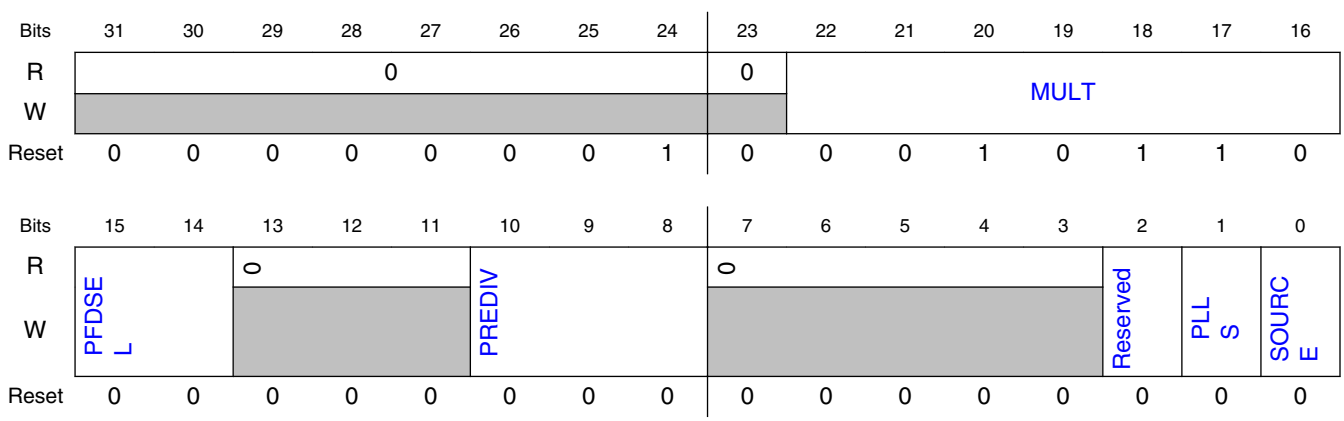
26.4.1.28.1 Offset

Register	Offset
SPLLCFG	608h

26.4.1.28.2 Function

The SPLLCFG register cannot be changed when the System PLL is enabled. When the System PLL is enabled, writes to this register are ignored, and there is no transfer error.

26.4.1.28.3 Diagram



26.4.1.28.4 Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Field	Function
23 —	Reserved
22-16 MULT	System PLL Multiplier Multiplier for the System PLL. The MULT bits establish the multiplication factor applied to the PLL reference clock frequency. • PLL Output Frequency = Divided Reference Frequency * MULT • Valid Div Input Reference frequency are (in MHz): 24 (default), 32, 30, 26, 19.2 and 16 • Valid MULT values are 33, 27, 22, 20, 17, 16
15-14 PFDSEL	PFD Clock Select Selects which PFD output clock will be used as a SCG system reference clock or as a SCG DDR reference clock. 00b - PFD0 output clock selected. 01b - PFD1 output clock selected. 10b - PFD2 output clock selected. 11b - PFD3 output clock selected.
13-11 —	Reserved
10-8 PREDIV	PLL Reference Clock Divider Selects the amount to divide down the reference clock for the System PLL. The resulting divided frequency one of the following valid reference frequencies: 24MHz(default), 32, 30, 26, 19.2 and 16.
7-3 —	Reserved
2 —	Reserved This field is reserved. Software should write 0 to this bit to maintain compatibility.
1 PLLS	PLL Select Selects the PLL or PFD as the SPLL output used to generate system clocks. 0b - SPLL output clocks selected 1b - SPLL PFD output clock selected.
0 SOURCE	Clock Source Configures the input clock source for the System PLL. 0b - System OSC (SOSC) 1b - Reserved

Table 26-5. System PLL Reference Divide Factor

PREDIV	Divide Factor
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

26.4.1.29 System PLL PFD Register (SPLLPFD)

26.4.1.29.1 Offset

Register	Offset
SPLLPFD	60Ch

26.4.1.29.2 Function

This register defines the control bits for the PFD3-PFD0 clocks derived from the System PLL.

When changing PFD values, the following is recommended:

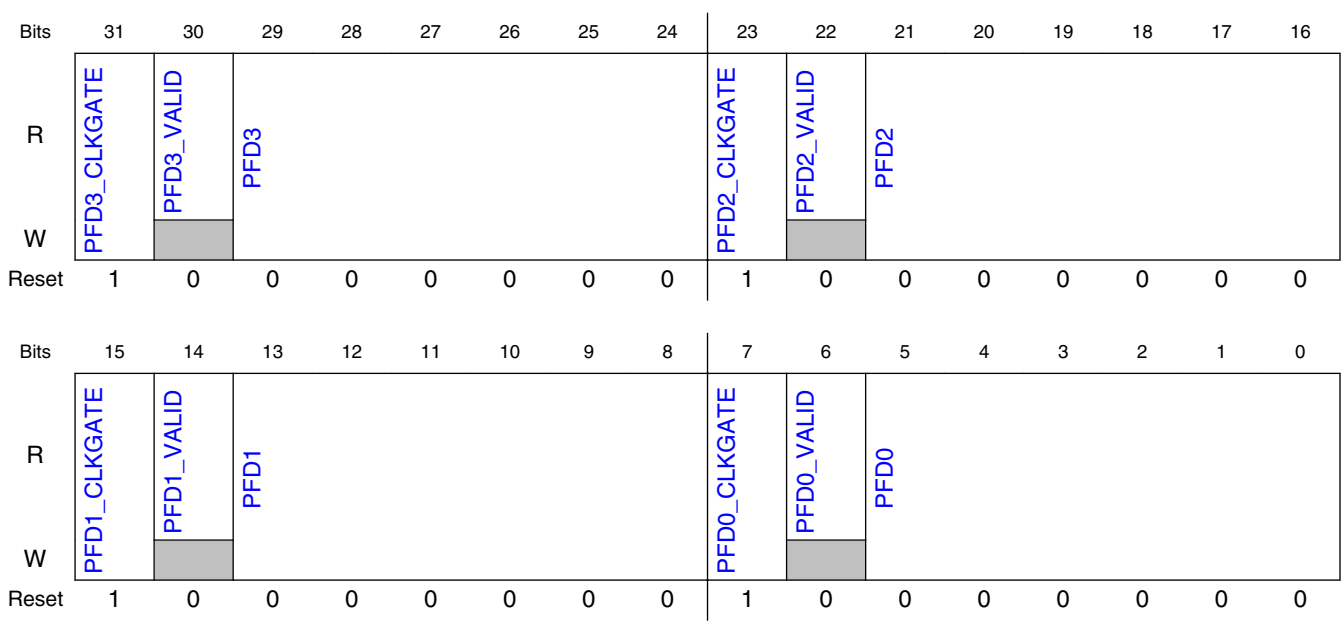
1. The PFDx clock is gated first by writing a value of 1 to PFDx_CLKGATE register
2. Program the new PFD value
3. Write a value of 0 to PFDx_CLKGATE to ungate the PFDx
4. Follow by polling the PFDx_VALID flag to set and allow the PFDx clock to run.

NOTE: It is recommended that PFD settings are kept between 12-35 for all PFDs.

SPLL Frequency = Fref * (MULT + NUM/DENOM)

PFD Clock Frequency = PLL output frequency * 18/frac value

26.4.1.29.3 Diagram



26.4.1.29.4 Fields

Field	Function
31 PFD3_CLKGATE	PFD3_CLKGATE PFD3 Clock Gate. 0b - PFD3 clock is not gated. 1b - PFD3 clock is gated.
30 PFD3_VALID	PFD3_VALID Indicates when PFD3 clock is valid. Will clear when PFD3 is written with a new value and will set after PFD3 clock becomes stable.
29-24 PFD3	PLL Fractional Divider 3 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
23 PFD2_CLKGATE	PFD2_CLKGATE PFD2 Clock Gate. 0b - PFD2 clock is not gated. 1b - PFD2 clock is gated.
22 PFD2_VALID	PFD2_VALID Indicates when PFD2 clock is valid. Will clear when PFD2 is written with a new value and will set after PFD2 clock becomes stable.
21-16 PFD2	PLL Fractional Divider 2 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
15 PFD1_CLKGATE	PFD1_CLKGATE PFD1 Clock Gate. 0b - PFD1 clock is not gated. 1b - PFD1 clock is gated.
14 PFD1_VALID	PFD1_VALID Indicates when PFD1 clock is valid. Will clear when PFD1 is written with a new value and will set after PFD1 clock becomes stable.
13-8 PFD1	PLL Fractional Divider 5 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
7 PFD0_CLKGATE	PFD0_CLKGATE PFD0 Clock Gate. 0b - PFD0 clock is not gated. 1b - PFD0 clock is gated.
6 PFD0_VALID	PFD0_VALID Indicates when PFD0 clock is valid. Will clear when PFD0 is written with a new value and will set after PFD0 clock becomes stable.
5-0 PFD0	PLL Fractional Divider 0 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac

26.4.1.30 System PLL Numerator Register (SPLLNUM)

26.4.1.30.1 Offset

Register	Offset
SPLLNUM	610h

26.4.1.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		NUM													
W																
Reset	0	0	0	0	0	1	0	0	1	1	0	1	1	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NUM															
W																
Reset	0	0	1	0	1	1	1	1	0	0	0	1	0	1	0	1

26.4.1.30.3 Fields

Field	Function
31-30 —	Reserved
29-0 NUM	30-bit numerator of the System PLL Fractional-Loop divider NOTE: The value of numerator must always be configured to be less than the value of the denominator.

26.4.1.31 System PLL Denominator Register (SPLLDENOM)

26.4.1.31.1 Offset

Register	Offset
SPLLDENOM	614h

26.4.1.31.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DENOM													
W																
Reset	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DENOM															
W																
Reset	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1

26.4.1.31.3 Fields

Field	Function
31-30 —	Reserved
29-0 DENOM	30-bit denominator of the System PLL Fractional-Loop divider The value of numerator must always be configured to be less than the value of the denominator. **NOTE:** The DENOM value must not be zero, because that would cause division by zero, which is undefined in mathematics.

26.4.1.32 System PLL Spread Spectrum Register (SPLLSS)

26.4.1.32.1 Offset

Register	Offset
SPLLSS	618h

26.4.1.32.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STOP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ENABLE	STEP														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.1.32.3 Fields

Field	Function
31-16 STOP	STOP and STEP together control the modulation depth (maximum frequency change) and Modulation Depth. Modulation Depth = (STOP/DENOM)*Fref where DENOM is the DENOM field value in DENOM register. Modulation Frequency = (STEP/(2*STOP))*Fref, where Fref = 24Mhz.
15 ENABLE	Enables the spread spectrum modulation. 0b - Spectrum modulation is disabled 1b - Spectrum modulation is enabled
14-0 STEP	STOP and STEP together control the modulation depth (maximum frequency change) and Modulation Depth. Modulation Depth = (STOP/DENOM)*Fref where DENOM is the DENOM field value in DENOM register. Modulation Frequency = (STEP/(2*STOP))*Fref, where Fref = 24Mhz.

26.4.1.33 System PLL LOCK Configuration Register (SPLLLOCK_CNFG)

26.4.1.33.1 Offset

Register	Offset
SPLLLOCK_CNFG	6F8h

26.4.1.33.2 Function

NOTE

The reset value for this register is chip-specific. See chip-specific section for more information.

26.4.1.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCK_TIME															
W																
Reset	0	0	0	0	0	1	0	0	0	1	1	0	1	0	0	0

26.4.1.33.4 Fields

Field	Function
31-16 —	Reserved
15-0 LOCK_TIME	Configures the number of reference clocks to count before SPLL is considered locked and valid.

26.4.2 SCG register descriptions

- For any writeable SCG registers, only 32-bit writes are allowed. 8-bit or 16-bit writes will result in transfer errors.
- For PLL programming, enable the PLL first, and then the PFD can be programmed.

26.4.2.1 SCG Memory map

SCG0 base address: 4102_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_0000h
4h	Parameter Register (PARAM)	32	RO	D800_007Eh
10h	Clock Status Register (CSR)	32	RO	See description.

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
14h	Run Clock Control Register (RCCR)	32	RW	See description.
18h	VLPR Clock Control Register (VCCR)	32	RW	See description.
1Ch	HSRUN Clock Control Register (HCCR)	32	RW	0300_0001h
20h	SCG CLKOUT Configuration Register (CLKOUTCNFG)	32	RW	0300_0000h
100h	System OSC Control Status Register (SOSCCSR)	32	RW	See description.
104h	System OSC Divide Register (SOSCDIV)	32	RW	0000_0000h
108h	System Oscillator Configuration Register (SOSCCFG)	32	RW	0000_0000h
200h	Slow IRC Control Status Register (SIRCCSR)	32	RW	0000_0005h
204h	Slow IRC Divide Register (SIRCDIV)	32	RW	0000_0000h
208h	Slow IRC Configuration Register (SIRCCFG)	32	RW	0000_0001h
300h	Fast IRC Control Status Register (FIRCCSR)	32	RW	0300_2001h
304h	Fast IRC Divide Register (FIRCDIV)	32	RW	0000_0000h
308h	Fast IRC Configuration Register (FIRCCFG)	32	RW	0000_0000h
30Ch	Fast IRC Trim Configuration Register (FIRCTCFG)	32	RW	0000_0000h
318h	Fast IRC Status Register (FIRCSTAT)	32	RW	See description.
400h	RTC OSC Control Status Register (ROSCCSR)	32	RW	See description.
500h	Auxiliary PLL Control Status Register (APLLCSR)	32	RW	0000_0000h
504h	Auxiliary PLL Divide Register (APLLDIV)	32	RW	0000_0000h
508h	Auxiliary PLL Configuration Register (APLLCFG)	32	RW	0116_0000h
50Ch	Auxiliary PLL PFD Register (APLLPFD)	32	RW	8080_8080h
510h	Auxiliary PLL Numerator Register (APLLNUM)	32	RW	04DD_2F15h
514h	Auxiliary PLL Denominator Register (APLLDENOM)	32	RW	1FFF_FFDBh
518h	Auxiliary PLL Spread Spectrum Register (APLLSS)	32	RW	0000_0000h
5F8h	Auxiliary PLL LOCK Configuration Register (APLLLOCK_CNFG)	32	RW	0000_0468h
600h	System PLL Control Status Register (SPLLCSR)	32	RW	0000_0000h
604h	System PLL Divide Register (SPLLDIV)	32	RW	0000_0000h
608h	System PLL Configuration Register (SPLLCFG)	32	RW	0003_0000h
60Ch	System PLL PFD Register (SPLLPFD)	32	RW	8080_8080h
6F8h	System PLL LOCK Configuration Register (SPLLLOCK_CNFG)	32	RW	0000_0168h

26.4.2.2 Version ID Register (VERID)

26.4.2.2.1 Offset

Register	Offset
VERID	0h

26.4.2.2.2 Function

Note: Writing to this register will result in a transfer error.

26.4.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VERSION															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VERSION															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.2.2.4 Fields

Field	Function
31-0 VERSION	SCG Version Number

26.4.2.3 Parameter Register (PARAM)

26.4.2.3.1 Offset

Register	Offset
PARAM	4h

26.4.2.3.2 Function

Note: Writing to this register will result in a transfer error.

26.4.2.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DIVPRES					0										
W																
Reset	1[1]	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CLKPRES							
W																
Reset	0	0	0	0	0	0	0	0	0[2]	1	1	1	1	1	1	0

1. The reset value is controlled by which SCG System Dividers are used by the SoC.
2. The reset value is controlled by which SCG Clock Sources are used by the SoC. Please reference the Reference manual clocking chapter.

26.4.2.3.4 Fields

Field	Function
31-27 DIVPRES	Divider Present Indicates which system clock dividers are present in this instance of SCG. 1xxxxb - System DIVCORE is present. x1xxxb - System DIVPLAT is present. xxx1xb - System DIVBUS is present. xxxx1b - System DIVSLOW is present.
26-16 —	Reserved
15-8 —	Reserved
7-0 CLKPRES	Clock Present Indicates which clock sources are present in this instance of SCG. Any bits not defined in this bit field are Reserved and always has the value 0 when read. 00000000-00000001b - Reserved. x1xxxxxxb - System PLL (SPLL) is present. xx1xxxxxb - Auxiliary PLL (APLL) is present. xxx1xxxxb - RTC OSC (ROSC) is present. xxxx1xxxb - Fast IRC (FIRC) is present. xxxxx1xxb - Slow IRC (SIRC) is present. xxxxxx1xb - System OSC (SOSC) is present.

26.4.2.4 Clock Status Register (CSR)

26.4.2.4.1 Offset

Register	Offset
CSR	10h

26.4.2.4.2 Function

The Clock Status Register (CSR) returns the currently configured system clock source and the system clock dividers for the core (DIVCORE) and peripheral interface clock (DIVSLOW). The SCG_CSR reports the configuration set by one of the clock control registers SCG_RCCR, SCG_VCCR, SCG_HCCR.

Note: Writing to the Clock Status Register (CSR) will result in a transfer error.

26.4.2.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	u[1]	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIVPLAT				0				DIVBUS				DIVSLOW			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

1. The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-1 or div-by-2 when resetting into RUN mode or div-by-4 or div-by-8 when resetting into VLPR mode

26.4.2.4.4 Fields

Field	Function
31-28 —	Reserved
27-24 SCS	System Clock Source Returns the currently configured clock source generating the system clock. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - RTC OSC (ROSC_CLK) 0101b - Auxiliary PLL (APLL_CLK) 0110b - System PLL (SPLL_CLK)

Table continues on the next page...

Field	Function
	0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 DIVPLAT	Platform Clock Divide Ratio 0000b - Divide-by-1 0001b - Reserved 0010b - Reserved 0011b - Reserved 0100b - Reserved 0101b - Reserved 0110b - Reserved 0111b - Reserved 1000b - Reserved 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - Reserved
11-8 —	Reserved
7-4 DIVBUS	Bus Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14

Table continues on the next page...

Field	Function
	1110b - Divide-by-15 1111b - Divide-by-16
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

26.4.2.5 Run Clock Control Register (RCCR)

26.4.2.5.1 Offset

Register	Offset
RCCR	14h

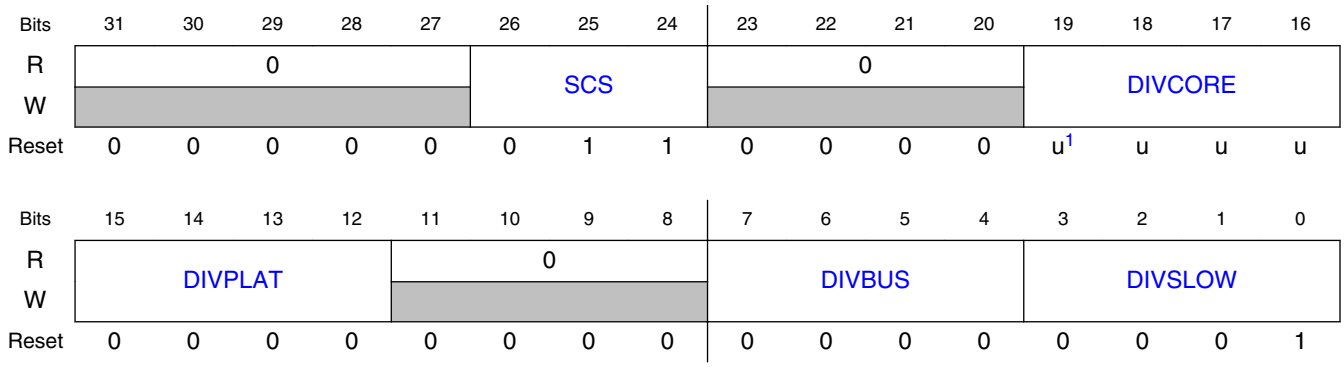
26.4.2.5.2 Function

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in Run mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in RUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in RUN, new system clock divide ratios will not take affect until a new clock source is valid.

NOTE

Switching to new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

26.4.2.5.3 Diagram



1. The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-1 and div-by-2

26.4.2.5.4 Fields

Field	Function
31-27 —	Reserved
26-24 SCS	System Clock Source Selects the clock source generating the system clock in Run mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in Run mode requires that clock source to be enabled first and be valid before system clocks are allowed to switch to that clock source. **NOTE:** Programming SCS to a different value should only be done after the previous SCS clock switch has finished. 000b - Reserved 001b - System OSC (SOSC_CLK) 010b - Slow IRC (SIRC_CLK) 011b - Fast IRC (FIRC_CLK) 100b - RTC OSC (ROSC_CLK) 101b - Auxiliary PLL (APLL_CLK) 110b - System PLL (SPLL_CLK) 111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12

Table continues on the next page...

Field	Function
	1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 DIVPLAT	Platform Clock Divide Ratio Only Divide-by-1 is supported. Software should write 0000b to this bit field to maintain compatibility. 0000b - Divide-by-1 0001b - Reserved 0010b - Reserved 0011b - Reserved 0100b - Reserved 0101b - Reserved 0110b - Reserved 0111b - Reserved 1000b - Reserved 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - Reserved
11-8 —	Reserved
7-4 DIVBUS	Bus Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13

Field	Function
	1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

26.4.2.6 VLPR Clock Control Register (VCCR)

26.4.2.6.1 Offset

Register	Offset
VCCR	18h

26.4.2.6.2 Function

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in VLPR mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in VLPR requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in VLPR, new system clock divide ratios will not take affect until new clock source is valid.

NOTE

Switching to a new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

26.4.2.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				SCS				0				DIVCORE			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	u[1]	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIVPLAT				0				DIVBUS				DIVSLOW			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

1. The reset value is controlled by user FOPT bits that get uploaded during reset. The two option reset values are div-by-4 and div-by-8.

26.4.2.6.4 Fields

Field	Function
31-28 —	Reserved
27-24 SCS	System Clock Source Selects the clock source generating the system clock in VLPR mode. Attempting to select a clock that is not valid will be ignored. Selects the clock source generating the system clock. Selecting a different clock source when in VLPR mode requires that clock source to be enabled first and be valid before system clocks switch to that clock source. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - RTC OSC (ROSC_CLK) 0101b - Reserved 0110b - Reserved 0111b - Reserved
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 DIVPLAT	Platform Clock Divide Ratio Only Divide-by-1 is supported. Software should write 0000b to this bit field to maintain compatibility. 0000b - Divide-by-1 0001b - Reserved 0010b - Reserved 0011b - Reserved 0100b - Reserved 0101b - Reserved 0110b - Reserved 0111b - Reserved 1000b - Reserved 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Reserved 1101b - Reserved

Table continues on the next page...

Field	Function
	1110b - Reserved 1111b - Reserved
11-8 —	Reserved
7-4 DIVBUS	Bus Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

26.4.2.7 HSRUN Clock Control Register (HCCR)

26.4.2.7.1 Offset

Register	Offset
HCCR	1Ch

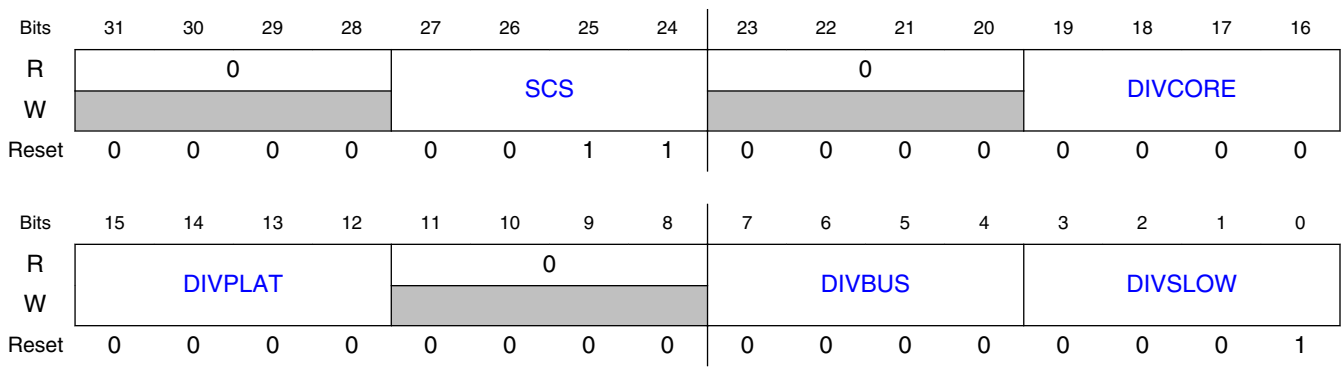
26.4.2.7.2 Function

This register controls the system clock source and the system clock dividers for the core, platform, external and bus clock domains when in HSRUN mode only. This register can only be written using a 32-bit write. Selecting a different clock source when in HSRUN requires that clock source to be enabled first and be valid before system clocks switch to that clock source. If system clock divide ratios also change when selecting a different clock mode when in HSRUN, new system clock divide ratios will not take affect until new clock source is valid.

NOTE

Switching to a new system clock source and system clock dividers must be done after previous RCCR changes have been completed.

26.4.2.7.3 Diagram



26.4.2.7.4 Fields

Field	Function
31-28 —	Reserved
27-24 SCS	System Clock Source Selects the clock source generating the system clock in HSRUN mode. Attempting to select a clock that is not valid will be ignored. Selecting a different clock source when in HSRUN mode will enable that clock source and switch to that clock mode when it is valid. 0000b - Reserved 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - RTC OSC (ROSC_CLK) 0101b - Auxiliary PLL (APLL_CLK) 0110b - System PLL (SPLL_CLK) 0111b - Reserved

Table continues on the next page...

Field	Function
23-20 —	Reserved
19-16 DIVCORE	Core Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
15-12 DIVPLAT	Platform Clock Divide Ratio Only Divide-by-1 is supported. Software should write 0000b to this bit field to maintain compatibility. 0000b - Divide-by-1 0001b - Reserved 0010b - Reserved 0011b - Reserved 0100b - Reserved 0101b - Reserved 0110b - Reserved 0111b - Reserved 1000b - Reserved 1001b - Reserved 1010b - Reserved 1011b - Reserved 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - Reserved
11-8 —	Reserved
7-4 DIVBUS	Bus Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14

Table continues on the next page...

Field	Function
	1110b - Divide-by-15 1111b - Divide-by-16
3-0 DIVSLOW	Slow Clock Divide Ratio 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16

26.4.2.8 SCG CLKOUT Configuration Register (CLKOUTCNFG)

26.4.2.8.1 Offset

Register	Offset
CLKOUTCNFG	20h

26.4.2.8.2 Function

This register controls which SCG clock source is selected to be ported out to the CLKOUT pin.

26.4.2.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				CLKOUTSEL				0							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.2.8.4 Fields

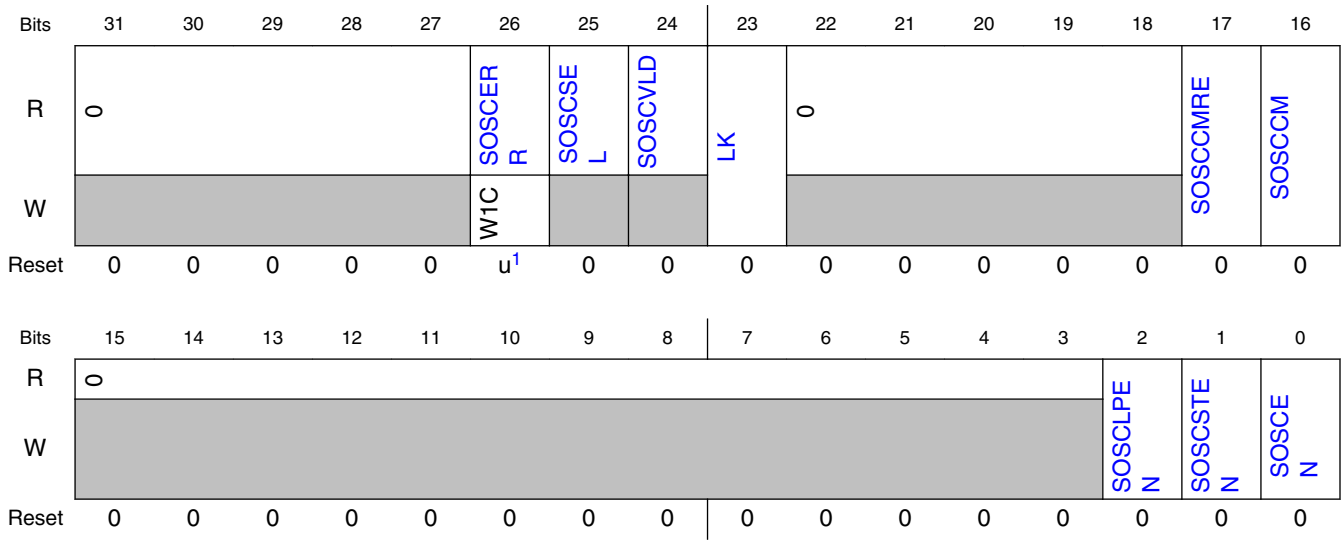
Field	Function
31-28 —	Reserved
27-24 CLKOUTSEL	SCG Clkout Select Selects the SCG system clock. 0000b - SCG SLOW Clock 0001b - System OSC (SOSC_CLK) 0010b - Slow IRC (SIRC_CLK) 0011b - Fast IRC (FIRC_CLK) 0100b - RTC OSC (ROSC_CLK) 0101b - Auxiliary PLL (APLL_CLK) 0110b - System PLL (SPLL_CLK) 0111b - Reserved 1111b - Reserved
23-0 —	Reserved

26.4.2.9 System OSC Control Status Register (SOSCCSR)

26.4.2.9.1 Offset

Register	Offset
SOSCCSR	100h

26.4.2.9.2 Diagram



1. This flag is reset on Chip POR only

26.4.2.9.3 Fields

Field	Function
31-27 —	Reserved
26 SOSCERR	System OSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one. 0b - System OSC Clock Monitor is disabled or has not detected an error 1b - System OSC Clock Monitor is enabled and detected an error
25 SOSCSEL	System OSC Selected 0b - System OSC is not the system clock source 1b - System OSC is the system clock source
24 SOSCVLD	System OSC Valid 0b - System OSC is not enabled or clock is not valid 1b - System OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - This Control Status Register can be written. 1b - This Control Status Register cannot be written.
22-18 —	Reserved
17 SOSCCMRE	System OSC Clock Monitor Reset Enable 0b - Clock Monitor generates interrupt when error detected 1b - Clock Monitor generates reset when error detected
16 SOSCCM	System OSC Clock Monitor Enables the clock monitor when SOSCVLD is set.

Table continues on the next page...

Field	Function
	NOTE: SOSC clock monitor remains enabled in VLPR and VLPS if SOSCCM is enabled. The clock monitor is always disabled in LLS/VLLS modes. When the clock monitor is disabled in a low power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. **NOTE:** The reference clock used to monitor the SOSC is the SIRC. This clock must be programmed to be enabled in order to monitor the SOSC. SIRC is automatically disabled in LLS and VLLS power modes and clock monitor of SOSC will be disabled. 0b - System OSC Clock Monitor is disabled 1b - System OSC Clock Monitor is enabled
15-3 —	Reserved
2 SOSCLPEN	System OSC Low Power Enable SOSCLPEN is required for low power modes. In VLPS mode (low power stop mode), if you want the clock to remain ON, then both SOSCLPEN and SOSCSTEN bits must be enabled. 0b - System OSC is disabled in VLP modes 1b - System OSC is enabled in VLP modes
1 SOSCSTEN	System OSC Stop Enable 0b - System OSC is disabled in Stop modes 1b - System OSC is enabled in Stop modes if SOSCEN=1. In VLLS0, system oscillator is disabled even if SOSCSTEN=1 and SOSCEN=1. When selected as the reference clock to the System PLL and if the System PLL is enabled in STOP mode, the SOSC will stay enabled even if SOSCSTEN=0.
0 SOSCEN	System OSC Enable 0b - System OSC is disabled 1b - System OSC is enabled

26.4.2.10 System OSC Divide Register (SOSCDIV)

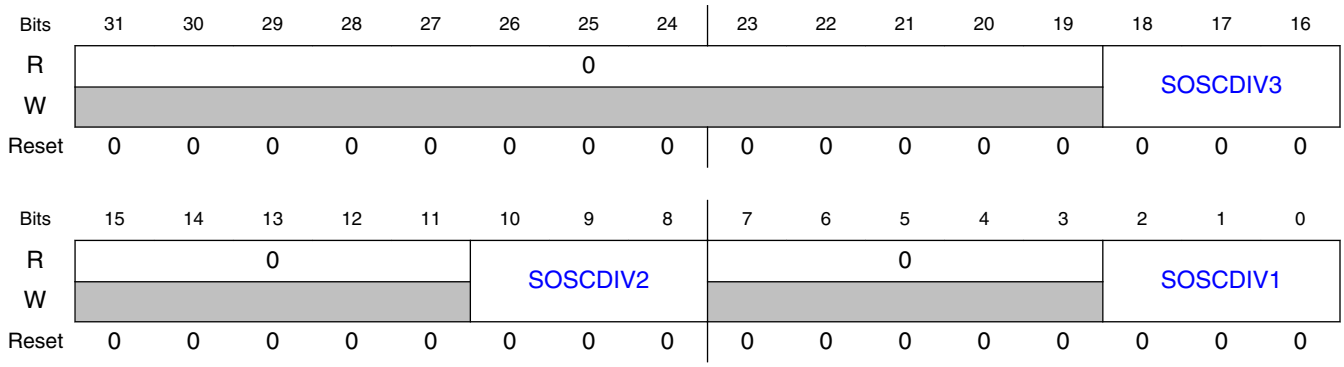
26.4.2.10.1 Offset

Register	Offset
SOSCDIV	104h

26.4.2.10.2 Function

The SCG_SOSCDIV register provides the control of 3 clock trees which can be used to provide optional peripheral functional clocks, or alternative module clocks. Each clock tree has optional dividers of the input SOSC clock. Changes to SOSCDIV should be done when System OSC is disabled to prevent glitches to output divided clock.

26.4.2.10.3 Diagram



26.4.2.10.4 Fields

Field	Function
31-19 —	Reserved
18-16 SOSCDIV3	System OSC Clock Divide 3 Clock divider 3 for System OSC. Used by slow clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved
10-8 SOSCDIV2	System OSC Clock Divide 2 Clock divider 2 for System OSC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 SOSCDIV1	System OSC Clock Divide 1 Clock divider 1 for System OSC. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1

Field	Function
	010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.2.11 System Oscillator Configuration Register (SOSCCFG)

26.4.2.11.1 Offset

Register	Offset
SOSCCFG	108h

26.4.2.11.2 Function

The SOSCCFG register cannot be changed when the System OSC is enabled. When the System OSC is enabled, writes to this register are ignored, and there is no transfer error.

26.4.2.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0		0		HGO	EREFS	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.2.11.4 Fields

Field	Function
31-12 —	Reserved
11-8	Reserved

Table continues on the next page...

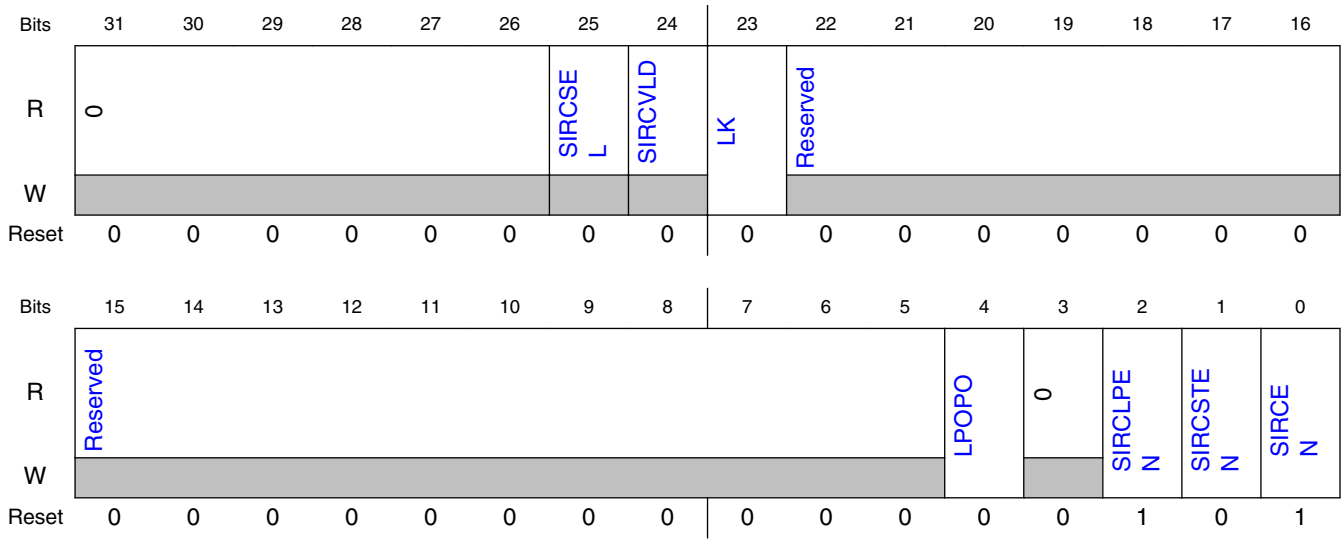
Field	Function
—	
7-6 —	Reserved
5-4 —	Reserved
3 HGO	High Gain Oscillator Select Controls the crystal oscillator power mode of operations. 0b - Configure crystal oscillator for low-power operation 1b - Configure crystal oscillator for high-gain operation
2 EREFS	External Reference Select Selects the source for the external reference clock. This bit selects which clock is output from the System OSC (SOSC) into the SCG, thus either the crystal oscillator or from an external clock input 0b - External reference clock selected 1b - Internal crystal oscillator of OSC requested. In VLLS0, the internal oscillator of OSC is disabled even if SOSCEN=1 and SOSCSTEN=1.
1-0 —	Reserved

26.4.2.12 Slow IRC Control Status Register (SIRCCSR)

26.4.2.12.1 Offset

Register	Offset
SIRCCSR	200h

26.4.2.12.2 Diagram



26.4.2.12.3 Fields

Field	Function
31-26 —	Reserved
25 SIRCSEL	Slow IRC Selected 0b - Slow IRC is not the system clock source 1b - Slow IRC is the system clock source
24 SIRCVLD	Slow IRC Valid 0b - Slow IRC is not enabled or clock is not valid 1b - Slow IRC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. The purpose for this bitfield is to prevent runaway code from changing SIRC clock configurations. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-5 —	This field is reserved and is always has the value 0
4 LPOPO	LPO Power Option Controls whether the 1khz LPO clock is enabled in LLS/VLLSx modes. 0b - LPO clock is enabled in LLS/VLLSx 1b - LPO clock is disabled in LLS/VLLSx
3 —	This field is reserved and is always has the value 0
2 SIRCLPEN	Slow IRC Low Power Enable 0b - Slow IRC is disabled in VLP modes 1b - Slow IRC is enabled in VLP modes
1	Slow IRC Stop Enable

Table continues on the next page...

Field	Function
SIRCSTEN	0b - Slow IRC is disabled in Stop modes 1b - Slow IRC is enabled in Stop modes
0 SIRCEN	Slow IRC Enable 0b - Slow IRC is disabled 1b - Slow IRC is enabled

26.4.2.13 Slow IRC Divide Register (SIRCDIV)

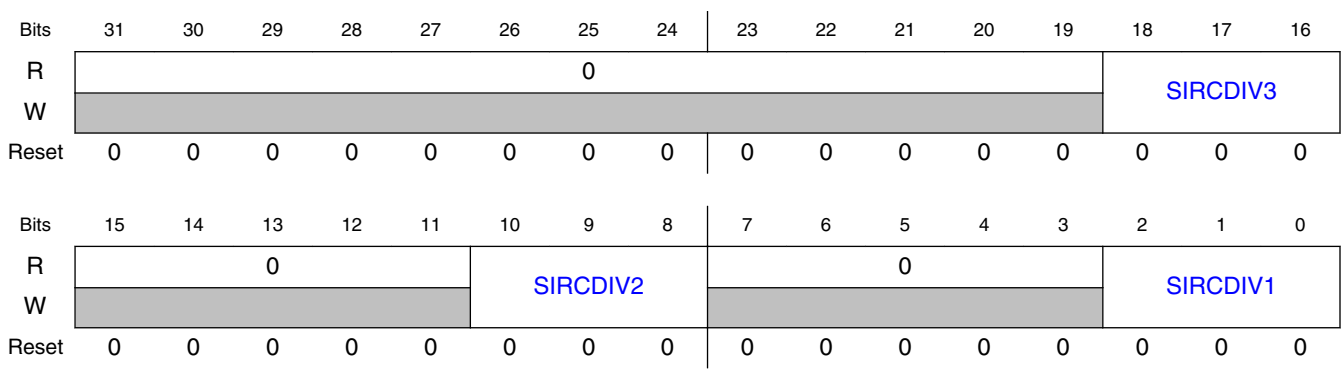
26.4.2.13.1 Offset

Register	Offset
SIRCDIV	204h

26.4.2.13.2 Function

To prevent glitches to the output divided clock, change SIRDIV when the Slow IRC is disabled.

26.4.2.13.3 Diagram



26.4.2.13.4 Fields

Field	Function
31-19 —	Reserved
18-16 SIRCDIV3	Slow IRC Clock Divider 3 Clock divider 3 for Slow IRC. Used by slow clock modules that need an asynchronous clock source.

Table continues on the next page...

Field	Function
	000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved
10-8 SIRCDIV2	Slow IRC Clock Divide 2 Clock divider 2 for Slow IRC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 SIRCDIV1	Slow IRC Clock Divide 1 Clock divider 1 for Slow IRC. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.2.14 Slow IRC Configuration Register (SIRCCFG)

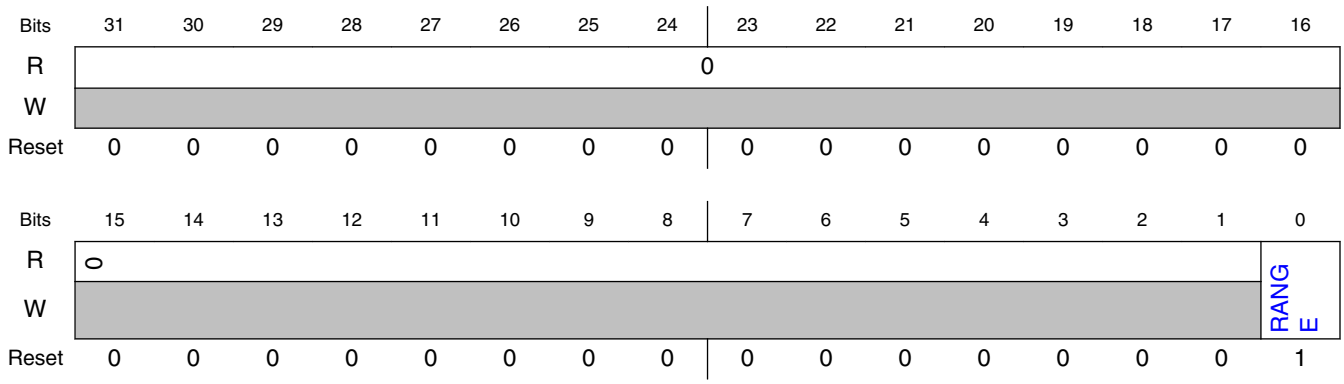
26.4.2.14.1 Offset

Register	Offset
SIRCCFG	208h

26.4.2.14.2 Function

The SIRCCFG register cannot be changed when the slow IRC clock is enabled. When the slow IRC clock is enabled, writes to this register are ignored, and there is no transfer error.

26.4.2.14.3 Diagram



26.4.2.14.4 Fields

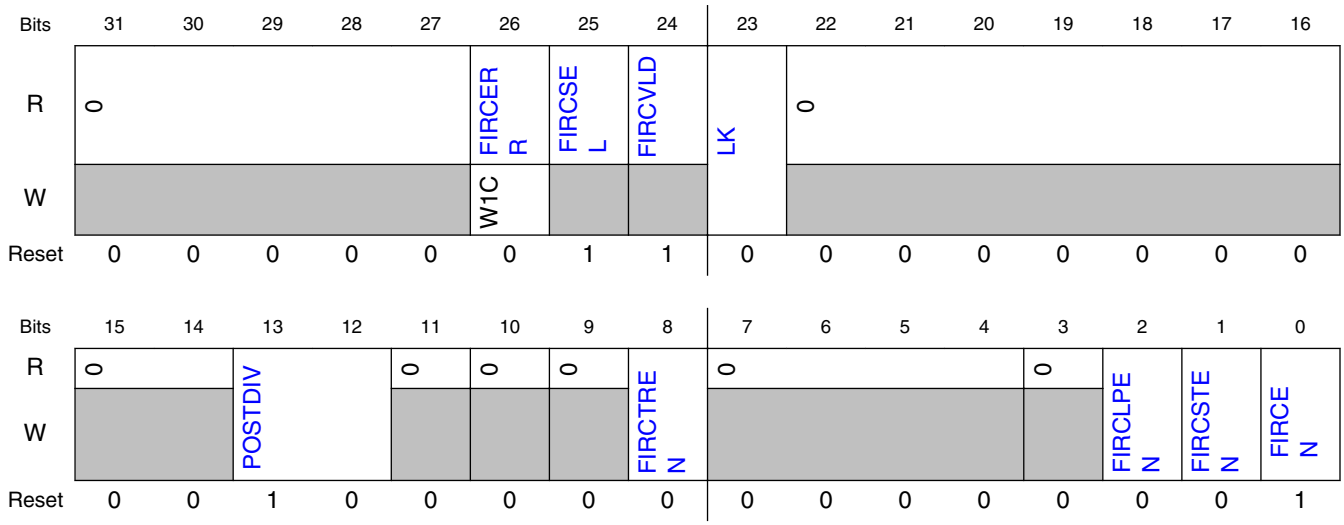
Field	Function
31-1 —	Reserved
0 RANGE	Frequency Range 0b - Slow IRC low range clock (4MHz) 1b - Slow IRC high range clock (16 MHz)

26.4.2.15 Fast IRC Control Status Register (FIRCCSR)

26.4.2.15.1 Offset

Register	Offset
FIRCCSR	300h

26.4.2.15.2 Diagram



26.4.2.15.3 Fields

Field	Function
31-27 —	Reserved
26 FIRCERR	Fast IRC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0b - Error not detected with the Fast IRC trimming. 1b - Error detected with the Fast IRC trimming.
25 FIRCSEL	Fast IRC Selected status 0b - Fast IRC is not the system clock source 1b - Fast IRC is the system clock source
24 FIRCVLD	Fast IRC Valid status 0b - Fast IRC is not enabled or clock is not valid. 1b - Fast IRC is enabled and output clock is valid. The clock is valid after there is an output clock from the FIRC analog.
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-14 —	Reserved
13-12 POSTDIV	Fast IRC Divider Divides the FIRC output clock selected by the RANGE bit. Divide values can be changed while FIRC is enabled 00b - Divide by 1 01b - Divide by 2 10b - Divide by 3

Table continues on the next page...

Field	Function
	11b - Divide by 4
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 FIRCTREN	Fast IRC Trim Enable 0b - Disable trimming Fast IRC to an external clock source 1b - Enable trimming Fast IRC to an external clock source
7-4 —	Reserved
3 —	Reserved
2 FIRCLPEN	Fast IRC Low Power Enable 0b - Fast IRC is disabled in VLP modes 1b - Fast IRC is enabled in VLP modes
1 FIRCSTEN	Fast IRC Stop Enable 0b - Fast IRC is disabled in Stop modes. 1b - Fast IRC is enabled in Stop modes
0 FIRCEN	Fast IRC Enable 0b - Fast IRC is disabled 1b - Fast IRC is enabled

26.4.2.16 Fast IRC Divide Register (FIRCDIV)

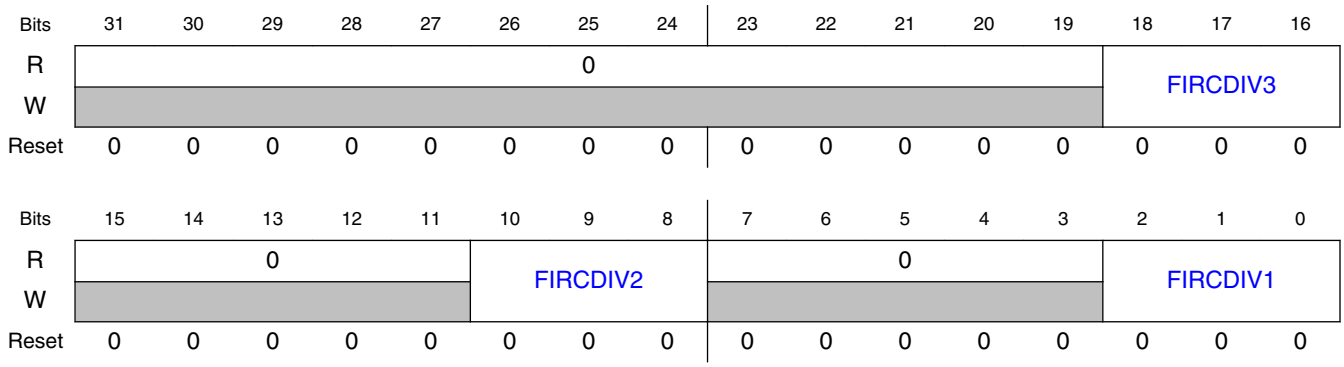
26.4.2.16.1 Offset

Register	Offset
FIRCDIV	304h

26.4.2.16.2 Function

Changes to FIRCDIV should be done when FAST IRC is disabled to prevent glitches to output divided clock.

26.4.2.16.3 Diagram



26.4.2.16.4 Fields

Field	Function
31-19 —	Reserved
18-16 FIRCDIV3	Fast IRC Clock Divider 3 Clock divider 3 for the Fast IRC. Used by slow clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved
10-8 FIRCDIV2	Fast IRC Clock Divide 2 Clock divider 2 for the Fast IRC. Used by bus clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 FIRCDIV1	Fast IRC Clock Divide 1 Clock divider 1 for Fast IRC. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Output disabled 001b - Divide by 1

Field	Function
	010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.2.17 Fast IRC Configuration Register (FIRCCFG)

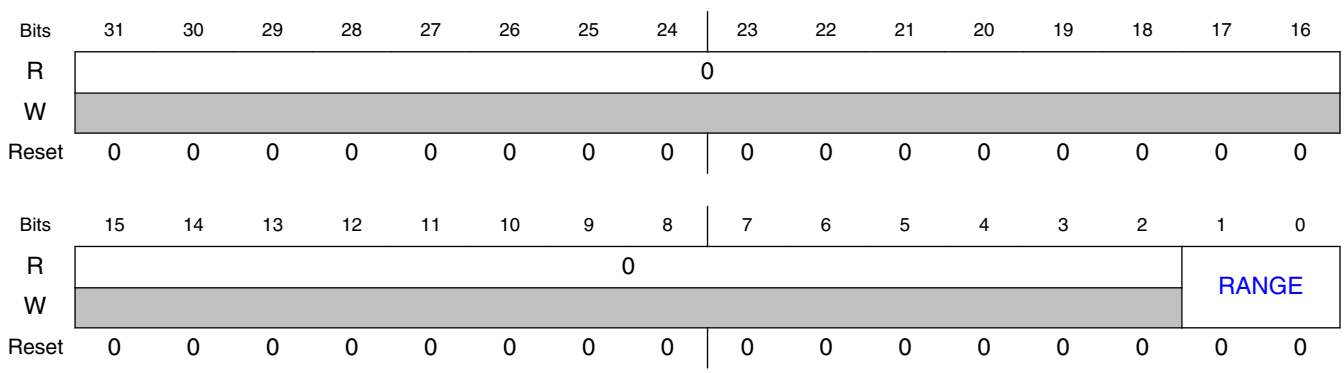
26.4.2.17.1 Offset

Register	Offset
FIRCCFG	308h

26.4.2.17.2 Function

The FIRCCFG register cannot be changed when the Fast IRC is enabled. When the Fast IRC is enabled, writes to this register are ignored, and there is no transfer error.

26.4.2.17.3 Diagram



26.4.2.17.4 Fields

Field	Function
31-2 —	Reserved
1-0	Frequency Range

Field	Function
RANGE	00b - Fast IRC is trimmed to 48 MHz 01b - Fast IRC is trimmed to 52 MHz 10b - Fast IRC is trimmed to 56 MHz 11b - Fast IRC is trimmed to 60 MHz

26.4.2.18 Fast IRC Trim Configuration Register (FIRCTCFG)

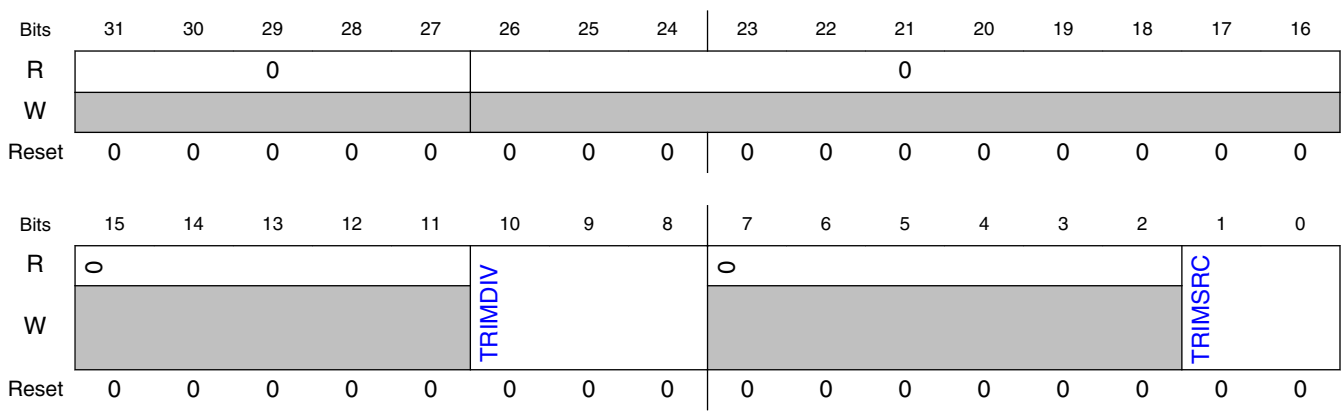
26.4.2.18.1 Offset

Register	Offset
FIRCTCFG	30Ch

26.4.2.18.2 Function

The FIRCTCFG register cannot be changed when Fast IRC tuning is enabled. When the Fast IRC tuning is enabled, writes to this register are ignored, and there is no transfer error.

26.4.2.18.3 Diagram



26.4.2.18.4 Fields

Field	Function
31-27 —	Reserved
26-16	Reserved

Table continues on the next page...

Field	Function
—	
15-11 —	Reserved
10-8 TRIMDIV	Fast IRC Trim Predivide Divide the System OSC down for Fast IRC trimming. 000b - Divide by 1 001b - Divide by 128 010b - Divide by 256 011b - Divide by 512 100b - Divide by 1024 101b - Divide by 2048 110b - Reserved. Writing this value will result in Divide by 1. 111b - Reserved. Writing this value will result in a Divide by 1.
7-2 —	Reserved
1-0 TRIMSRC	Trim Source Configures the external clock source to tune the Fast IRC. TRIMSRC must be configured before programming FIRCSTAT register for trim update 00b - Reserved 01b - Reserved 10b - System OSC. This option requires that SOSC be divided using the TRIMDIV field to get a frequency slower than 32kHz. 11b - RTC OSC (32.768 kHz)

26.4.2.19 Fast IRC Status Register (FIRCSTAT)

26.4.2.19.1 Offset

Register	Offset
FIRCSTAT	318h

26.4.2.19.2 Function

This register is loaded from IFR during reset. When FIR is enabled and FIRCTREN=1, writes to this register are allowed, and values written to this register are used to trim the FIRC clock.

26.4.2.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		TRIMCOAR						0	TRIMFINE						
W																
Reset	0	0	u[1]	u	u	u	u	u	0	u[1]	u	u	u	u	u	u

1. Reset values are loaded out of IFR.

26.4.2.19.4 Fields

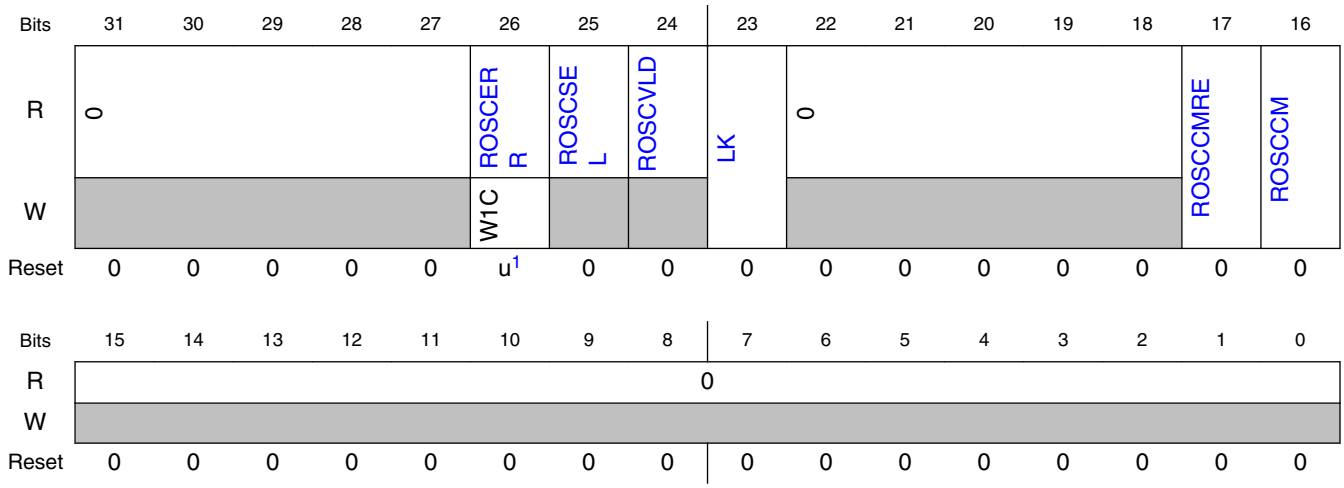
Field	Function
31-16 —	Reserved
15-14 —	Reserved
13-8 TRIMCOAR	Trim Coarse TRIMCOAR bits are used to coarsely trim the Fast IRC Clock to within approximately ±0.7% of the target frequency.
7 —	Reserved
6-0 TRIMFINE	Trim Fine Status Once the Fast IRC Clock is trimmed to ±0.7% of the target frequency using the TRIMCOAR bits, the TRIMFINE bits can be used to trim the Fast IRC Clock to within ±0.04% of the target frequency.

26.4.2.20 RTC OSC Control Status Register (ROSCCSR)

26.4.2.20.1 Offset

Register	Offset
ROSCCSR	400h

26.4.2.20.2 Diagram



1. Reset values are loaded out of IFR.

26.4.2.20.3 Fields

Field	Function
31-27 —	Reserved
26 ROSCERR	RTC OSC Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0b - RTC OSC Clock Monitor is disabled or has not detected an error 1b - RTC OSC Clock Monitor is enabled and detected an RTC loss of clock error
25 ROSCSEL	RTC OSC Selected 0b - RTC OSC is not the system clock source 1b - RTC OSC is the system clock source
24 ROSCVLD	RTC OSC Valid 0b - RTC OSC is not enabled or clock is not valid 1b - RTC OSC is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-18 —	Reserved
17 ROSCCMRE	RTC OSC Clock Monitor Reset Enable 0b - Clock Monitor generates interrupt when error detected 1b - Clock Monitor generates reset when error detected
16 ROSCCM	RTC OSC Clock Monitor Enables the clock monitor when ROSCVLD is set. **NOTE:** ROSC clock monitor remains enabled in VLPR and VLPS if ROSCCM is enabled. The clock monitor is always disabled in LLS/VLLS modes. When the clock monitor is disabled in a low

Table continues on the next page...

Field	Function
	power mode, it remains disabled until the clock valid flag is set following exit from the low power mode. **NOTE:** The reference clock used to monitor the ROSC is the SIRC. This clock must be programmed to enabled in order to monitor the ROSC. SIRC is automatically disabled in LLS and VLLS power modes and clock monitor of ROSC will be disabled. 0b - RTC OSC Clock Monitor is disabled 1b - RTC OSC Clock Monitor is enabled
15-0 —	Reserved

26.4.2.21 Auxiliary PLL Control Status Register (APLLCSR)

26.4.2.21.1 Offset

Register	Offset
APLLCSR	500h

26.4.2.21.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						APLLSEL	APLLVLD	LK	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														APLLSTEN	APLLEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.2.21.3 Fields

Field	Function
31-26	Reserved

Table continues on the next page...

Field	Function
—	
25 APLLSEL	APLL Selected 0b - APLL is not the system clock source 1b - APLL is the system clock source
24 APLLVLD	APLL Valid 0b - APLL is not enabled or clock is not valid 1b - APLL is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-2 —	Reserved
1 APLLSTEN	APLL Stop Enable 0b - APLL is disabled in Stop modes 1b - APLL is enabled in Stop modes
0 APLLEN	Auxiliary PLL (APLL) Enable **When configuring the APLL, APLLEN should be set/cleared first. Then the rest of the configuration process can be completed (disable PFDs, etc).** 0b - APLL is disabled 1b - APLL is enabled

26.4.2.22 Auxiliary PLL Divide Register (APLLDIV)

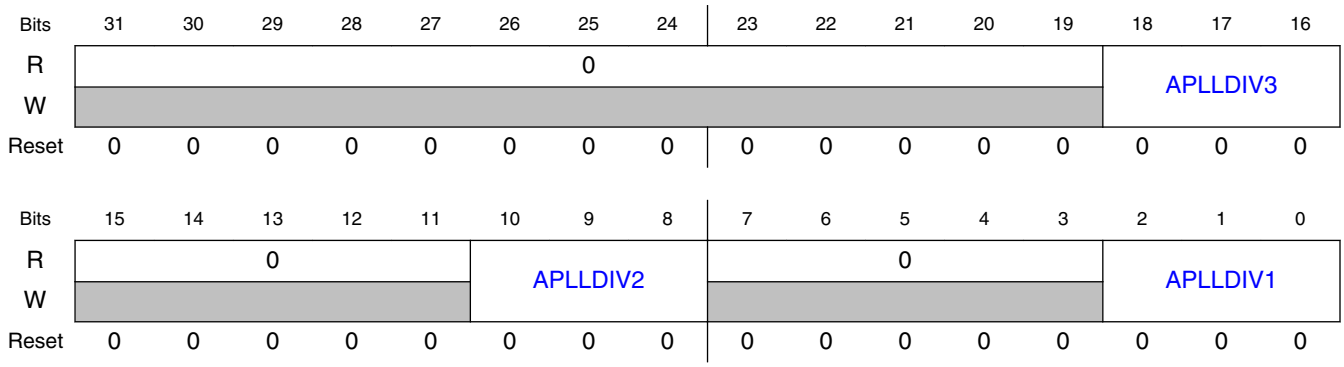
26.4.2.22.1 Offset

Register	Offset
APLLDIV	504h

26.4.2.22.2 Function

Changes to APLLDIV should be done when Auxiliary PLL is disabled to prevent glitches to output divided clock.

26.4.2.22.3 Diagram



26.4.2.22.4 Fields

Field	Function
31-19 —	Reserved
18-16 APLLDIV3	Auxiliary PLL Clock Divide 3 Clock divider 3 for Auxiliary PLL. Used by slow clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved
10-8 APLLDIV2	Auxiliary PLL Clock Divide 2 Clock divider 2 for Auxiliary PLL. Used by bus clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 APLLDIV1	Auxiliary PLL Clock Divide 1 Clock divider 1 for Auxiliary PLL. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1

Field	Function
	010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.2.23 Auxiliary PLL Configuration Register (APLLCFG)

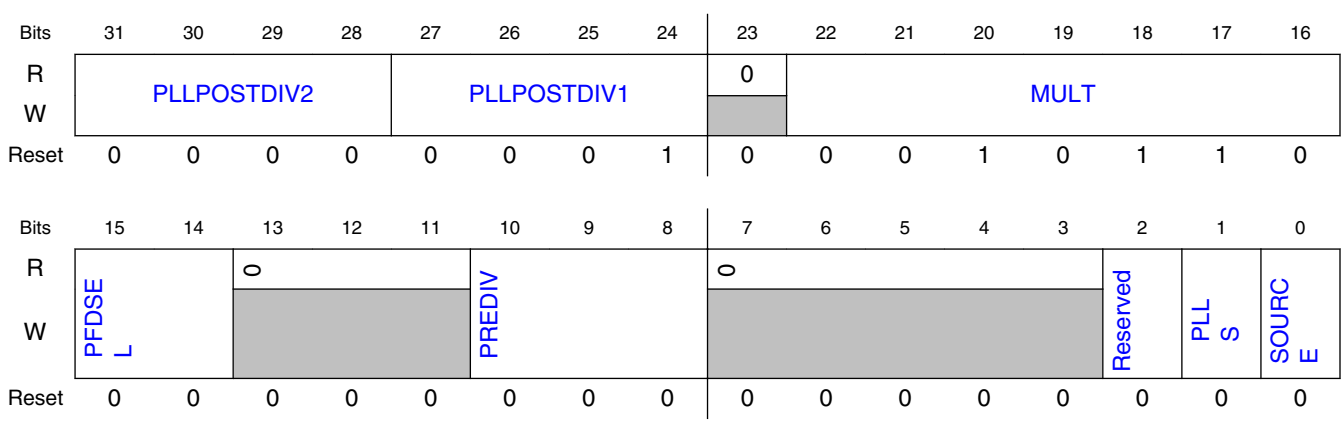
26.4.2.23.1 Offset

Register	Offset
APLLCFG	508h

26.4.2.23.2 Function

The APLLCFG register cannot be changed when the Auxiliary PLL is enabled. When the Auxiliary PLL is enabled, writes to this register are ignored, and there is no transfer error.

26.4.2.23.3 Diagram



26.4.2.23.4 Fields

Field	Function
31-28 PLLPOSTDIV2	Auxiliary PLL Post Clock Divide2 Ratio

Table continues on the next page...

Field	Function
	Auxiliary PLL 2nd stage post divider. Post clock dividers apply to APLL clock (non-PFD) only. NOTE: Correct divider ratio must be set prior to enabling of PLL. Changes to divider will be ignored if Auxiliary PLL is enabled. 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
27-24 PLLPOSTDIV1	Auxiliary PLL Post Clock Divide1 Ratio Auxiliary PLL 1st stage post divider. Post clock dividers apply to APLL clock (non-PFD) only. NOTE: Correct divider ratio must be set prior to enabling of PLL. Changes to divider will be ignored if Auxiliary PLL is enabled. 0000b - Divide-by-1 0001b - Divide-by-2 0010b - Divide-by-3 0011b - Divide-by-4 0100b - Divide-by-5 0101b - Divide-by-6 0110b - Divide-by-7 0111b - Divide-by-8 1000b - Divide-by-9 1001b - Divide-by-10 1010b - Divide-by-11 1011b - Divide-by-12 1100b - Divide-by-13 1101b - Divide-by-14 1110b - Divide-by-15 1111b - Divide-by-16
23 —	Reserved
22-16 MULT	Auxiliary PLL Multiplier Multiplier for the Auxiliary PLL. The MULT bits establish the multiplication factor applied to the PLL reference clock frequency. • PLL Output Frequency = Divided Reference Frequency * MULT • Valid Div Input Reference frequency are (in MHz): 24 (default), 32, 30, 26, 19.2 and 16 • Valid MULT values are 33, 27, 22, 20, 17, 16
15-14 PFDSEL	PFD Clock Select Selects which PFD output clock will be used as a SCG system reference clock or as a SCG DDR reference clock. 00b - PFD0 output clock selected. 01b - PFD1 output clock selected. 10b - PFD2 output clock selected. 11b - PFD3 output clock selected.

Table continues on the next page...

Field	Function
13-11 —	Reserved
10-8 PREDIV	PLL Reference Clock Divider Selects the amount to divide down the reference clock for the Auxiliary PLL. The resulting frequency must be 24 MHz(default),32 MHz, 30 MHz, 26 MHz, 19.2 MHz and 16 MHz.
7-3 —	Reserved
2 —	Reserved This field is reserved. Software should write 0 to this bit to maintain compatibility.
1 PLLS	PLL Select Selects the PFD or APLL as the PLL output clock used to generate system clocks. The PFD output clock used if PLLS=1 is determined by the PFDSEL bits. 0b - APLL clock selected. 1b - APLL PFD output clock selected
0 SOURCE	Clock Source Configures the input clock source for the Auxiliary PLL. Valid reference frequency must be 24 MHz, 32 MHz, 30 MHz,26 MHz, 19.2 MHz and 16 MHz. 0b - System OSC 1b - Reserved

Table 26-6. Auxiliary PLL Reference Divide Factor

PREDIV	Divide Factor
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

26.4.2.24 Auxiliary PLL PFD Register (APLLPFD)

26.4.2.24.1 Offset

Register	Offset
APLLPFD	50Ch

26.4.2.24.2 Function

This register defines the control bits for the PFD3-PFD0 clocks derived from the Auxiliary PLL.

When changing PFD values, the following is recommended:

1. The PFDx clock is gated first by writing a value of 1 to PFDx_CLKGATE register
2. Program the new PFD value
3. Write a value of 0 to PFDx_CLKGATE to ungate the PFDx
4. Follow by polling the PFDx_VALID flag to set and allow the PFDx clock to run.

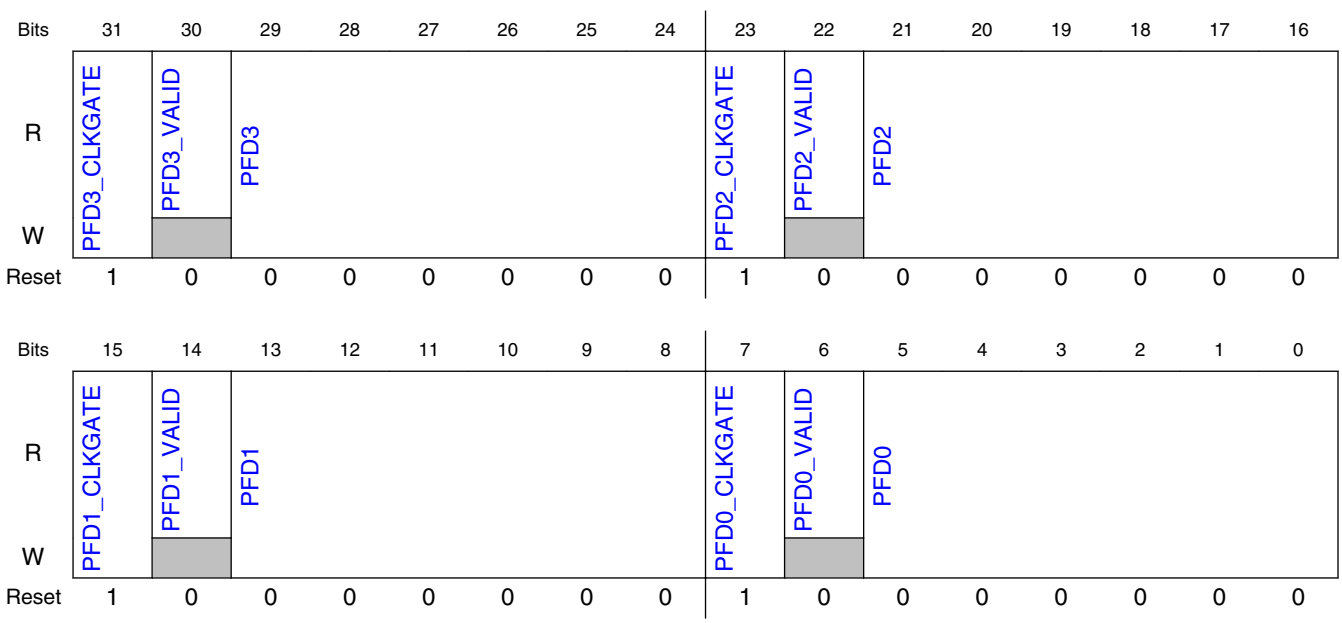
NOTE

It is recommended that PFD settings are kept between 12-35 for all PFDs.

APLL Frequency = Fref * (MULT + NUM/DENOM)

PFD Clock Frequency = PLL output frequency * 18/frac value

26.4.2.24.3 Diagram



26.4.2.24.4 Fields

Field	Function
31 PFD3_CLKGATE	PFD3_CLKGATE PFD3 Clock Gate. 0b - PFD3 clock is not gated.

Table continues on the next page...

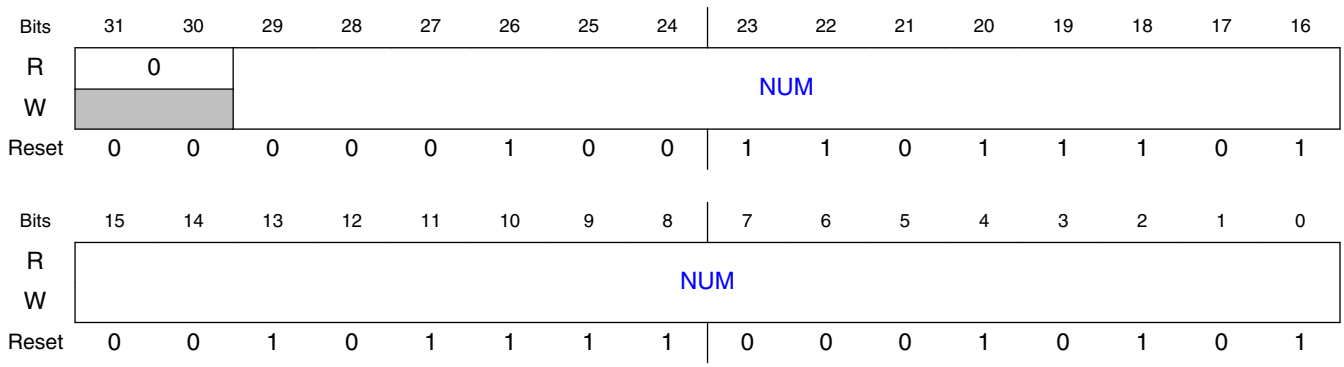
Field	Function
	1b - PFD3 clock is gated.
30 PFD3_VALID	PFD3_VALID Indicates when PFD3 clock is valid. Will clear when PFD3 is written with a new value and will set after PFD3 clock becomes stable.
29-24 PFD3	PLL Fractional Divider 3 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
23 PFD2_CLKGATE	PFD2_CLKGATE PFD2 Clock Gate. 0b - PFD2 clock is not gated. 1b - PFD2 clock is gated.
22 PFD2_VALID	PFD2_VALID Indicates when PFD2 clock is valid. Will clear when PFD2 is written with a new value and will set after PFD2 clock becomes stable.
21-16 PFD2	PLL Fractional Divider 2 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
15 PFD1_CLKGATE	PFD1_CLKGATE PFD1 Clock Gate. 0b - PFD1 clock is not gated. 1b - PFD1 clock is gated.
14 PFD1_VALID	PFD1_VALID Indicates when PFD1 clock is valid. Will clear when PFD1 is written with a new value and will set after PFD1 clock becomes stable.
13-8 PFD1	PLL Fractional Divider 1 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
7 PFD0_CLKGATE	PFD0_CLKGATE PFD0 Clock Gate. 0b - PFD0 clock is not gated. 1b - PFD0 clock is gated.
6 PFD0_VALID	PFD0_VALID Indicates when PFD0 clock is valid. Will clear when PFD0 is written with a new value and will set after PFD0 clock becomes stable.
5-0 PFD0	PLL Fractional Divider 0 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac

26.4.2.25 Auxiliary PLL Numerator Register (APLLNUM)

26.4.2.25.1 Offset

Register	Offset
APLLNUM	510h

26.4.2.25.2 Diagram



26.4.2.25.3 Fields

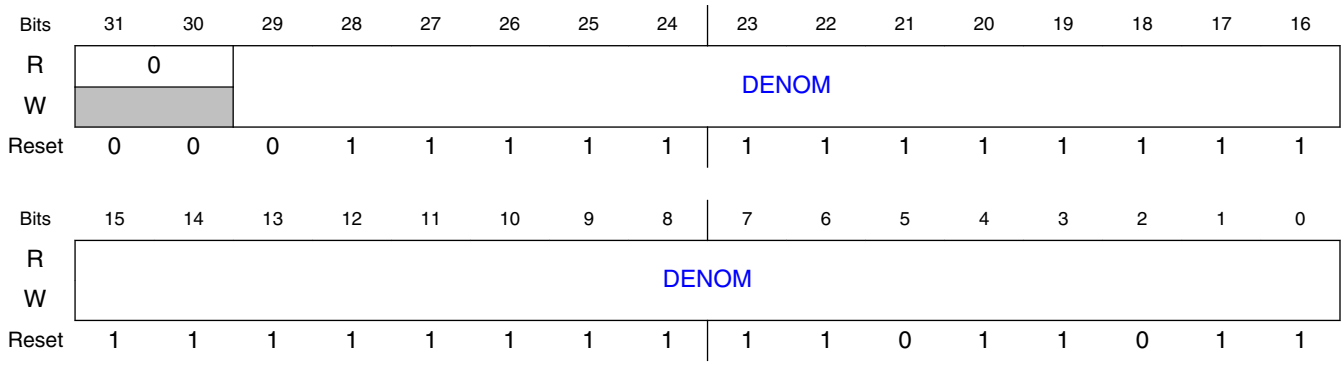
Field	Function
31-30 —	Reserved
29-0 NUM	30-bit numerator of the Auxiliary PLL Fractional-Loop divider NOTE: The value of numerator must always be configured to be less than the value of the denominator.

26.4.2.26 Auxiliary PLL Denominator Register (APLLDENOM)

26.4.2.26.1 Offset

Register	Offset
APLLDENOM	514h

26.4.2.26.2 Diagram



26.4.2.26.3 Fields

Field	Function
31-30 —	Reserved
29-0 DENOM	30-bit denominator of the Auxiliary PLL Fractional-Loop divider The value of numerator must always be configured to be less than the value of the denominator. **NOTE:** The DENOM value must not be zero, because that would cause division by zero, which is undefined in mathematics.

26.4.2.27 Auxiliary PLL Spread Spectrum Register (APLLSS)

26.4.2.27.1 Offset

Register	Offset
APLLSS	518h

26.4.2.27.2 Function

This register controls the APLL spread spectrum modulation. Spread spectrum modulation is used to reduce EMI.

26.4.2.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STOP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ENABLE	STEP														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

26.4.2.27.4 Fields

Field	Function
31-16 STOP	STOP and STEP together control the modulation depth (maximum frequency change) and modulation depth. Modulation Depth = (STOP/DENOM)*Fref where DENOM is the DENOM field value in DENOM register. Modulation Frequency = (STEP/(2*STOP))*Fref, where Fref = 24Mhz.
15 ENABLE	Enables the spread spectrum modulation. 0b - Spectrum modulation is disabled 1b - Spectrum modulation is enabled
14-0 STEP	STOP and STEP together control the modulation depth (maximum frequency change) and modulation frequency. Modulation Depth = (STOP/DENOM)*Fref where DENOM is the DENOM field value in DENOM register. Modulation Frequency = (STEP/(2*STOP))*Fref, where Fref = 24Mhz.

26.4.2.28 Auxiliary PLL LOCK Configuration Register (APLLLOCK_CNFG)

26.4.2.28.1 Offset

Register	Offset
APLLLOCK_CNFG	5F8h

26.4.2.28.2 Function

NOTE

The reset value for this register is chip-specific. See chip-specific section for more information.

26.4.2.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCK_TIME															
W																
Reset	0	0	0	0	0	1	0	0	0	1	1	0	1	0	0	0

26.4.2.28.4 Fields

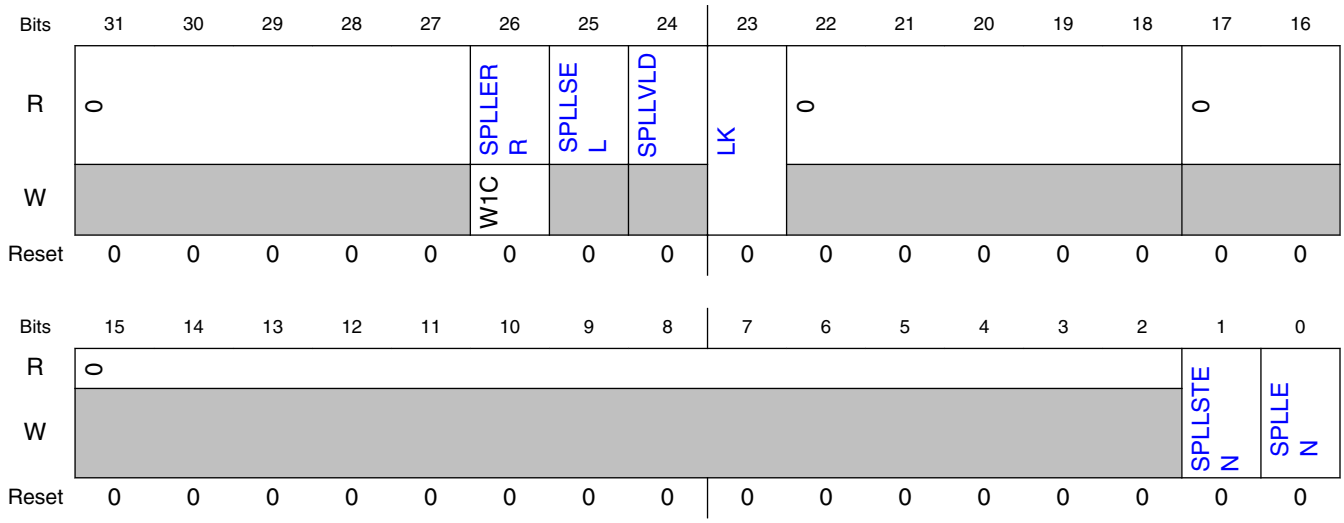
Field	Function
31-16 —	Reserved
15-0 LOCK_TIME	Configures the number of reference clocks to count before APLL is considered locked and valid.

26.4.2.29 System PLL Control Status Register (SPLLCSR)

26.4.2.29.1 Offset

Register	Offset
SPLLCSR	600h

26.4.2.29.2 Diagram



26.4.2.29.3 Fields

Field	Function
31-27 —	Reserved
26 SPLLERR	System PLL Clock Error This flag is reset on Chip POR only, software can also clear this flag by writing a logic one 0b - System PLL Clock Monitor is disabled or has not detected an error 1b - System PLL Clock Monitor is enabled and detected an error. System PLL Clock Error flag will not set when System OSC is selected as its source and SOSCERR has set.
25 SPLLSEL	System PLL Selected 0b - System PLL is not the system clock source 1b - System PLL is the system clock source
24 SPLLVLD	System PLL Valid Indicates when the SPLL clock is valid. Disabling the SPLL or a SOSC error when selected as the reference clock to the SPLL will cause the SPLLVLD to clear without setting SPLLERROR. **NOTE:** The System PLL Valid bit (SPLLVLD) should only be used to verify that the SPLL is locked after initialization. 0b - System PLL is not enabled or clock is not valid 1b - System PLL is enabled and output clock is valid
23 LK	Lock Register This bit field can be cleared/set at any time. 0b - Control Status Register can be written. 1b - Control Status Register cannot be written.
22-18 —	Reserved
17-16 —	Reserved

Table continues on the next page...

Field	Function
15-2 —	Reserved
1 SPLLSTEN	System PLL Stop Enable 0b - System PLL is disabled in Stop modes 1b - System PLL is enabled in Stop modes
0 SPLLEN	System PLL Enable **When configuring the SPLL, SPLLEN should be set/cleared first. Then the rest of the configuration process can be completed (disable PFDs, etc).** 0b - System PLL is disabled 1b - System PLL is enabled

26.4.2.30 System PLL Divide Register (SPLLDIV)

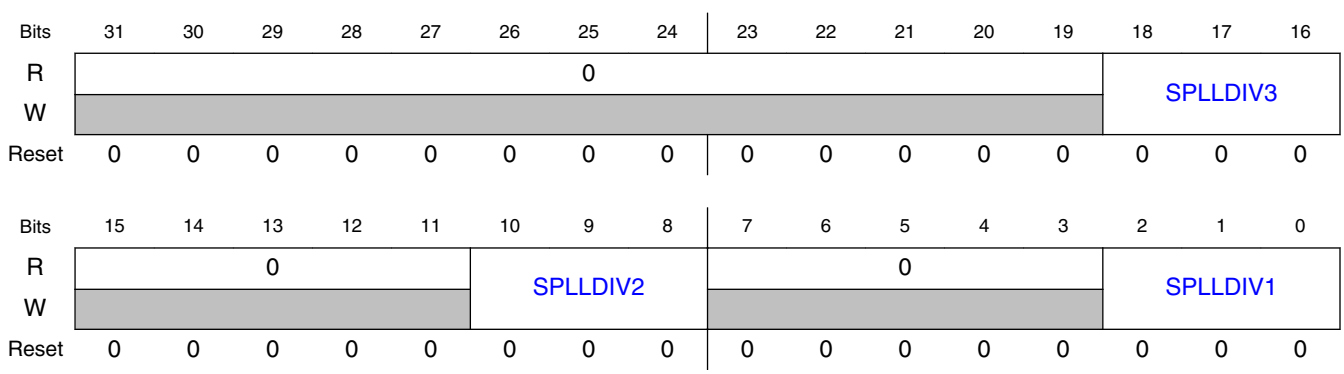
26.4.2.30.1 Offset

Register	Offset
SPLLDIV	604h

26.4.2.30.2 Function

Changes to SPLLDIV should be done when System PLL is disabled to prevent glitches to output divided clock.

26.4.2.30.3 Diagram



26.4.2.30.4 Fields

Field	Function
31-19 —	Reserved
18-16 SPLLDIV3	System PLL Clock Divide 3 Clock divider 3 for System PLL. Used by slow clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
15-11 —	Reserved
10-8 SPLLDIV2	System PLL Clock Divide 2 Clock divider 2 for System PLL. Used by bus clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64
7-3 —	Reserved
2-0 SPLLDIV1	System PLL Clock Divide 1 Clock divider 1 for System PLL. Used to generate the system clock source. Used by platform clock modules that need an asynchronous clock source. 000b - Clock disabled 001b - Divide by 1 010b - Divide by 2 011b - Divide by 4 100b - Divide by 8 101b - Divide by 16 110b - Divide by 32 111b - Divide by 64

26.4.2.31 System PLL Configuration Register (SPLLCFG)

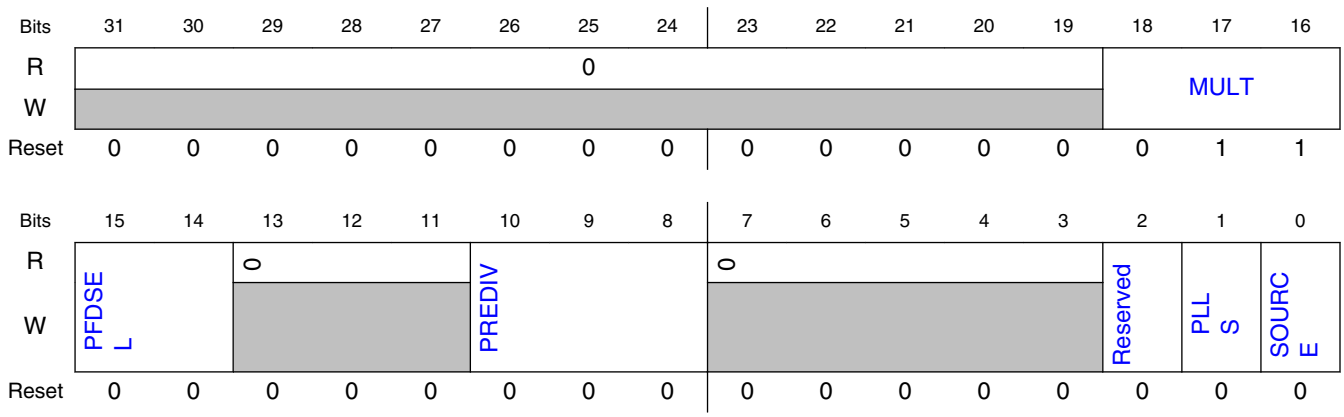
26.4.2.31.1 Offset

Register	Offset
SPLLCFG	608h

26.4.2.31.2 Function

The SPLLCFG register cannot be changed when the System PLL is enabled. When the System PLL is enabled, writes to this register are ignored, and there is no transfer error.

26.4.2.31.3 Diagram



26.4.2.31.4 Fields

Field	Function
31-19 —	Reserved
18-16 MULT	System PLL Multiplier Multiplier for the System PLL. The MULT bits establish the multiplication factor applied to the PLL reference clock frequency. • PLL Output Frequency = Divided Reference Frequency * MULT • Valid Div Input Reference frequency are (in MHz): 24 (default), 32, 30, 19.2 and 16 • Valid MULT values are 15, 16, 20, 22, 25, or 30 000b - Reserved 001b - MULT = 15 010b - MULT = 16 011b - MULT = 20 100b - MULT = 22 101b - MULT = 25 110b - MULT = 30 111b - Reserved
15-14 PFDSEL	PFD Clock Select Selects which PFD output clock will be used as a SCG system reference clock or as a SCG DDR reference clock. 00b - PFD0 output clock selected. 01b - PFD1 output clock selected. 10b - PFD2 output clock selected. 11b - PFD3 output clock selected.

Table continues on the next page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>13-11
—</td><td>Reserved</td></tr>
<tr><td>10-8
PREDIV</td><td>PLL Reference Clock Divider
Selects the amount to divide down the reference clock for the System PLL. The resulting divided frequency one of the following valid reference frequencies: 24MHz(default), 32, 30, 19.2 and 16.
Table 26-7. System PLL Reference Divide Factor<table><tr><td>PREDIV</td><td>Divide Factor</td></tr><tr><td>000</td><td>1</td></tr><tr><td>001</td><td>2</td></tr><tr><td>010</td><td>3</td></tr><tr><td>011</td><td>4</td></tr><tr><td>100</td><td>5</td></tr><tr><td>101</td><td>6</td></tr><tr><td>110</td><td>7</td></tr><tr><td>111</td><td>8</td></tr></table></td></tr>
<tr><td>7-3
—</td><td>Reserved</td></tr>
<tr><td>2
—</td><td>Reserved
This field is reserved. Software should write 0 to this bit to maintain compatibility.</td></tr>
<tr><td>1
PLLS</td><td>PLL Select
Selects the PLL or PFD as the SPLL output used to generate system clocks.
0b - SPLL output clocks selected
1b - SPLL PFD output clock selected.</td></tr>
<tr><td>0
SOURCE</td><td>Clock Source
Configures the input clock source for the System PLL.
0b - System OSC (SOSC)
1b - Reserved</td></tr>
</table>

26.4.2.32 System PLL PFD Register (SPLLPFD)

26.4.2.32.1 Offset

Register	Offset
SPLLPFD	60Ch

26.4.2.32.2 Function

This register defines the control bits for the PFD3-PFD0 clocks derived from the System PLL.

When changing PFD values, the following is recommended:

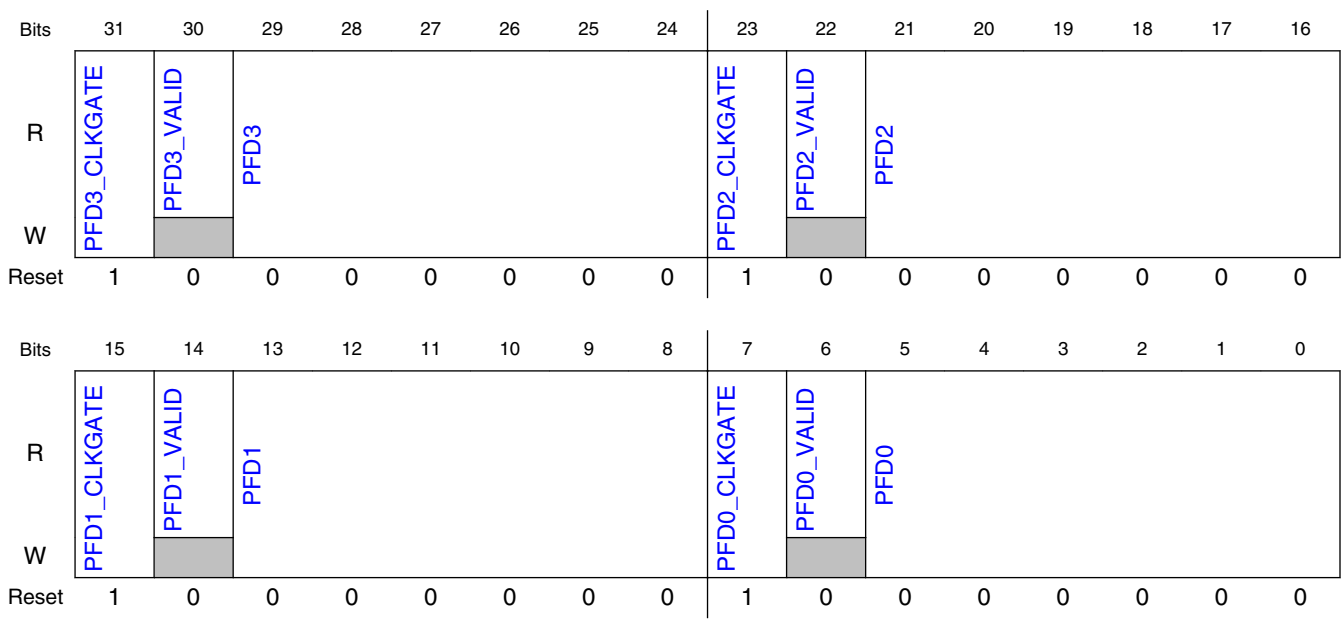
1. The PFDx clock is gated first by writing a value of 1 to PFDx_CLKGATE register
2. Program the new PFD value
3. Write a value of 0 to PFDx_CLKGATE to ungate the PFDx
4. Follow by polling the PFDx_VALID flag to set and allow the PFDx clock to run.

NOTE: It is recommended that PFD settings are kept between 12-35 for all PFDs.

SPLL Frequency = Fref * (MULT)

PFD Clock Frequency = PLL output frequency * 18/frac value

26.4.2.32.3 Diagram



26.4.2.32.4 Fields

Field	Function
31 PFD3_CLKGATE	PFD3_CLKGATE PFD3 Clock Gate. 0b - PFD3 clock is not gated. 1b - PFD3 clock is gated.
30	PFD3_VALID

Table continues on the next page...

Field	Function
PFD3_VALID	Indicates when PFD3 clock is valid. Will clear when PFD3 is written with a new value and will set after PFD3 clock becomes stable.
29-24 PFD3	PLL Fractional Divider 3 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
23 PFD2_CLKGATE	PFD2_CLKGATE PFD2 Clock Gate. 0b - PFD2 clock is not gated. 1b - PFD2 clock is gated.
22 PFD2_VALID	PFD2_VALID Indicates when PFD2 clock is valid. Will clear when PFD2 is written with a new value and will set after PFD2 clock becomes stable.
21-16 PFD2	PLL Fractional Divider 2 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
15 PFD1_CLKGATE	PFD1_CLKGATE PFD1 Clock Gate. 0b - PFD1 clock is not gated. 1b - PFD1 clock is gated.
14 PFD1_VALID	PFD1_VALID Indicates when PFD1 clock is valid. Will clear when PFD1 is written with a new value and will set after PFD1 clock becomes stable.
13-8 PFD1	PLL Fractional Divider 5 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac
7 PFD0_CLKGATE	PFD0_CLKGATE PFD0 Clock Gate. 0b - PFD0 clock is not gated. 1b - PFD0 clock is gated.
6 PFD0_VALID	PFD0_VALID Indicates when PFD0 clock is valid. Will clear when PFD0 is written with a new value and will set after PFD0 clock becomes stable.
5-0 PFD0	PLL Fractional Divider 0 Controls the fractional divide value. Valid pfd values are 12-35. PFD output frequency = PLL output frequency * 18/pfd0_frac

26.4.2.33 System PLL LOCK Configuration Register (SPLLLOCK_CNFG)

26.4.2.33.1 Offset

Register	Offset
SPLLLOCK_CNFG	6F8h

26.4.2.33.2 Function

NOTE

The reset value for this register is chip-specific. See chip-specific section for more information.

26.4.2.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCK_TIME															
W																
Reset	0	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0

26.4.2.33.4 Fields

Field	Function
31-16 —	Reserved
15-0 LOCK_TIME	Configures the number of reference clocks to count before SPLL is considered locked and valid.

Chapter 27
Power Management

27.1 Overview

This device implements a simplified power architecture to minimize the PMIC requirements. The intention is to use inexpensive external components. There are two completely separate main power domains: one for the Cortex-M4 power domain and the other for the Cortex-A7 power domain. Either domain can be completely shut off by removing external voltage or by turning off the internal voltage supply. In addition, there are other power islands that maintain power on for specific logics in very low power modes. Several low power techniques are implemented on this device to enable power efficient applications.

NOTE

References in this chapter to “Core 0” or “Processor A” correspond to the Cortex M4 core. References in this chapter to “Core 1” or “Processor B” correspond to the Cortex A7 core.

27.2 Low power techniques

This device reduces power consumption and thermal dissipation by applying the following low power techniques:

- Multiple power domains and low power modes allow flexible power optimization for power-conscious applications
- Voltage and frequency scaling in dynamic operating modes
- Software-controlled clock gating for cores and peripherals
- Efficient on-chip LDO regulators and power management control

27.3 Low power use case through PSTOP3 mode

This section provides a very low-power use case where power critical applications may control and manage data in Real-Time Domain (RTD) while requiring that part of the Application Domain (APD) to be enabled to access its peripherals. An example of such application would be to use FlexBus at APD domain to interface with an external PSRAM while M4 core does all the data management by copying data from PSRAM to M4's memories. In order to accomplish that and save power, the device is required to enter Partial STOP mode #3 (PSTOP3). When the device is in this mode and if none of the A7 NIC0 peripherals such as GPU 2D, GPU 3D, LCDIF, and MMDC, are enabled, A7 NIC0 sub-system is turned off by gating off its clock. Extra power saving is further obtained by selecting M4 Platform clock as the clock source for the A7 NICs' sub-system. This clock selection is done by writing to the SCG1_NICCCR[NICCS] bitfield. Furthermore, to prevent A7 core from waking-up during an ongoing cross-domain transaction, such as due to an interrupt from the peripheral, wake-up sources to WKPU unit shall be disabled by writing to the WKPU_WAKEUP_ENABLE register in the SIM module. The following procedure is used to enter PSTOP3 mode:

1. Ensure that there are no pending interrupts to the CA7 domain.
2. Ensure that peripherals connected to the NIC0 sub-system are disabled via PCC[1]
3. Configure WKPU channels via WKPU_WAKEUP_ENABLE register in the SIM module to disable all wake-up sources, except channel assigned to MU_B NMI[2]
4. Configure the MU_B NMI handler in the GIC unit
5. Configure the SCG1_NICCCR[NICCS] bitfield to select FIRC
6. Disable A7 PLLs
7. Configure the SCG1_NICCCR[NICCS] field to select M4 Platform clock. Ensure that clock source is ready for use.[3]
8. Configure A7 peripheral(s) in NIC1 sub-system for cross-domain transaction
9. Switch the A7 domain into PSTOP3 power mode

27.4 VLLS mode

The following points must be considered about entering/exiting the VLLS mode.

- Before entering VLLS mode, all interrupts belonging to peripherals powered by the same supply rail turned off by this mode must be disabled. If any of these interrupts wake up the domain when it has logically entered VLLS mode but has not effectively

1. NIC0's peripherals may still be used at the cost of increasing power. If any of the NIC0 peripherals are enabled via PCC, NIC0 sub-system is totally functional.
2. By keeping MU_B NMI channel in WKPU enabled, M4 core has total control over A7 domain and can wake-up A7 via MU_ACR[NMI]
3. Optional step in case a clock source different from FIRC is needed.

powered off yet, the associated interrupt flag will be reset (during the VLLS wake-up procedure), preventing the identification of the wake-up source once the domain is back to RUN mode. This affects both the Real Time and the Application domains, but the former one is more susceptible, due to the (external) capacitor connected to its power rail.

- When entering VLLS mode, both the respective domain and the associated pins are automatically isolated.
- When exiting (or resuming) from VLLS mode, the resume flow may include a jump to a user-specified entry point, as described in Resume support in ROM and SIM DGO GPR used by ROM
- Once VLLS mode is exited, domain's isolation is automatically removed, while pins' isolation removal is user controlled. See Supported modes of operation for details.

NOTE

It is a safe procedure to set SIM's WKPU_WAKEUP_ENABLE[7] bit to 0 and disable M4 NVIC IRQ 69 (USB PHY) before entering VLLS mode or enabling A7 switch

27.5 Power management scheme

Following figure shows the base Power Management scheme for this device.

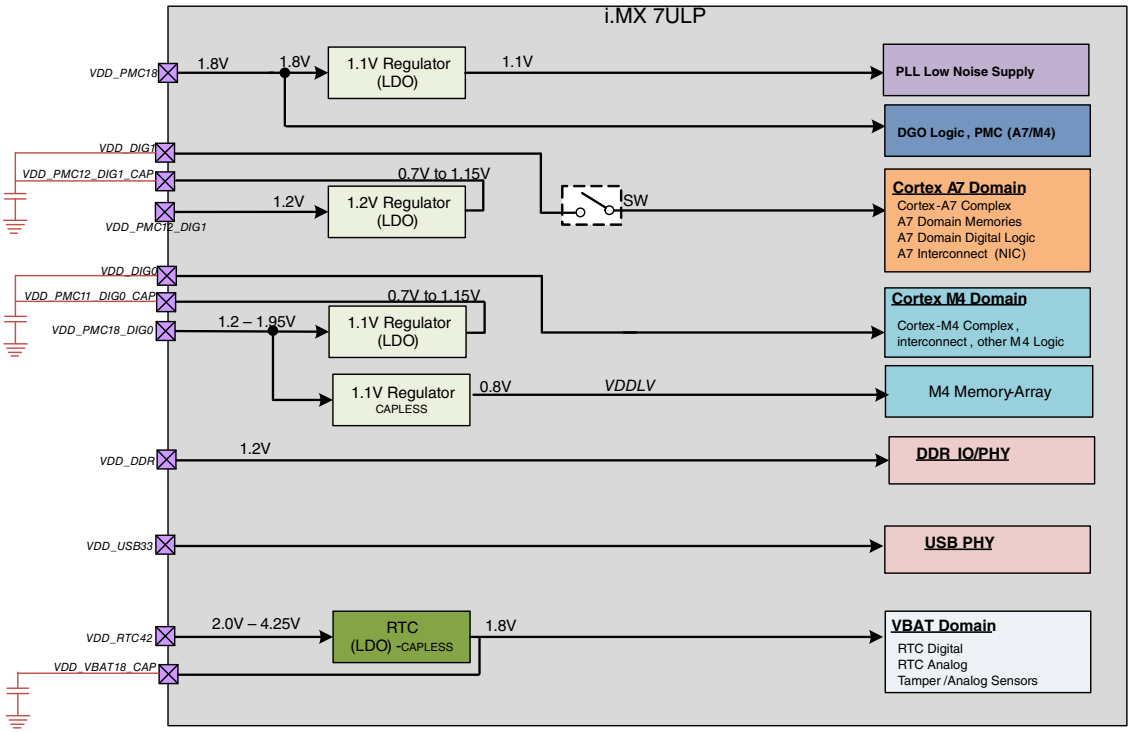


Figure 27-1. i.MX 7ULP power management scheme

27.6 VBAT mode

VBAT is a separate mode decouple from CM4/CA7 power domains. This is accessible to the user via external VBAT Pin on i.MX 7ULP.

Table 27-1. VBAT mode description in i.MX 7ULP

Mode	Description	Typical IDD	Recovery Method	Recovery Time
VBAT	Only RTC domain retained and powered through coin-cell/ Lithium-Ion battery	5 µA	Availability of VDD Supply/RTC Alarm	NA

27.7 Allowed power modes between multicore

Separate power modes on Arm CA7 and CM4 cores can result in several combinations, however not all are valid on i.MX 7ULP. The following table provides a summary of recommended power modes between the two cores.

In the table:

- NR: Not recommended
- NO: Not supported in hardware

Table 27-2. Allowed power modes in CM4 and CA7 core

CM4→	*HSRUN*	*RUN*	*VLPR*	*WAIT*	*VLPW*	*PSTOPx*	*STOP*	*VLPS*	*LLS*	*VLLS*
CA7↓[1], [2]										
HSRUN	YES	YES	YES[3]	YES	YES	YES	YES	YES	YES[4]	NO[5]
RUN	YES	YES	YES	YES	YES	YES	YES	YES	YES[4]	NO[5]
VLPR	YES	YES	YES	YES	YES	YES	YES	YES	NO[5], [6]	NO[5], [6]
PSTOPx	YES	YES	YES[4]	YES	YES[7]	YES	YES[7]	YES[7]	YES[7], [4]	NO[5]
STOP	YES	YES	YES	YES	YES	YES	YES	YES	YES	NO[5]
VLPS	YES	YES	YES	YES	YES	YES	YES	YES	YES	NO[5]
LLS	YES	YES	YES	YES	YES	YES	YES	YES	YES	NO[5]
VLLS	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES

1. CA7 only supports static (w/o clock) reverse body bias.
2. APD may have its system clocks (i.e. core, plat0/1, bus, ...) gated, even if in a Run mode (HSRUN, RUN, VLPR), should the dependency between APD and RTD's primary clock sources (for example, SIRC, FIRC, SOSC, ...) is not observed.
3. Main benefit of CM4 [VLPR] is to provide lowest power run mode by using RBB while main advantage for CA7 [HSRUN] is to provide higher performance by using FBB. However using this combination will provide no power advantage so benefits with CM4 [VLPR] will be in the noise when CA7 is in HSRUN.
4. For cases where CM4 is in LLS mode, the external oscillator being the only clock source available to CA7, register field SCG0[SOSCCSR] should be programmed to remain enabled.
5. i.MX 7ULP does not support any isolation from the CM4 to CA7 domain so any scenarios where CM4 is kept in VLLS mode while CA7 is in higher power modes cannot be supported. There are no particular power advantages supporting these scenarios.
6. SIRC/FIRC clocks not available for CA7 when CM4 is in LLS/VLLS mode. No significant power advantage to have CA7 running while CM4 is in LLS/VLLS mode.
7. CA7 can choose to keep peripherals running in full operation in CA7 [PSTOPx] mode, but will significantly increase power consumption (since PLLs will be ON with the external oscillator as the reference clock) while CM4 is in low power mode (VLPR/VLPW/STOP/ VLPS mode), thus making this mode not useful to the application.

27.8 Biasing options

I.MX 7ULP supports both Forward and Reverse bias; however only variants of options would be available based on power modes.

The following table shows applicable power modes for both the biasing options.

Table 27-3. Biasing options

Biasing Options[1]	Bias Voltage Range (Typical)	Applicable Power modes
Forward Body Bias (FBB)	50 mV to 350 mV, with 50 mV voltage steps	HSRUN Mode
Reverse Body Bias(RBB)	500 mV to 1.3 V, with 100 mV voltage steps	VLPR (CM4 domain only), VLPS and LLS modes

1. *All options for FBB/RBB are fully programmable*

The following figure shows the basic operation of NMOS/PMOS transistor in C28SOI. VBBp/VBBn corresponds to the bias voltage.

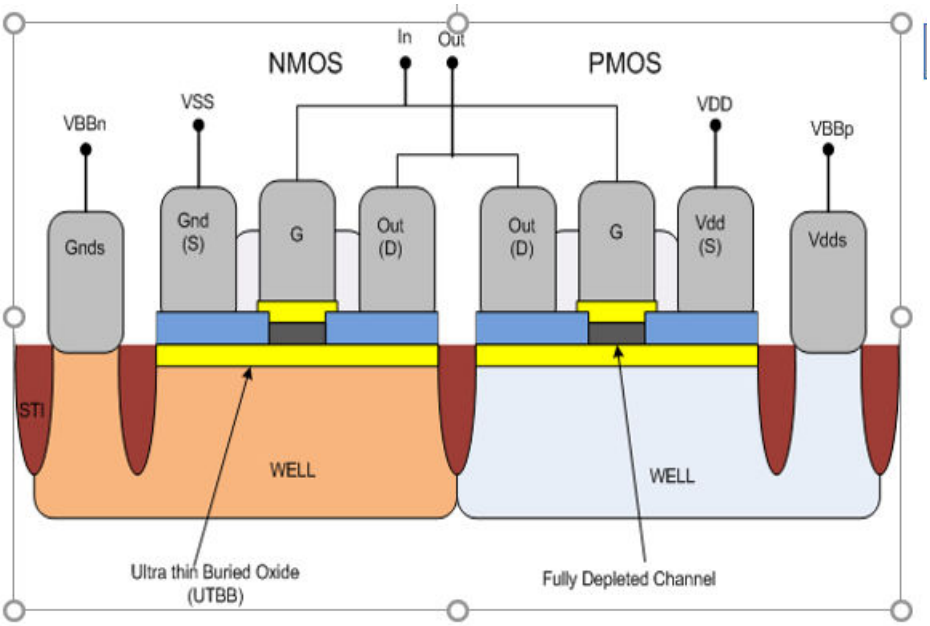


Figure 27-2. NMOS/PMOS transistor operation

The following table shows the valid RBB/FBB combinations for i.MX 7ULP

Table 27-4. Valid RBB/FBB combinations for i.MX 7ULP

A7 Domain	M4 Domain	Valid i.MX 7ULP use-case	Applicable A7/M4 Power mode
No Biasing	No Biasing	YES	No Restriction
FBB	No Biasing	YES	• A7 (HSRUN Mode) • M4 (RUN Mode)
FBB	FBB	YES	• A7 (HSRUN Mode) • M4 (HSRUN Mode)
FBB	RBB	YES	N/A
Power Gated	FBB	YES	M4 (HSRUN Mode)
Power Gated	RBB	YES	M4 (VLPR, VLPS, LLS modes)
RBB	No Biasing	YES	• A7 (VLPS, LLS modes) • M4 (RUN mode)
RBB	FBB	YES	N/A
RBB	RBB	YES	• A7 (VLPS, LLS modes) • M4 (VLPR, VLPS, LLS modes)

27.9 Power Management Controller (PMC)

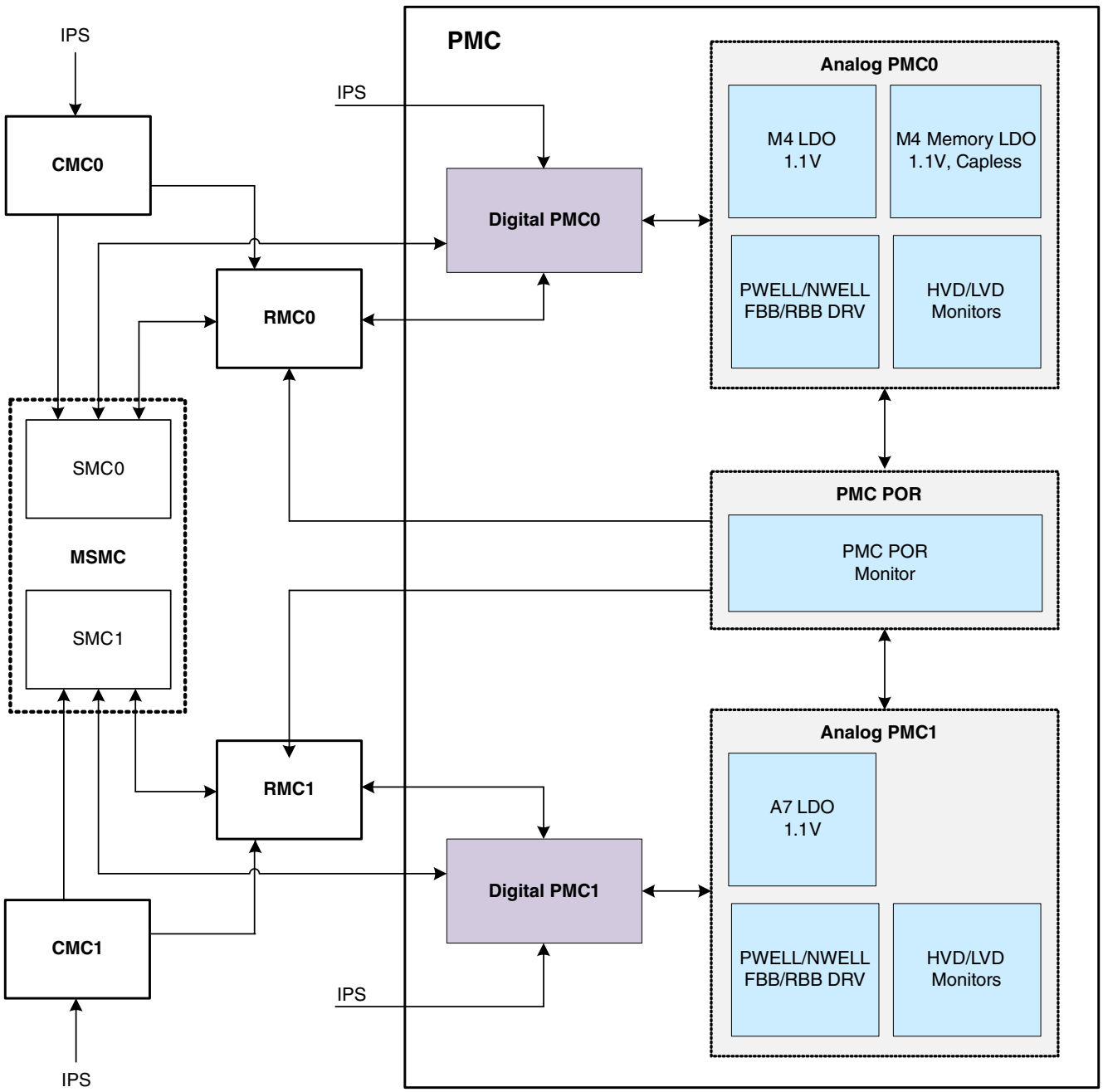


Figure 27-3. PMC System and Interface

PMC comprises Digital PMC modules and Analog PMC components. The Digital PMC module allows user software to control power modes of the chip and to optimize power consumption for the level of functionality needed.

There are two instances of Digital PMC on this device: Digital PMC0 controlling M4 and M4 Memory power domains and Digital PMC1 controlling A7 power domain. Digital PMCs work in conjunction with CMCs, MSMC and RMCs to facilitate the transition of operating modes and power modes of the device. A part of the Digital PMC module resides in the "always-on" DGO power domain.

The Analog PMC consists of voltage and current references, core logic supply regulators, memory supply regulators, back bias and forward bias regulators, monitors and power switches. The Analog PMC is made of Analog PMC0, Analog PMC1, and PMC POR circuit. Analog PMC0 manages supply to M4 domain and M4 Memory domain, and Analog PMC1 manages supply to the entire A7 domain.

The Digital PMC and Analog PMC supply is constantly monitored by the PMC POR. This circuit is always enabled. When the PMC POR monitor detects a supply less than 1.8 V it will generate a power-on reset to the CMCs.

27.10 Power supply use cases

The i.MX 7ULP microcontroller can be supplied by a simple PMIC, which has switched and linear regulators. This device can also be supplied by a combination of linear regulators for cost optimization purpose. The following figures show some possible power supply options for i.MX 7ULP.

Use case 1: Companion PMIC:

- An external PMIC that has at least 1 switched regulator and 1.2 V, 1.8 V, and 3.3 V linear regulators.
- M4 domain is always powered by an internal regulator which gets its supply from the PMIC 1.8V rail.
- A7 domain is supplied directly from the PMIC switched regulator.
- GPIO ports are on either 1.8 V or 3.3 V supply rails.
- DDR IO is on 1.2 V supply rail.

Use case 2: A combination of linear regulators shared between M4 and A7 domains::

- Common 1.2 V, 1.8 V, and 3.3 V regulators shared between A7 and M4 power domains
- M4 domain is always powered by an internal regulator which gets its supply from the 1.8 V rail.
- A7 is powered by an internal regulator which gets its supply from the 1.2V rail.

- GPIO ports are on either 1.8 V or 3.3 V supply rails.
- DDR IO is on 1.2 V supply rail.

Use case 3: Independent supply sources for M4 and A7 domains:

- Separate sets of linear regulators for M4 and A7 power domains
- M4 domain is always powered by an internal regulator which gets its supply from the 1.8V rail dedicated for M4 side.
- M4 GPIO ports are on either 1.8V or 3.3V supply rails dedicated for M4 side.
- A7 is powered by an internal regulator which gets its supply from the 1.2V rail.
- A7 GPIO ports are on either 1.8V or 3.3V supply rails dedicated for A7 side.
- DDR IO is on 1.2V supply rail.

27.11 Power modes

27.11.1 A7 domain power

The A7 domain has multiple power modes. The following table describes the state of A7 domain in those modes.

Table 27-5. A7 power modes

A7 Power Mode	Description	Recovery Method	Recovery Time
HSRUN	• All logic is functional in this mode. • Bus clock and peripherals functional. • Allows FBB (optional) • Allows Dynamic Voltage Scaling(DVS)	N/A	N/A
RUN	• All logic is functional in this mode. • Bus clock and peripherals functional. • FBB/RBB not allowed. • Allows Dynamic Voltage Scaling(DVS)	N/A	N/A
VLPR	• All logic is functional in this mode. • Max Frequency restricted to FIRC (48 MHz). PLLs disabled. • Option to disable LVD/HVD	N/A	N/A

Table continues on the next page...

Table 27-5. A7 power modes (continued)

A7 Power Mode	Description	Recovery Method	Recovery Time
	• DDR in self-refresh mode. DDR self-refresh must be done by software by writing to MMDC_MAPSR[LPMD] bit • RBB not allowed		
WAIT[1], [2]	• Allows Peripherals to function while keeping core in sleep (clock-gated). • A7 processor in Wait-for-Interrupt (WFI) state.	Interrupt[3]/ Reset, [4]	0 ns
STOP[5]/VLPS	• I.MX 7ULP is in static state with all registers retained with maintaining LVD protection. • RBB only allowed in VLPS mode. • FIRCCSR[FIRCLPEN] in the SCG module, keeps FIRC enabled in VLPS mode. • LVDs could be turned off in VLPS mode.	Interrupt/ Reset[4]	7 µs(STOP) and 23 µs (VLPS with RBB)/21.5 µs (VLPS without RBB)[6]
LLS[7]	• A7 supply ON • RBB is allowed • LVD protection • IO supplies ON • A7 processor is in a wait-for-interrupt (WFI) state. The core clock is gated. • Bus and DMA clocks are gated • All peripheral clocks are gated. • SRAM contents are retained. • DDR can be in self-refresh	Interrupt/ Reset[4]	Wake up time: 41.5 µs (LLS with RBB)/40 µs (LLS without RBB)[8]
VLLS	• A7 domain fully power gated. • Wake-up only via MU_A (CM4 domain) or reset. • DDR can be in self-refresh	MU_A[9]/ Reset, [4]	60 µs
BAT	• A7 supply OFF • IO supplies OFF • Logic registers and RAMs are not retained • DDR can be in self-refresh		• M4 turns on A7 and A7 IO power supplies • PMIC power on sequence • Wake up time: NA

1. i.MX 7ULP supports additional variant of WAIT mode called VLPS mode that allows peripheral operation at reduced frequency. See MSMC or PMC for details.
2. See SLEEPDEEP mode
3. Interrupts from A7, M4 or external device via GPIO pin could act as a wake-up. GIC detects wake-up source for A7 processor.
4. A7 reset sources or M4 reset sources
5. i.MX 7ULP supports variation of STOP mode called Partial STOP [1/2/3] that allows additional options (enable/disable) to control system and bus clock. PSTOP1/2 mode recovery time is less than the one for STOP mode, while PSTOP3 recovery time is around the same as that for WAIT mode. See MSMC or PMC for details.
6. These values depend on LDO setting during low-power mode
7. VLPS and LLS modes work in a very similar way, except that LLS causes the memories to enter RETENTION mode and their supply to be reduced (supply reduction applies to M4 domain only). Besides, pins remain isolated while the domain is in LLS mode.

8. On CA7, LLS uses PMC0_CTRL[LDOOKDIS]=1
9. MU_ACR[BHR] or MU_ACR[NMI]

Table 27-6. A7 power mode detail

Power modes		HS RUN	RUN	VLPR	WAIT[1]	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[2]	VLLS	BAT
Modules	**Power State Power Domain**	**A7 supply ON, FBB optional, Allows DVS, IO supplies ON**	**A7 supply ON, Allows DVS, IO supplies ON**	**A7 supply ON, RBB OFF, IO supplies ON**	**A7 supply ON, IO supplies ON**	**A7 supply ON, IO supplies ON, sys/bus clks ON**	**A7 supply ON, IO supplies ON, sys clk OFF/bus clk ON**	**A7 supply ON, IO supplies ON, sys/bus clks OFF**	**A7 supply ON, LVD protection, IO supplies ON**	**A7 supply ON, RBB optional, LVD protection, IO supplies ON**	**A7 supply ON, LVD protection, IO supplies ON, RBB optional**	**A7 supply OFF, IO supplies ON (optional)**	**A7 supply OFF, IO supplies OFF**
SCG1	A7	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static [3]	Static	Static	Power Gated	Power Gated
PLL2	PLL LDO	Functional	Functional	Disabled	Functional	Functional	Functional	Functional	Disabled	Disabled	Disabled	Power Gated	Power Gated
PLL3	PLL LDO	Functional	Functional	Disabled	Functional	Functional	Functional	Functional	Disabled	Disabled	Disabled	Power Gated	Power Gated
PCC2	A7	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static	Power Gated	Power Gated
PCC3	A7	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static	Power Gated	Power Gated
Digital PMC1	A7/DGO	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Powered off
MSMC (SMC1)	DGO	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Powered off
WKPU	A7	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static	Power Gated	Power Gated
A7 Core	A7	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
A7 SCU	A7	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
L1 Cache	A7	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
L2 Cache	A7	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
SRAM0-1[4]	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Power Gated	Power Gated

Table continues on the next page...

Table 27-6. A7 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT[1]	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[2]	VLLS	BAT
Modules	**Power State Power Domain**	**A7 supply ON, FBB optional, Allows DVS, IO supplies ON**	**A7 supply ON, Allows DVS, IO supplies ON**	**A7 supply ON, RBB OFF, IO supplies ON**	**A7 supply ON, IO supplies ON**	**A7 supply ON, IO supplies ON, sys/bus clks ON**	**A7 supply ON, IO supplies ON, sys clk OFF/bus clk ON**	**A7 supply ON, IO supplies ON, sys/bus clks OFF**	**A7 supply ON, LVD protection, IO supplies ON**	**A7 supply ON, RBB optional, LVD protection, IO supplies ON**	**A7 supply ON, LVD protection, IO supplies ON, RBB optional**	**A7 supply OFF, IO supplies ON (optional)**	**A7 supply OFF, IO supplies OFF**
NIC0-1[4]	A7	Functional	Functional	Functional	Functional	Functional[5]	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Power Gated	Power Gated
AHB-PBridge-1[4]	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Power Gated	Power Gated
ROM1	A7	Functional	Functional	Functional	Functional	Static	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
DMA1[4]	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Power Gated	Power Gated
MMDC	A7	Functional	Functional	Functional	Functional	Functional	Static	Static	Clock Gated	Clock Gated	Async Operation	Power Gated	Power Gated
FlexBus	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
WDOG1	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Power Gated	Power Gated	Power Gated
WDOG2 (security)	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Power Gated	Power Gated	Power Gated
SEMA42_1	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
IOMXUC1	A7	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Static	Static	Static	Power Gated	Power Gated
PCTL_C-F[6]	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Static	Static	Static	Power Gated	Power Gated
IOMUXC_DDR	A7	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Static	Static	Static	Power Gated	Power Gated

Table continues on the next page...

Table 27-6. A7 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT[1]	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[2]	VLLS	BAT
Modules	**Power State Power Domain**	**A7 supply ON, FBB optional, Allows DVS, IO supplies ON**	**A7 supply ON, Allows DVS, IO supplies ON**	**A7 supply ON, RBB OFF, IO supplies ON**	**A7 supply ON, IO supplies ON**	**A7 supply ON, IO supplies ON, sys/bus clks ON**	**A7 supply ON, IO supplies ON, sys clk OFF/bus clk ON**	**A7 supply ON, IO supplies ON, sys/bus clks OFF**	**A7 supply ON, LVD protection, IO supplies ON**	**A7 supply ON, RBB optional, LVD protection, IO supplies ON**	**A7 supply ON, LVD protection, IO supplies ON, RBB optional**	**A7 supply OFF, IO supplies ON (optional)**	**A7 supply OFF, IO supplies OFF**
PORTC-FIO	VDD_PTC-F	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Static	Static	Power Gated	Power Gated	
DDR IO[7]	VDD DDR	Functional	Functional	Functional	Functional	Functional	Static	Static	Static	Static	Static	Power Gated	Power Gated
RGPIO2P1	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock gated	Power Gated	Power Gated
FlexIO1	A7	Functional	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Clock Gated	Power Gated	Power Gated
GPU-3D	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
GPU-2D	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
DSI	A7	Functional	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
LCDIF	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
VIU	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
uSDHC0	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
uSDHC1	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
USB-OTG1	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
USB0 PHY	USB/A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated

Table continues on the next page...

Table 27-6. A7 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT[1]	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[2]	VLLS	BAT
Modules	**Power State Power Domain**	**A7 supply ON, FBB optional, Allows DVS, IO supplies ON**	**A7 supply ON, Allows DVS, IO supplies ON**	**A7 supply ON, RBB OFF, IO supplies ON**	**A7 supply ON, IO supplies ON**	**A7 supply ON, IO supplies ON, sys/bus clks ON**	**A7 supply ON, IO supplies ON, sys clk OFF/bus clk ON**	**A7 supply ON, IO supplies ON, sys/bus clks OFF**	**A7 supply ON, LVD protection, IO supplies ON**	**A7 supply ON, RBB optional, LVD protection, IO supplies ON**	**A7 supply ON, LVD protection, IO supplies ON, RBB optional**	**A7 supply OFF, IO supplies ON (optional)**	**A7 supply OFF, IO supplies OFF**
MIPI DSI PHY	DSI/A7	Functional	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
USB-OTG2	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
LPI2C4-7	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
LPSPI2-3	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
LPUART4-7	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
LPTPM4-7	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
LPIT1	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
TRGMUX1	A7	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Static	Power Gated	Power Gated
CAAM	A7	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated

1. See SLEEPDEEP mode
2. VLPS and LLS modes work in a very similar way, except that LLS causes the memories to enter RETENTION mode and their supply to be reduced (supply reduction applies to M4 domain only). Besides, pins remain isolated while the domain is in LLS mode.
3. Static: clock(s) is(are) ON, but the module is not operating.
4. Here, Async Operation means the block supports the DMA transfers from/to IPs that are working in Assynchronous Operation state
5. NIC0 is clock gated if none of its peripherals (GPU 2D, GPU3D, LCDIF, and MMDC) is enabled.
6. PCTLs are capable of detection external interrupts
7. DDR_CKE will be pulled down at the board level if DDR is in self-refresh mode

27.11.2 M4 domain power

The M4 domain has multiple power modes. The following table describes the state of M4 domain in those modes.

Table 27-7. M4 power modes

M4 Power Mode	Description	Recovery Method	Recovery Time
HSRUN	• All logic is functional in this mode. • Bus clock and peripherals functional. • Allows FBB (optional) • Allows Dynamic Voltage Scaling(DVS)	N/A	N/A
RUN	• All logic is functional in this mode. • Bus clock and peripherals functional. • FBB/RBB not allowed. • Allows Dynamic Voltage Scaling (DVS)	N/A	N/A
VLPR	• All logic is functional in this mode. • Maximum frequency restricted to FIRC (48 MHz). PLL disabled • Allows RBB (Optional) • Allows to disable LVD/ HVD(optional)	N/A	N/A
WAIT	• Allows Peripherals to function while keeping core in sleep (clock-gated). • M4 processor in WFI state.	Interrupt[1]/Reset[2]	0 ns
STOP[3]/VLPS	• i.MX 7ULP is static state with all registers retained with maintaining LVD protection. • Peripheral optionally operational in STOP mode[4] • RBB only allowed in VLPS mode. • FIRC enabled in VLPS mode via SCG0_FIRCCSR register. • LVDs could be turned off in VLPS mode.	Interrupt[5] /Reset[2]	4 µs (STOP) and 11.5 µs (VLPS with RBB)/9 µs (VLPS without RBB)[6]
LLS [7]	• Static mode with no active transition. • CM4 in WFI mode with core clock gated • RBB allowed	Interrupt[5] /Reset[2]	62 µs (LLS with RBB)/58 µs (LLS without RBB)[8]

Table continues on the next page...

Table 27-7. M4 power modes (continued)

M4 Power Mode	Description	Recovery Method	Recovery Time
VLLS[9]	• M4 core supply OFF with majority of the logic power gated. • AWIC detects wake-up sources for M4 (via LLWU) • Selectable Memory retention (32K/64K/256K) • ADC, Comparators, LP Timers Optionally functional • RBB allowed (Optional) • DGO [10](aka Always ON) Logic Active	Wake-up Interrupt[11]/Reset[12]	375 µs

1. Interrupts from A7, M4 or external device via GPIO pin could act as a wake-up. NVIC detects wake-up source for M4.
2. M4 reset sources or A7 reset sources (when A7_TO_M4_RESET_EN fuse is blown).
3. i.MX 7ULP supports variation of STOP mode called Partial STOP [1/2/3] that allows additional options (enable/disable) to control system and bus clock. PSTOP1/2 mode recovery time is less than the one for STOP mode, while PSTOP3 recovery time is around the same as that for WAIT mode See MSMC or PMC chapter for details.
4. Peripherals capable of asynchronous operation during STOP (at full frequency) and VLPS (at limited frequency). This includes low power peripherals (LPUART, LPSPI, and LPI2C) operational on SIRC/FIRC/SOSC clock in STOP/VLPS mode.
5. Wakeup via Interrupts from M4 peripherals or external GPIO pins. NVIC disabled, AWIC detects the wake-up source.
6. This value depends on LDO setting during low-power mode
7. VLPS and LLS modes work in a very similar way, except that LLS causes the memories to enter RETENTION mode and their supply to be reduced (supply reduction applies to M4 domain only). Besides, pins remain isolated while the domain is in LLS mode.
8. On CM4, LLS uses LDO HP mode
9. See MSMC or PMC chapter for details.
10. Only Peripherals in DGO domain (CMPx, LPTMRx) are functional.
11. VLLS wake-up through LLWU and NMI pin (PTA9). SIM_DGO_GP11[NMI_VLLS_WAKEUP_EN] in the SIM module must be set prior to entering VLLS mode when using the NMI pin. VLLS exit forces a system reboot.
12. RESET0_b or RESET1_b (when both M4 and A7 domains are in VLLS mode, regardless of A7_TO_M4_RESET_EN fuse's state).

Table 27-8. M4 power mode detail

Power modes		HS RUN	RUN	VLPR	WAIT	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[1]	VLLS	BAT
Modules	**Power State Power Domain**	**M4 supply on M4 RAM supply on DGO supply on FBB optional Allows DVS IO supplies on**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, Allows DVS, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, RBB optional, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, IO supplies ON**	**M4 supplies ON, IO supplies ON, sys/bus clk ON**	**M4 supplies ON, IO supplies ON, sys clk OFF/bus clk ON**	**M4 supplies ON, IO supplies ON, sys/bus clks OFF**	**M4 supply ON, M4 array supply ON, DGO supply ON, LVD protection, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 supply ON, reduced M4 RAM supply, reduced DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 Supply OFF, M4 RAM Supply ON, DGO supply ON, IO Supplies ON**	**M4 supply OFF, M4 RAM supply OFF, DGO supply OFF, IO supplies OFF**
SCG0	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Limited Functional	Static [2]	Power Gated
PLL0	PLL LDO	Functional	Functional	Disabled	Functional	Functional	Functional	Functional	Disabled	Disabled	Disabled	Disabled	Power Gated
PLL1 (audio)	PLL LDO	Functional	Functional	Disabled	Functional	Functional	Functional	Functional	Disabled	Disabled	Disabled	Disabled	Power Gated
SYS OSC	M4	Functional	Functional	Disabled	Functional	Functional	Functional	Functional	Disabled	Disabled	Disabled	Disabled	Power Gated
FIRC	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Disabled	Disabled	Disabled	Power Gated
SIRC	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Disabled	Disabled	Disabled	Power Gated
PCC0	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static	Disabled	Power Gated
PCC1	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static	Disabled	Power Gated
CMC0	DGO	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Power Gated
RMC0	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static	Disabled	Power Gated

Table continues on the next page...

Table 27-8. M4 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[1]	VLLS	BAT
Modules	**Power State Power Domain**	**M4 supply on M4 RAM supply on DGO supply on FBB optional Allows DVS IO supplies on**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, Allows DVS, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, RBB optional, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, IO supplies ON**	**M4 supplies ON, IO supplies ON, sys/bus clk ON**	**M4 supplies ON, IO supplies ON, sys clk OFF/bus clk ON**	**M4 supplies ON, IO supplies ON, sys/bus clks OFF**	**M4 supply ON, M4 array supply ON, DGO supply ON, LVD protection, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 supply ON, reduced M4 RAM supply, reduced DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 Supply OFF, M4 RAM Supply ON, DGO supply ON, IO Supplies ON**	**M4 supply OFF, M4 RAM supply OFF, DGO supply OFF, IO supplies OFF**
Digital PMC0	M4/DGO	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Power Gated
PMC Core	PMC	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Power Gated
MSMC (SMC0)	DGO	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Power Gated
LLWU	DGO	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Power Gated
AWIC	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Static	Power Gated
SIM	M4	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Power Gated	Power Gated
OCOTP_CTRL	M4	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Static	Static	Static	Power Gated	Power Gated
Fusebox	M4	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Static	Static	Static	Power Gated	Power Gated
Boot Config Regs	M4	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Functional	Functional	Functional	Functional	Power Gated
M4 Core	M4	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated

Table continues on the next page...

Table 27-8. M4 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[1]	VLLS	BAT
Modules	**Power State Power Domain**	**M4 supply on M4 RAM supply on DGO supply on FBB optional Allows DVS IO supplies on**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, Allows DVS, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, RBB optional, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, IO supplies ON**	**M4 supplies ON, IO supplies ON, sys/bus clk ON**	**M4 supplies ON, IO supplies ON, sys clk OFF/bus clk ON**	**M4 supplies ON, IO supplies ON, sys/bus clks OFF**	**M4 supply ON, M4 array supply ON, DGO supply ON, LVD protection, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 supply ON, reduced M4 RAM supply, reduced DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 Supply OFF, M4 RAM Supply ON, DGO supply ON, IO Supplies ON**	**M4 supply OFF, M4 RAM supply OFF, DGO supply OFF, IO supplies OFF**
M4 L1 Cache	M4 Array	Functional	Functional	Functional	Static	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
M4 TCM[3]	M4 Array	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Clock Gated	Clock Gated	Clock Gated	Static	Power Gated
AXBS-Lite[3]	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Power Gated	Power Gated
AIPS0&1[3]	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Power Gated	Power Gated
ROM0	M4	Functional	Functional	Functional	Functional	Static	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
DMA0[3]	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Power Gated	Power Gated
QuadSPI	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Clock Gated	Power Gated	Power Gated
OTFAD	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Clock Gated	Power Gated	Power Gated
WDOG0	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Static	Static	Static	Power Gated	Power Gated
EWM	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Static	Static	Static	Power Gated	Power Gated

Table continues on the next page...

Table 27-8. M4 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[1]	VLLS	BAT
Modules	**Power State Power Domain**	**M4 supply on M4 RAM supply on DGO supply on FBB optional Allows DVS IO supplies on**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, Allows DVS, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, RBB optional, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, IO supplies ON**	**M4 supplies ON, IO supplies ON, sys/bus clk ON**	**M4 supplies ON, IO supplies ON, sys clk OFF/bus clk ON**	**M4 supplies ON, IO supplies ON, sys/bus clks OFF**	**M4 supply ON, M4 array supply ON, DGO supply ON, LVD protection, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 supply ON, reduced M4 RAM supply, reduced DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 Supply OFF, M4 RAM Supply ON, DGO supply ON, IO Supplies ON**	**M4 supply OFF, M4 RAM supply OFF, DGO supply OFF, IO supplies OFF**
SEMA42_0	M4	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Static	Static	Static	Power Gated	Power Gated
MU	M4	Functional	Functional	Functional	Functional	Functional	Functional	Clock Gated	Static	Static	Static	Power Gated	Power Gated
IOMXUC0	M4	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Static	Static	Static	Power Gated	Power Gated
PCTL_A-B[4]	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Static	Static	Static	Power Gated	Power Gated
PORTA-B IO[5]	VDD_PTA-B	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Static	Power Gated	Power Gated
RGPIO2P0	M4	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Async Operation	Async Operation	Clock Gated	Power Gated	Power Gated
FlexIO0	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Clock Gated	Power Gated	Power Gated
SAI0-1	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Clock Gated	Power Gated	Power Gated
LPI2C0-3	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
LPSPI0-1	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated

Table continues on the next page...

Table 27-8. M4 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[1]	VLLS	BAT
Modules	**Power State Power Domain**	**M4 supply on M4 RAM supply on DGO supply on FBB optional Allows DVS IO supplies on**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, Allows DVS, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, RBB optional, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, IO supplies ON**	**M4 supplies ON, IO supplies ON, sys/bus clk ON**	**M4 supplies ON, IO supplies ON, sys clk OFF/bus clk ON**	**M4 supplies ON, IO supplies ON, sys/bus clks OFF**	**M4 supply ON, M4 array supply ON, DGO supply ON, LVD protection, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 supply ON, reduced M4 RAM supply, reduced DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 Supply OFF, M4 RAM Supply ON, DGO supply ON, IO Supplies ON**	**M4 supply OFF, M4 RAM supply OFF, DGO supply OFF, IO supplies OFF**
LPUART0-3	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
LPTPM0-3	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
LPTMR0-1	DG0	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Async Operation	Functional	Power Gated
LPIT0	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Static	Power Gated	Power Gated
TRGMUX0	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Static	Power Gated	Power Gated
LTC	M4	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
TRNG	M4	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
SNVS_HP_Wrapper	M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Async Operation	Async Operation	Clock Gated	Power Gated	Power Gated
SNVS_DGO	VBAT	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional	Functional
CRC	M4	Functional	Functional	Functional	Functional	Functional	Static	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated

Table continues on the next page...

Table 27-8. M4 power mode detail (continued)

Power modes		HS RUN	RUN	VLPR	WAIT	PSTOP3	PSTOP2	PSTOP1	STOP	VLPS	LLS[1]	VLLS	BAT
Modules	**Power State Power Domain**	**M4 supply on M4 RAM supply on DGO supply on FBB optional Allows DVS IO supplies on**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, Allows DVS, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, RBB optional, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, IO supplies ON**	**M4 supplies ON, IO supplies ON, sys/bus clk ON**	**M4 supplies ON, IO supplies ON, sys clk OFF/bus clk ON**	**M4 supplies ON, IO supplies ON, sys/bus clks OFF**	**M4 supply ON, M4 array supply ON, DGO supply ON, LVD protection, IO supplies ON**	**M4 supply ON, M4 RAM supply ON, DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 supply ON, reduced M4 RAM supply, reduced DGO supply ON, LVD protection, RBB optional, IO supplies ON**	**M4 Supply OFF, M4 RAM Supply ON, DGO supply ON, IO Supplies ON**	**M4 supply OFF, M4 RAM supply OFF, DGO supply OFF, IO supplies OFF**
MMCAU	M4	Functional	Functional	Functional	Clock Gated	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
ADC0-1	ANA/M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Functional	Power Gated	Power Gated
DAC0-1	ANA/M4	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Functional	Power Gated	Power Gated
CMP0-1	ANA/DGO	Functional	Functional	Functional	Functional	Functional	Async Operation	Async Operation	Functional	Functional	Functional	Functional	Power Gated
CoreSight Components	M4	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Clock Gated	Power Gated	Power Gated
ETF buffer	M4 RAM	Functional	Functional	Functional	Functional	Functional	Clock Gated	Clock Gated	Static	Static	Static	Power Gated	Power Gated

1. VLPS and LLS modes work in a very similar way, except that LLS causes the memories to enter RETENTION mode and their supply to be reduced (supply reduction applies to M4 domain only). Besides, pins remain isolated while the domain is in LLS mode.
2. Static: clock(s) is(are) on, but the module is not operating.
3. Here, Async Operation means the block supports the DMA transfers from/to IPs that are working in Assynchronous Operation state
4. PCTLs are capable of detection external interrupts
5. LLWU wake-up pins are active in VLLS mode.

M4 VLLS mode has different levels of granularity. The following table shows the continuum of options in VLLS. The user has the option to select various features that meet the requirements of the application. This table shows what is possible in VLLS from the lowest leakage to more functionality with more power consumption. There are many more options available in the continuum. As an example, the memories can be turned on or off in 32KB blocks.

Table 27-9. M4 VLLS mode

M4 VLLS Mode		Description
VLLS	Highest leakage	• Common VLLS logic state. • Comparator and LPTMR peripherals are running. • Selected low power clock sources are on. • All memory are retained.
		• Common VLLS logic state. • Selected Comparator and LPTMR peripherals are running • Selected low power clock sources are on. • Selected memory are retained.
	...	...
		• Common VLLS logic state. • No peripherals active. • Selected memory are retained.
	Lowest leakage	• Common VLLS logic state. • No peripheral active. • Only 32KB memory retained.

Warning

It is recommended to keep PTA29 low (=0) while waking up the RTD from VLLS mode to avoid unexpected behavior.

27.11.3 VBAT domain power

The VBAT domain supplies a small, very low leakage piece of logic that contains a RTC, wake-up logic, temperature and voltage sensors, a small amount of memory, and a voltage regulator. The input range to the VBAT domain is 2.0 to 4.2 Volts so it can be powered directly from the systems battery or from an independent source, such as a super cap, or from the main supply.

Table 27-10. VBAT mode

VBAT mode	Description
VBAT	• Battery mode. • Secure RTC is on. • Tamper logic retained.

Chapter 28
Power Management Controller (PMC)

28.1 Chip-specific PMC information

Table 28-1. Reference links to related information

Topic	Related module	Reference
Full description	PMC	PMC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

28.1.1 Digital Power Management Controller (Digital PMC)

The Digital PMC module allows user software to control power modes of the chip and to optimize power consumption for the level of functionality needed. There are two instances of Digital PMC on this device, one for each main power domain.

Table 28-2. Digital PMC configuration

Parameter	Description
Name	Digital Power Management Controller (Digital PMC)
Instances	2
Configurable features	Refer to Power Management for detail
Interface speed	NA
External I/O pins	NA

28.1.2 Supported modes of operation

- VLPS and LLS modes work in a very similar way, except that LLS causes the memories to enter RETENTION mode and their supply to be reduced (supply reduction applies to M4 domain only). Besides, pins remain isolated while the domain is in LLS mode.
- RBB should not be enabled on A7 domain when any clock is active. Only static RBB is supported for A7 domain.
- After exiting VLLS mode, IOMUXC will be reset and all pad configurations will default to reset values. Before asserting CTRL[ISOACK] bit, IOMUX needs to be reconfigured to resume pad configuration prior to entering VLLS mode. See Pads Isolation for more details.
- When exiting VLLS mode, the isolation in PTA26, PTA27 and PTA28 pads is released automatically if one of PTA26 or PTA27 is configured as JTAG/SWD function before entering VLLS mode. See Pads Isolation

28.1.3 Memory power control

Memory power controls are divided into three main groups:

- M4 power domain memories that are controlled by PMC registers PMC0_SRAMCTRL_0|1|2: These are basically the core TCM memories. The idea is that application has control over the amount of TCM memory it can power-gate in lower power modes. The table below shows how each bit of these registers are mapped and the corresponding TCM system address range.

Table 28-3. Bit mapping of PMC_0_SRAMCTRL_0|1|2 registers and corresponding TCM address range

PWR GROUPS	PMC0_SRAMCTRL_0|1|2	i.MX 7ULP system	Size
	bit	Address	
cm4_tcm_bank0	0	0x1FFD_0000 - 0x1FFD_7FFF	32 KB
cm4_tcm_bank1	1	0x1FFD_8000 - 0x1FFD_FFFF	32 KB
cm4_tcm_bank2	2	0x1FFE_0000 - 0x1FFE_FFFF	64 KB
cm4_tcm_bank3	3	0x1FFF_0000 - 0x1FFF_FFFF	64 KB
cm4_tcm_bank4	4	0x2000_0000 - 0x2000_FFFF	64 KB

- M4 domain memories without register control (CM4_NO_TCM): This comprises the DMA, Caches, Trace Memory, and ROM.
- A7 domain memories: All of A7 domain have PMC register control. The table below shows how PMC register bits are mapped to chip-specific memories.

Table 28-4. Bit mapping of PMC_1_SRAMCTRL_0 register to chip-specific memories

PWR GROUPS	PMC1_SRAMCTRL_0
	bit
CA7 core	0
GPU3D	1
GPU2D	2
CA7 OCRAMs	3
USB	4
VIU	5
LCDIF	6
MISC(DSI, DMA1, CAAM, uSDHC0/1)	7

NOTE

USB PHY-related interrupt (NVIC/GIC) and wake-up channels (AWIC/WKPU) must be disabled before turning PMC_1 ON (CTRL[PMC1ON] in PMC_0).

28.1.4 Operation voltage requirements

The default operation voltage levels defined via PMC0 *power mode* [COREREGVL] and PMC1 *power mode*[LDOVL] fields don't necessarily satisfy the operation voltage levels required by the system. Such operation voltage requirements can be found on i.MX 7ULP Data Sheet. In this sense, no power transition should be performed before ensuring the mentioned register fields are matching the required operation voltage levels expressed on the data sheet.

In a scenario that the application domain is supplied by an external PMIC, the same operation voltage requirements hold; the only difference being that the operational voltage levels must be observed by the respective PMIC supply rather than adjusted via PMC1 *power mode*[LDOVL] fields.

28.2 Introduction

The Power Management Controller (PMC) can be divided into two parts: PMC 0 and PMC 1. The PMC 0 controls the Core 0 SoG supply, the Core 0 SRAMs power modes and the Core 0 biasing. The PMC 1 controls the Core 1 SoG supply, the Core 1 SRAMs power modes and the Core 1 biasing. The PMC also has voltage monitors in both power domains (PMC 0 and PMC 1).

Both PMCs receive requests from the MSMC to change the current power mode. Each PMC allows the customer to choose what features will be enabled or disabled by each power mode.

The Figure 28-1 shows the PMC block diagram.

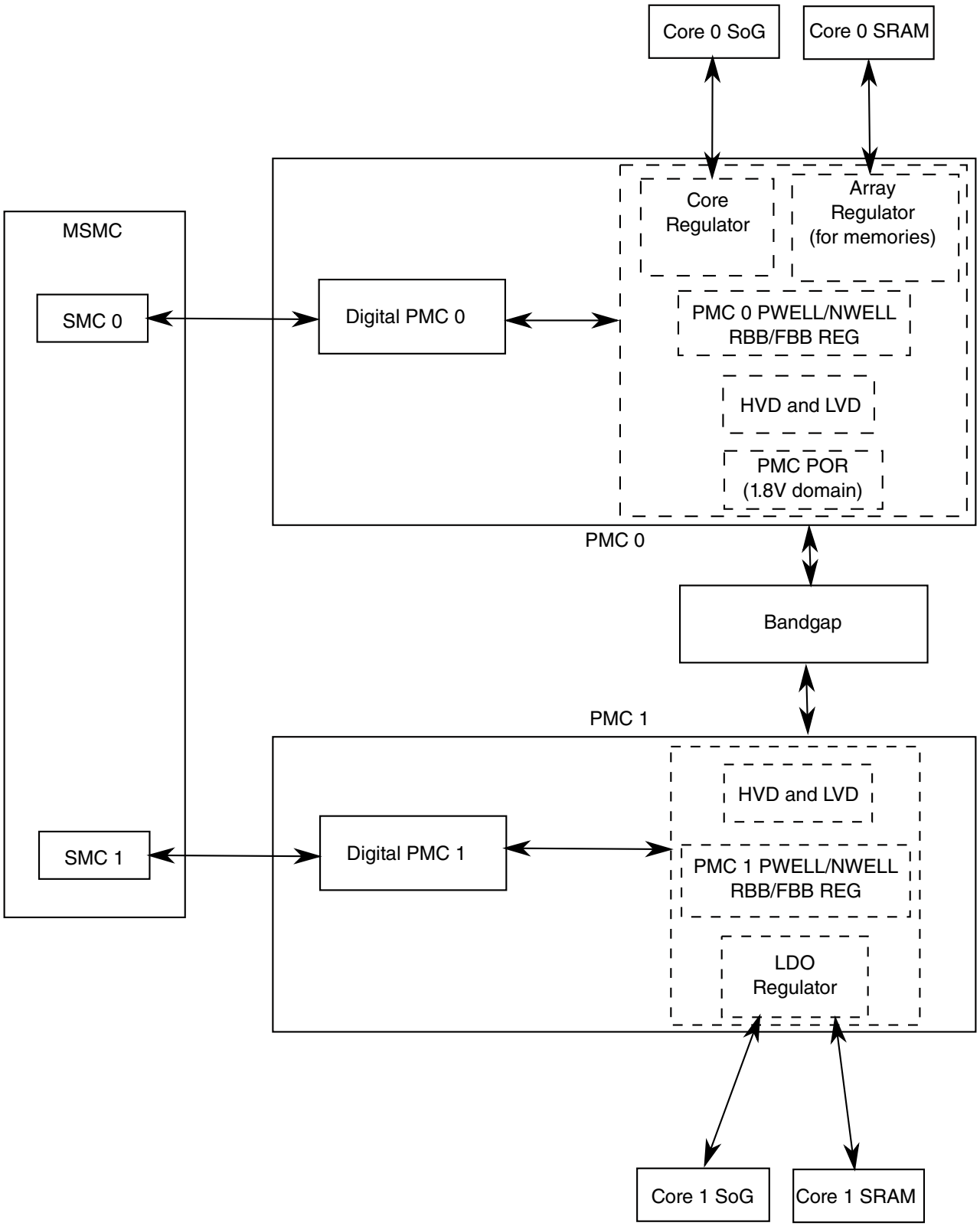


Figure 28-1. PMC Block Diagram

28.3 Features

The PMC 0 has the following features:

- a high-power (HP) and a low-power (LP) Core Regulator;
- a high-power (HP) and a low-power (LP) Array Regulator;
- a high-power (HP) and a low-power (LP) 1.2V Low Voltage Detector (LVD) monitor (in regulator input);
- a high-power (HP) 1.8V High Voltage Detector (HVD) monitor (in regulator supply);
- a bandgap;
- a forward back bias (FBB) and a reverse back bias (RBB) regulator.
- a temperature sensor.

The PMC 1 has the following features:

- a high-power (HP) and a low-power (LP) Linear LDO regulator;
- a set of power switches;
- a high-power (HP) and a low-power (LP) Low Voltage Detector (LVD) monitor;
- a high-power (HP) 1.2V High Voltage Detector (HVD) monitor;
- configurable LVD/HVD sense point between regulator supply and regulator output;
- a forward back bias (FBB) and a reverse back bias (RBB) regulator.

In addition, the PMC has a 1.8V POR (Power-On Reset) monitor to detect the voltage level in the Always-On power domain.

28.4 PMC register descriptions

This section describes the PMC registers.

NOTE

All registers must be written only during the RUN mode. Any intention to write the registers during VLPR or HSRUN modes could result in an unxpected behavior of the system. The exceptions to this rule are the COREREGVL (PMC0_HSRUN) and LDOVL (PMC1_HSRUN) bitfields that also could be written during the PMC 0 and PMC 1 HSRUN modes respectively.

28.4.1 PMC1 Memory map

PMC1 base address: 4040_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PMC 1 Version register (VERID)	32	RO	0000_0000h
4h	PMC 1 HSRUN mode register (HSRUN)	32	RW	0028_0000h
8h	PMC 1 RUN mode register (RUN)	32	RW	0028_0000h
Ch	PMC 1 VLPR mode register (VLPR)	32	RW	000A_0000h
10h	PMC 1 STOP mode register (STOP)	32	RW	0028_0000h
14h	PMC 1 VLPS mode register (VLPS)	32	RW	000A_0000h
18h	PMC 1 LLS mode register (LLS)	32	RW	000A_0000h
1Ch	PMC 1 VLLS mode register (VLLS)	32	RW	0000_0000h
20h	PMC 1 Status register (STATUS)	32	RO	0000_0000h
24h	PMC 1 Control register (CTRL)	32	RW	0000_0000h
34h	PMC 1 Biasing Control register (BCTRL)	32	RW	See description.
44h	PMC 1 SRAMs Control register (SRAMCTRL)	32	RW	0000_0000h

28.4.2 PMC0 Memory map

PMC0 base address: 410A_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PMC 0 Version register (VERID)	32	RO	0000_0000h
4h	PMC 0 Power Mode Status register (PM_STAT)	32	RO	0001_0001h
8h	PMC 0 HSRUN mode register (HSRUN)	32	RW	001C_0000h
Ch	PMC 0 RUN mode register (RUN)	32	RW	001C_0000h
10h	PMC 0 VLPR mode register (VLPR)	32	RW	000A_0000h
14h	PMC 0 STOP mode register (STOP)	32	RW	001C_0000h
18h	PMC 0 VLPS mode register (VLPS)	32	RW	000A_0000h
1Ch	PMC 0 LLS mode register (LLS)	32	RW	000A_0000h
20h	PMC 0 VLLS mode register (VLLS)	32	RW	0000_0002h
24h	PMC 0 Status register (STATUS)	32	RO	8000_0000h
28h	PMC 0 Control register (CTRL)	32	RW	See description.
30h	PMC 0 Analog Core Control register (ACTRL)	32	RW	See description.

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
38h	PMC 0 Biasing Control register (BCTRL)	32	RW	See description.
48h	PMC 0 SRAMs Control 0 register (SRAMCTRL_0)	32	RW	0000_0000h
4Ch	PMC 0 SRAMs Control 1 register (SRAMCTRL_1)	32	RW	0000_0000h
50h	PMC 0 SRAMs Control 2 register (SRAMCTRL_2)	32	RW	0000_0000h

28.4.3 PMC 0 Version register (VERID)

28.4.3.1 Offset

Register	Offset
VERID	0h

28.4.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.3.3 Fields

Field	Function
31-24	Major Version Number
MAJOR	This read only field returns the major version number for the module specification.
23-16	Minor Version Number
MINOR	This read only field returns the minor version number for the module specification.

Table continues on the next page...

Field	Function
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000000000000b - Standard features implemented

28.4.4 PMC 1 Version register (VERID)

28.4.4.1 Offset

Register	Offset
VERID	0h

28.4.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.4.3 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000000000000b - Standard features implemented

28.4.5 PMC 0 Power Mode Status register (PM_STAT)

28.4.5.1 Offset

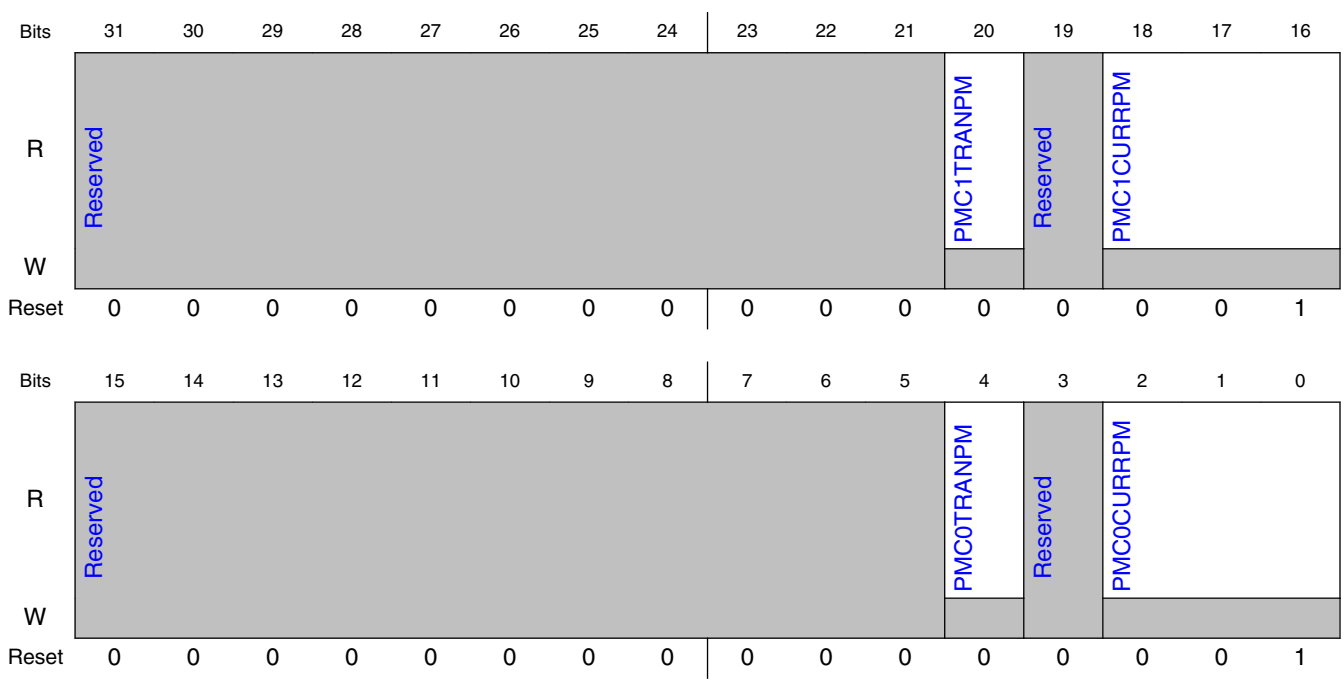
Register	Offset
PM_STAT	4h

28.4.5.2 Function

NOTE

It is expected that bus clock is kept stable during all mode transition, otherwise PM_STAT information can be inconsistent.

28.4.5.3 Diagram



28.4.5.4 Fields

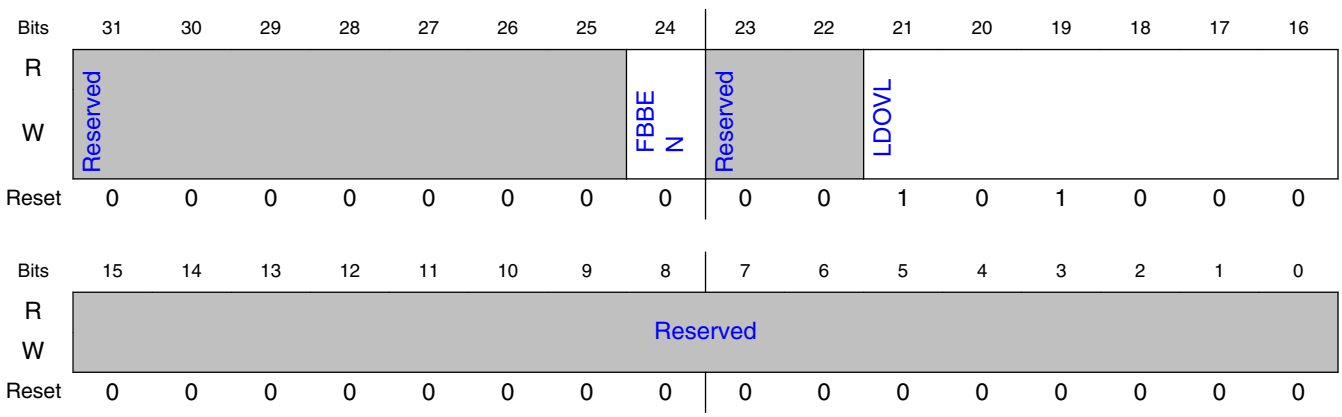
Field	Function
31-21 —	Reserved field
20 PMC1TRANPM	PMC 1 Power Mode transition status Signals whether or not a power mode transition is ongoing. 0b - PMC 1 is not in a power mode transition. 1b - PMC 1 is in a power mode transition.
19 —	Reserved field
18-16 PMC1CURRPM	PMC 1 Current Power Mode Current power mode of PMC 1. **NOTE:** Ignore this bitfield if PMC0_CTRL[PMC1ON] bit is cleared, PMC 1 is not powered up in this case. 000b - HSRUN Mode 001b - RUN Mode 010b - STOP Mode 011b - VLPR Mode 100b - VLPS Mode 101b - LLS Mode 110b - VLLS Mode
15-5 —	Reserved field
4 PMC0TRANPM	PMC 0 Power Mode transition status Signals whether or not a power mode transition is ongoing. **NOTE:** This register does not change in transitions from/to STOP, VLPS, LLS or VLLS. 0b - PMC 0 is not in a power mode transition. 1b - PMC 0 is in a power mode transition.
3 —	Reserved field
2-0 PMC0CURRPM	PMC 0 Current Power Mode Current power mode of PMC 0. **NOTE:** This register does not change in transitions from/to STOP, VLPS, LLS or VLLS. 000b - HSRUN Mode 001b - RUN Mode 010b - STOP Mode 011b - VLPR Mode 100b - VLPS Mode 101b - LLS Mode 110b - VLLS Mode

28.4.6 PMC 1 HSRUN mode register (HSRUN)

28.4.6.1 Offset

Register	Offset
HSRUN	4h

28.4.6.2 Diagram



28.4.6.3 Fields

Field	Function
31-25 —	Reserved field
24 FBBEN	Forward Back Bias Enable 0b - FBB is disabled 1b - FBB is enabled
23-22 —	Reserved field
21-16 LDOVL	LDO Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.6V to 1.1V with resolution of 10mV.The reset value corresponds to 1.0V. **NOTE:** When a new value is written in this bitfield during the PMC 1 HSRUN mode, a voltage level change request will be sent to regulator be adjusted and the LDOVLF flag will be set. Otherwise, when the PMC 1 isn't in HSRUN mode, the new voltage value will be adjusted in a mode transition to PMC 1 HSRUN only. 000000b - LDO Voltage Level is 0.60V

Table continues on the next page...

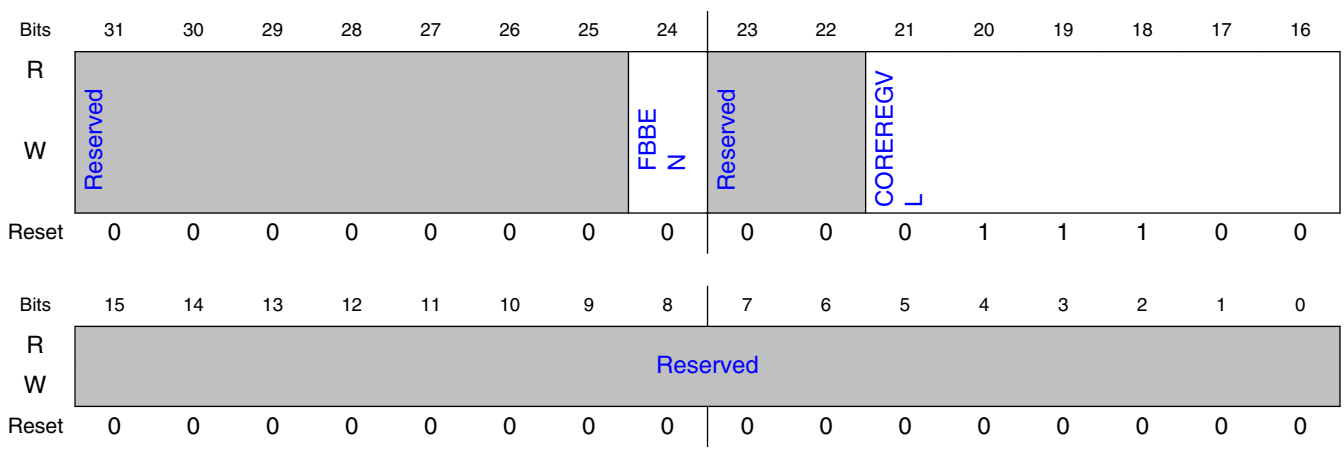
Field	Function
	000001b - LDO Voltage Level is 0.61V 110001b - LDO Voltage Level is 1.09V 110010b - LDO Voltage Level is 1.10V
15-0 —	Reserved field

28.4.7 PMC 0 HSRUN mode register (HSRUN)

28.4.7.1 Offset

Register	Offset
HSRUN	8h

28.4.7.2 Diagram



28.4.7.3 Fields

Field	Function
31-25 —	Reserved field
24	Forward Back Bias Enable 0b - FBB is disabled

Table continues on the next page...

Field	Function
FBBEN	1b - FBB is enabled
23-22 —	Reserved field
21-16 COREREGVL	Core Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.596V to 1.138V with resolution of 10.83mV.The reset value corresponds to 0.9V. **NOTE:** When a new value is written in this bitfield during the PMC 0 HSRUN mode, a voltage level change request will be sent to regulator be adjusted and the COREVLF flag will be set. Otherwise, when the PMC 0 isn't in HSRUN mode, the new voltage value will be adjusted in a mode transition to PMC 0 HSRUN only. 000000b - Core Voltage Level is 0.596V 000001b - Core Voltage Level is 0.607V 110001b - Core Voltage Level is 1.127V 110010b - Core Voltage Level is 1.138V
15-0 —	Reserved field

28.4.8 PMC 1 RUN mode register (RUN)

28.4.8.1 Offset

Register	Offset
RUN	8h

28.4.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										LDOVL					
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.8.3 Fields

Field	Function
31-22 —	Reserved field
21-16 LDOVL	LDO Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.6V to 1.1V with resolution of 10mV.The reset value corresponds to 1.0V. **NOTE:** When a new value is written in this bitfield during the PMC 1 RUN mode, a voltage level change request will be sent to regulator be adjusted and the LDOVLF flag will be set. Otherwise, when the PMC 1 isn't in RUN mode, the new voltage value will be adjusted in a mode transition to PMC 1 RUN only. **NOTE:** During RUN mode, the voltage adjustment should be less or equal than 100mV. **NOTE:** The user always must write the default value in this bitfield after any reset. 000000b - LDO Voltage Level is 0.60V 000001b - LDO Voltage Level is 0.61V 110001b - LDO Voltage Level is 1.09V 110010b - LDO Voltage Level is 1.10V
15-0 —	Reserved field

28.4.9 PMC 0 RUN mode register (RUN)

28.4.9.1 Offset

Register	Offset
RUN	Ch

28.4.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										CORREGVL					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.9.3 Fields

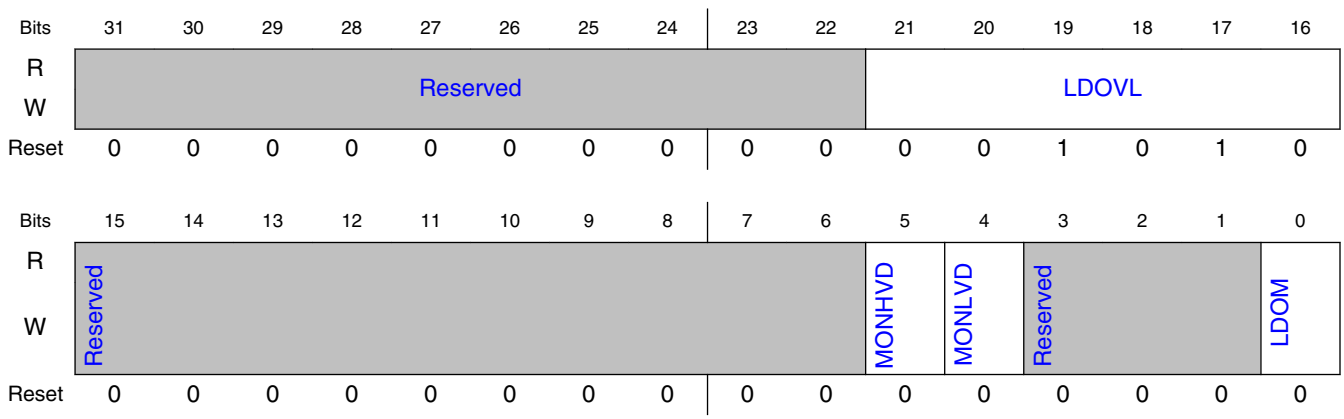
Field	Function
31-22 —	Reserved field
21-16 COREREGVL	Core Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.596V to 1.138V with resolution of 10.83mV.The reset value corresponds to 0.9V. **NOTE:** When a new value is written in this bitfield during the PMC 0 RUN mode, a voltage level change request will be sent to regulator be adjusted and the COREVLF flag will be set. Otherwise, when the PMC 0 isn't in RUN mode, the new voltage value will be adjusted in a mode transition to PMC 0 RUN only. **NOTE:** During RUN mode, the voltage adjustment should be less or equal than 100mV. **NOTE:** Due to SRAM electrical restrictions this bitfield shall not be configured to values higher than 1.0V prior to a LLS or VLLS entry. 000000b - Core Voltage Level is 0.596V 000001b - Core Voltage Level is 0.607V 110001b - Core Voltage Level is 1.127V 110010b - Core Voltage Level is 1.138V
15-0 —	Reserved field

28.4.10 PMC 1 VLPR mode register (VLPR)

28.4.10.1 Offset

Register	Offset
VLPR	Ch

28.4.10.2 Diagram



28.4.10.3 Fields

Field	Function
31-22 —	Reserved field
21-16 LDOVL	LDO Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.6V to 1.1V with resolution of 10mV. The reset value corresponds to 0.7V. 000000b - LDO Voltage Level is 0.60V 000001b - LDO Voltage Level is 0.61V 110001b - LDO Voltage Level is 1.09V 110010b - LDO Voltage Level is 1.10V
15-6 —	Reserved field
5 MONHVD	1.2V HP High-Voltage Detector 0b - The monitor is disabled. 1b - The monitor is enabled.
4 MONLVD	Low-Voltage Detector 0b - LP monitor is enabled. 1b - HP monitor is enabled.
3-1	Reserved field

Table continues on the next page...

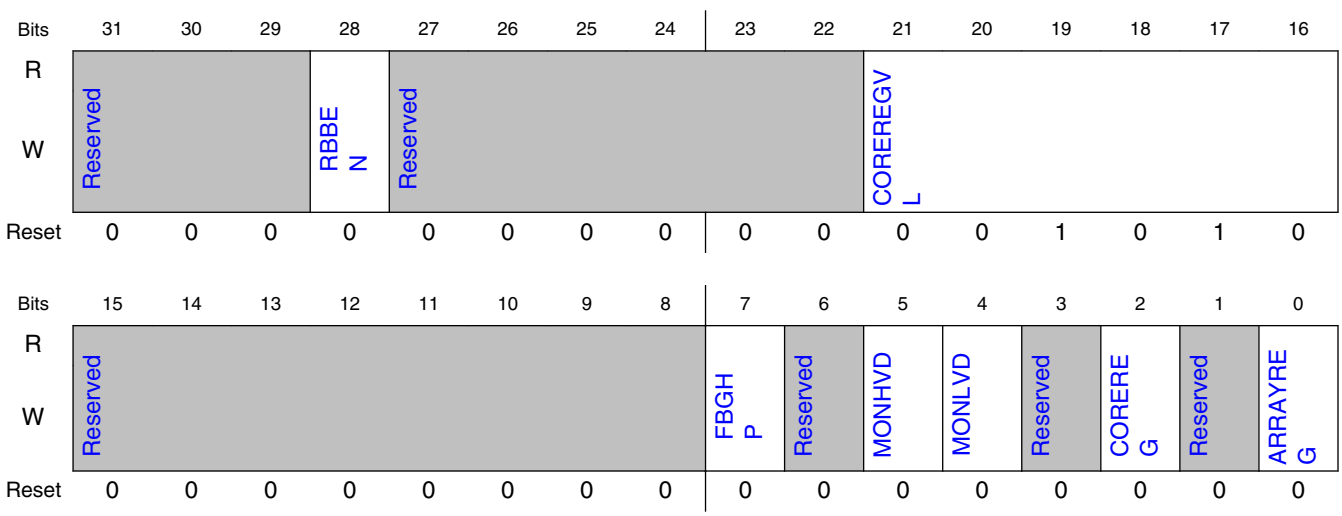
Field	Function
—	
0 LDOM	LDO Regulator Mode 0b - Linear LDO LP Regulator is enabled. 1b - Linear LDO HP Regulator is enabled.

28.4.11 PMC 0 VLPR mode register (VLPR)

28.4.11.1 Offset

Register	Offset
VLPR	10h

28.4.11.2 Diagram



28.4.11.3 Fields

Field	Function
31-29 —	Reserved field
28	Reverse Back Bias Enable

Table continues on the next page...

Field	Function
RBBEN	0b - RBB is disabled 1b - RBB is enabled
27-22 —	Reserved field
21-16 COREREGVL	Core Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.596V to 1.138V with resolution of 10.83mV.The reset value corresponds to 0.704V. **NOTE:** Due to SRAM electrical restrictions this bitfield shall not be configured to values higher than 1.0V prior to a LLS or VLLS entry. 000000b - Core Voltage Level is 0.596V 000001b - Core Voltage Level is 0.607V 110001b - Core Voltage Level is 1.127V 110010b - Core Voltage Level is 1.138V
15-8 —	Reserved field
7 FBGHP	Force HP band-gap 0b - No action 1b - Turn on the HP band-gap
6 —	Reserved field
5 MONHVD	1.8V HVD HP Monitor Enable 0b - The monitor is disabled 1b - The monitor is enabled
4 MONLVD	1.2V LVD HP Monitor Enable 0b - LP monitor is enabled 1b - HP monitor is enabled
3 —	Reserved field
2 COREREG	Core Regulator Enable 0b - LP Regulator is on 1b - HP Regulator is on
1 —	Reserved field
0 ARRAYREG	Array Regulator 0b - LP Regulator is on 1b - HP Regulator is on

28.4.12 PMC 1 STOP mode register (STOP)

28.4.12.1 Offset

Register	Offset
STOP	10h

28.4.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										LDOVL					
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.12.3 Fields

Field	Function
31-22 —	Reserved field
21-16 LDOVL	LDO Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.6V to 1.1V with resolution of 10mV. The reset value corresponds to 1.0V. 000000b - LDO Voltage Level is 0.60V 000001b - LDO Voltage Level is 0.61V 110001b - LDO Voltage Level is 1.09V 110010b - LDO Voltage Level is 1.10V
15-0 —	Reserved field

28.4.13 PMC 0 STOP mode register (STOP)

28.4.13.1 Offset

Register	Offset
STOP	14h

28.4.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										COREREGVL					
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.13.3 Fields

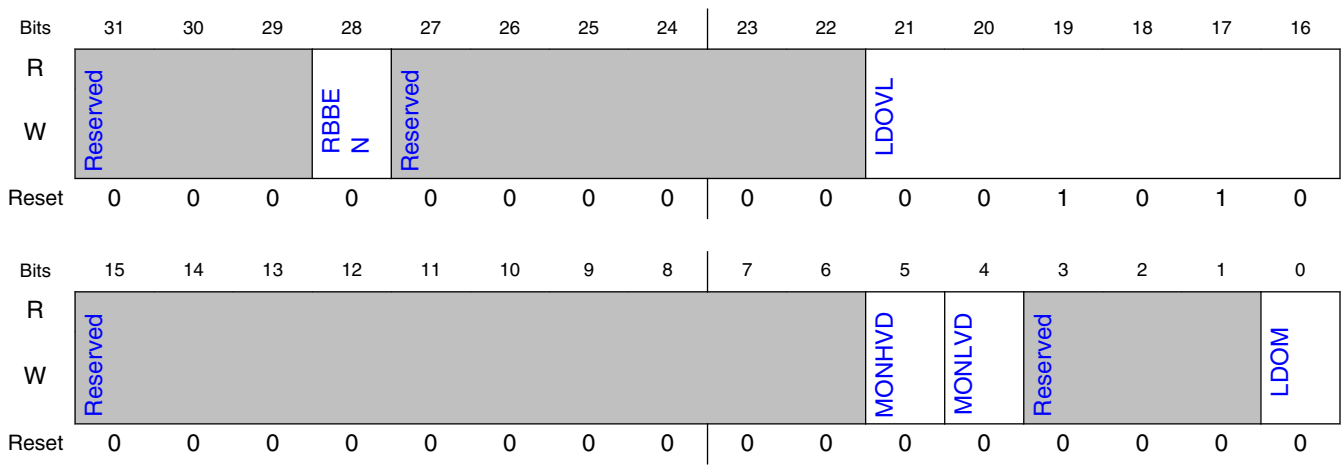
Field	Function
31-22 —	Reserved field
21-16 COREREGVL	Core Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.596V to 1.138V with resolution of 10.83mV. The reset value corresponds to 0.9V. 000000b - Core Voltage Level is 0.596V 000001b - Core Voltage Level is 0.607V 110001b - Core Voltage Level is 1.127V 110010b - Core Voltage Level is 1.138V
15-0 —	Reserved field

28.4.14 PMC 1 VLPS mode register (VLPS)

28.4.14.1 Offset

Register	Offset
VLPS	14h

28.4.14.2 Diagram



28.4.14.3 Fields

Field	Function
31-29 —	Reserved field
28 RBBEN	Reverse Back Bias Enable 0b - RBB is disabled 1b - RBB is enabled
27-22 —	Reserved field
21-16 LDOVL	LDO Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.6V to 1.1V with resolution of 10mV. The reset value corresponds to 0.7V. 000000b - LDO Voltage Level is 0.60V 000001b - LDO Voltage Level is 0.61V 110001b - LDO Voltage Level is 1.09V 110010b - LDO Voltage Level is 1.10V
15-6 —	Reserved field

Table continues on the next page...

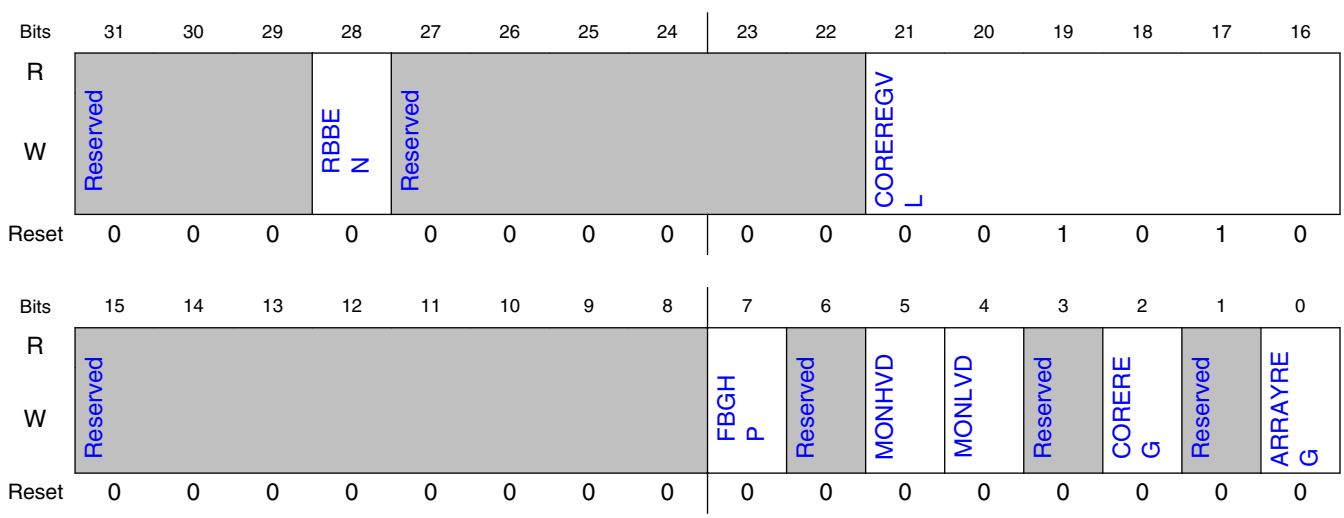
Field	Function
5 MONHVD	1.2V HP High-Voltage Detector 0b - The monitor is disabled. 1b - The monitor is enabled.
4 MONLVD	Low-Voltage Detector 0b - LP monitor is enabled. 1b - HP monitor is enabled.
3-1 —	Reserved field
0 LDOM	LDO Regulator Mode 0b - Linear LDO LP Regulator is enabled. 1b - Linear LDO HP Regulator is enabled.

28.4.15 PMC 0 VLPS mode register (VLPS)

28.4.15.1 Offset

Register	Offset
VLPS	18h

28.4.15.2 Diagram



28.4.15.3 Fields

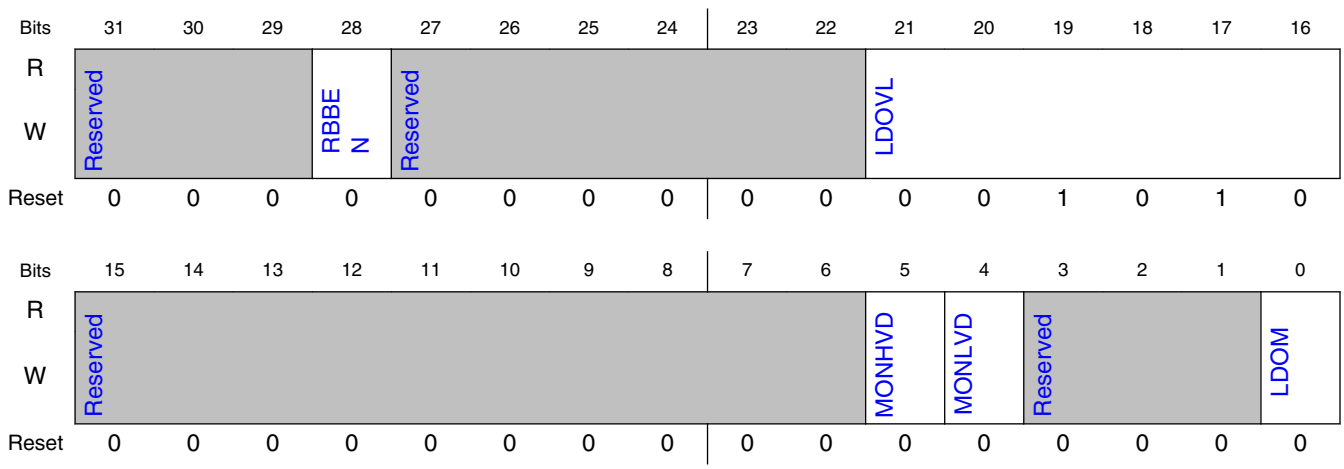
Field	Function
31-29 —	Reserved field
28 RBBEN	Reverse Back Bias Enable 0b - RBB is disabled 1b - RBB is enabled
27-22 —	Reserved field
21-16 COREREGVL	Core Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.596V to 1.138V with resolution of 10.83mV. The reset value corresponds to 0.704V. 000000b - Core Voltage Level is 0.596V 000001b - Core Voltage Level is 0.607V 110001b - Core Voltage Level is 1.127V 110010b - Core Voltage Level is 1.138V
15-8 —	Reserved field
7 FBGHP	Force HP band-gap 0b - No action 1b - Turn on the HP band-gap
6 —	Reserved field
5 MONHVD	1.8V HVD HP Monitor Enable 0b - The monitor is disabled 1b - The monitor is enabled
4 MONLVD	1.2V LVD HP Monitor Enable 0b - LP monitor is enabled 1b - HP monitor is enabled
3 —	Reserved field
2 COREREG	Core Regulator 0b - LP Regulator is on 1b - HP Regulator is on
1 —	Reserved field
0 ARRAYREG	Array Regulator 0b - LP Regulator is on 1b - HP Regulator is on

28.4.16 PMC 1 LLS mode register (LLS)

28.4.16.1 Offset

Register	Offset
LLS	18h

28.4.16.2 Diagram



28.4.16.3 Fields

Field	Function
31-29 —	Reserved field
28 RBBEN	Reverse Back Bias Enable 0b - RBB is disabled 1b - RBB is enabled
27-22 —	Reserved field
21-16 LDOVL	Linear LDO Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.6V to 1.1V with resolution of 10mV. The reset value corresponds to 0.7V. 000000b - LDO Voltage Level is 0.60V 000001b - LDO Voltage Level is 0.61V 110001b - LDO Voltage Level is 1.09V 110010b - LDO Voltage Level is 1.10V
15-6 —	Reserved field

Table continues on the next page...

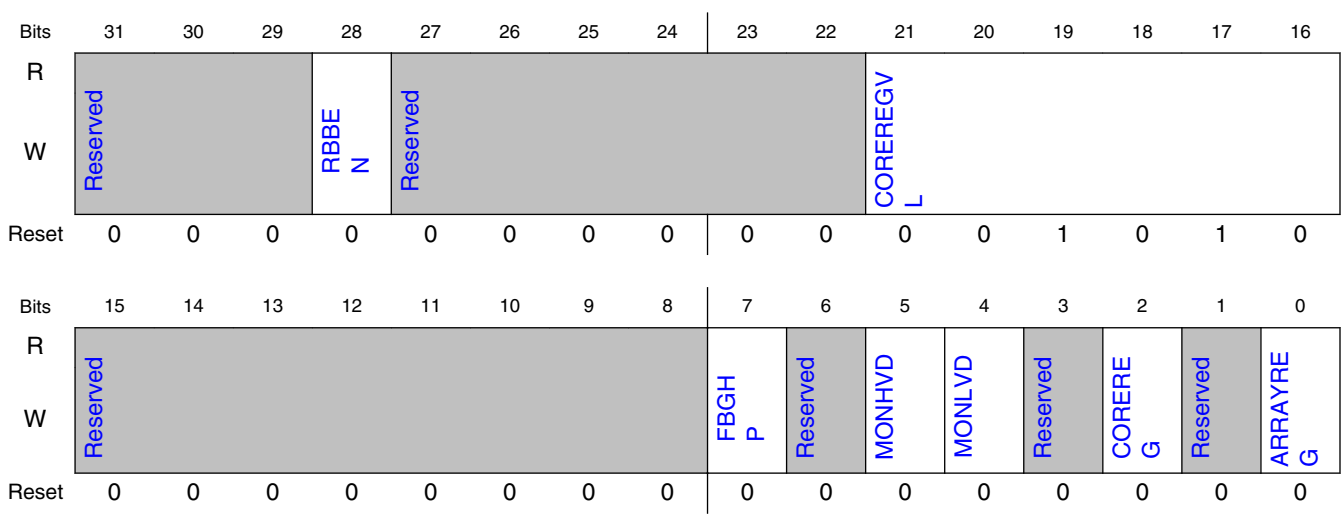
Field	Function
5 MONHVD	1.2V HP High-Voltage Detector 0b - The monitor is disabled. 1b - The monitor is enabled.
4 MONLVD	Low-Voltage Detector 0b - LP monitor is enabled. 1b - HP monitor is enabled.
3-1 —	Reserved field
0 LDOM	LDO Regulator Mode 0b - Linear LDO LP Regulator is enabled. 1b - Linear LDO HP Regulator is enabled.

28.4.17 PMC 0 LLS mode register (LLS)

28.4.17.1 Offset

Register	Offset
LLS	1Ch

28.4.17.2 Diagram



28.4.17.3 Fields

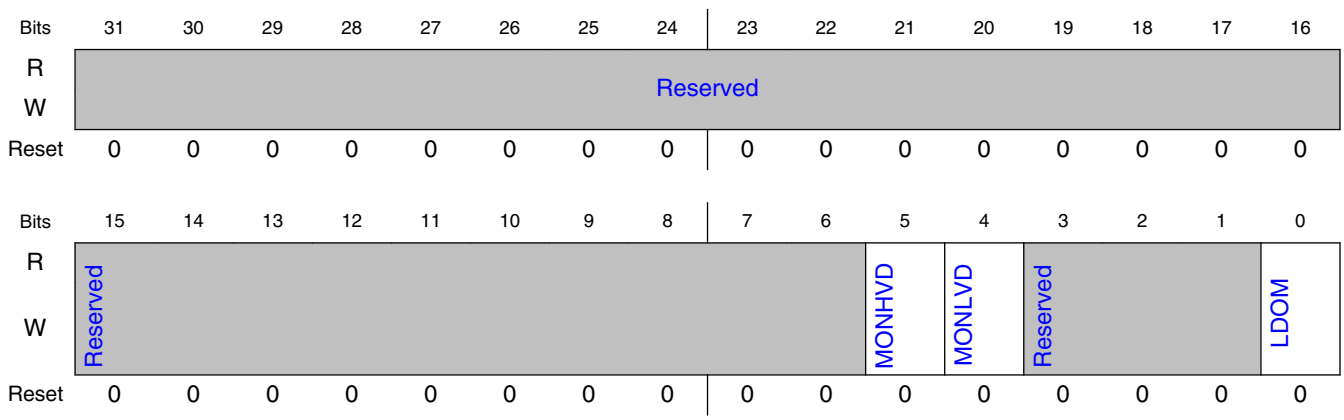
Field	Function
31-29 —	Reserved field
28 RBBEN	Reverse Back Bias Enable 0b - RBB is disabled 1b - RBB is enabled
27-22 —	Reserved field
21-16 COREREGVL	Core Regulator Voltage Level The valid values are between 0 and 50. These values correspond to a valid range of 0.596V to 1.138V with resolution of 10.83mV.The reset value corresponds to 0.704V. **NOTE:** Due to SRAM electrical restrictions this bitfield shall not be configured to values higher than 1.0V. 000000b - Core Voltage Level is 0.596V 000001b - Core Voltage Level is 0.607V 110001b - Core Voltage Level is 1.127V 110010b - Core Voltage Level is 1.138V
15-8 —	Reserved field
7 FBGHP	Force HP band-gap 0b - No action 1b - Turn on the HP band-gap
6 —	Reserved field
5 MONHVD	1.8V HVD HP Monitor Enable 0b - The monitor is disabled 1b - The monitor is enabled
4 MONLVD	1.2V LVD HP Monitor Enable 0b - LP monitor is enabled 1b - HP monitor is enabled
3 —	Reserved field
2 COREREG	Core Regulator 0b - LP Regulator is on 1b - HP Regulator is on
1 —	Reserved field
0 ARRAYREG	Array Regulator 0b - LP Regulator is on 1b - HP Regulator is on

28.4.18 PMC 1 VLLS mode register (VLLS)

28.4.18.1 Offset

Register	Offset
VLLS	1Ch

28.4.18.2 Diagram



28.4.18.3 Fields

Field	Function
31-6 —	Reserved field
5 MONHVD	1.2V HP High-Voltage Detector 0b - The monitor is disabled. 1b - The monitor is enabled.
4 MONLVD	Low-Voltage Detector 0b - LP monitor is enabled. 1b - HP monitor is enabled.
3-1 —	Reserved field
0 LDOM	LDO Regulator Mode 0b - Linear LDO LP Regulator is enabled. 1b - Linear LDO HP Regulator is enabled.

28.4.19 PMC 0 VLLS mode register (VLLS)

28.4.19.1 Offset

Register	Offset
VLLS	20h

28.4.19.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								FBGHP	Reserved	MONHVD	MONLVD	Reserved		ARRAYREG	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

28.4.19.3 Fields

Field	Function
31-8 —	Reserved field
7 FBGHP	Force HP band-gap 0b - No action 1b - Turn on the HP band-gap
6 —	Reserved field
5 MONHVD	1.8V HVD HP Monitor Enable 0b - The monitor is disabled 1b - The monitor is enabled
4	1.2V LVD HP Monitor Enable 0b - LP monitor is enabled

Table continues on the next page...

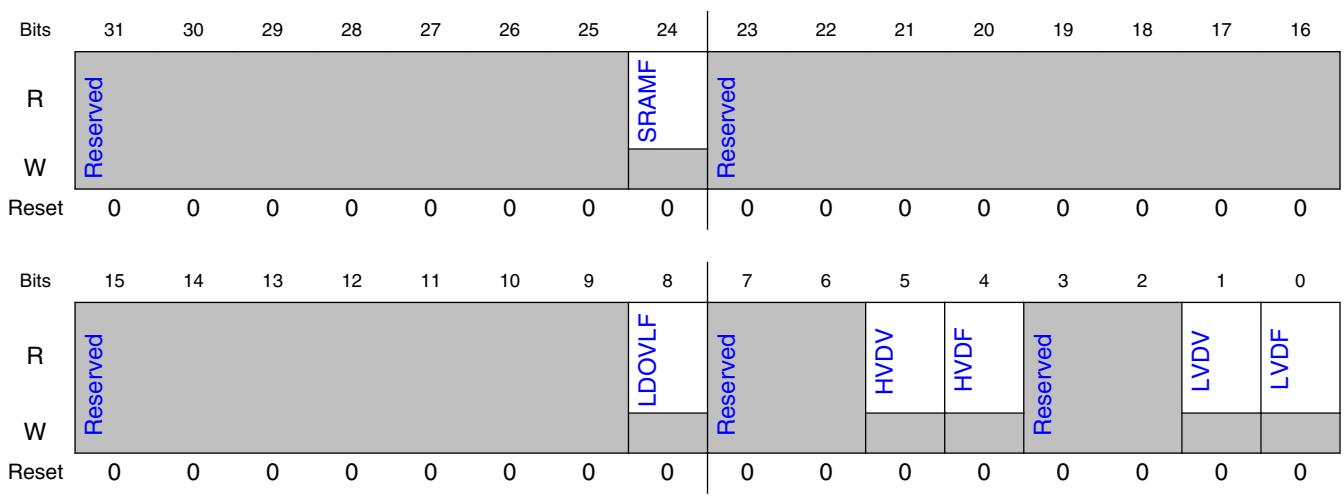
Field	Function
MONLVD	1b - HP monitor is enabled
3-2 —	Reserved field
1-0 ARRAYREG	Array Regulator 00b - Regulator is off 01b - Reserved 10b - LP Regulator is on 11b - HP Regulator is on

28.4.20 PMC 1 Status register (STATUS)

28.4.20.1 Offset

Register	Offset
STATUS	20h

28.4.20.2 Diagram



28.4.20.3 Fields

Field	Function
31-25 —	Reserved field
24 SRAMF	SRAM Flag This flag indicates if a request to some SRAM entry(or exit) to some power mode is being processed. For more information see Change SRAMs power mode in RUN . **NOTE:** The user must wait for this to be cleared to start a new power mode request in the SRAMs. To start a request when this flag is asserted could cause an unxpected behavior in the system. **NOTE:** This bit is asserted one clock after to change the PMC1_SRAMCTRL_0 value. 0b - No change request in the SRAMs. 1b - A change mode request is being processed in the SRAMs.
23-9 —	Reserved field
8 LDOVLF	LDO Voltage Level Flag This flag is asserted when the PMC is changing the LDO Voltage Level due a different value writing in the field PMC1_RUN[LDOVL] (or PMC1_HSRUN[LDOVL], depending on the PMC 1 mode). The software needs to wait for the end of the voltage level changing (when this flag is cleared) to write a new value to PMC1_RUN[LDOVL] (or PMC1_HSRUN[LDOVL]) to start a new change process. For more information see Change regulator voltage in RUN or HSRUN . **NOTE:** Before a power mode transition, the user needs to confirm this bit is cleared. **NOTE:** This bit is asserted one clock after to change the PMC1_RUN[LDOVL] value (or PMC1_HSRUN[LDOVL], depending on the PMC 1 mode). **NOTE:** In case PMC0_CTRL[LDOOKDIS] bit is cleared, the PMC checks the regulated LDO output to clear this flag. Otherwise, when LDOOKDIS bit is asserted, PMC doesn't check the LDO output and it will take ~2µs by each 30mV in a voltage increment or ~3µs by each 20mV in a voltage decrement to clear this flag. 0b - LDO Voltage Level is stable 1b - LDO Voltage Level is changing
7-6 —	Reserved field
5 HVDV	1.2V High-Voltage Detector Value This bit is the current value of the 1.2V HVD monitor in the PMC 1. For more information see Voltage Monitors . 0b - High-voltage event was not detected by the 1.2V HVD monitor in PMC 1 1b - High-voltage event was detected by the 1.2V HVD monitor in PMC 1
4 HVDF	1.2V High-Voltage Detector Flag This flag is set when a high-voltage event was detected by the 1.2V HVD monitor in the PMC 1. This flag is cleared when 1 is written to the bit HVDACK in the register PMC1_CTRL. For more information see Voltage Monitors . **NOTE:** This bit will only be cleared if HVDV is cleared. 0b - High-voltage event was not detected by the 1.2V HVD monitor in PMC 1 1b - High-voltage event was detected by the 1.2V HVD monitor in PMC 1
3-2 —	Reserved field
1	1.2V Low-Voltage Detector Value

Table continues on the next page...

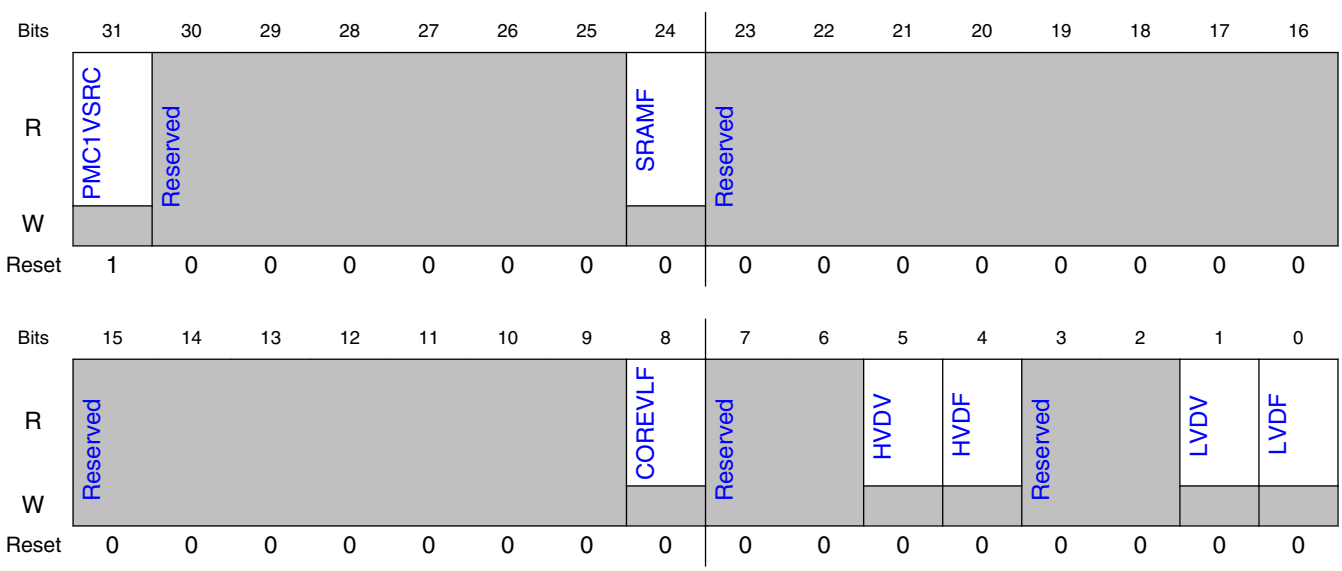
Field	Function
LVDV	This bit is the current value of the 1.2V LVD monitor in the PMC 1. For more information see Voltage Monitors . 0b - Low-voltage event was not detected by the 1.2V LVD monitor in the PMC 1 1b - Low-voltage event was detected by the 1.2V LVD monitor in the PMC 1
0 LVDF	1.2V Low-Voltage Detector Flag This flag is set when a low-voltage event was detected by the 1.2V LVD monitor in the PMC 1. This flag is cleared when 1 is written to the bit LVDACK in the register PMC1_CTRL. For more information see Voltage Monitors . **NOTE:** This bit will only be cleared if LVDV is cleared. 0b - Low-voltage event was not detected by the 1.2V LVD monitor in the PMC 1 1b - Low-voltage event was detected by the 1.2V LVD monitor in the PMC 1

28.4.21 PMC 0 Status register (STATUS)

28.4.21.1 Offset

Register	Offset
STATUS	24h

28.4.21.2 Diagram



28.4.21.3 Fields

Field	Function
31 PMC1VSRC	PMC 1 Voltage Source This flag indicates what is the voltage source selected to supply the PMC 1 and where the sense point of the PMC 1's LVD/HVD is placed. **NOTE:** The voltage source and LVD/HVD sense point of the PMC 1 are selected by the PMC0_CTRL[LDOEN] bit during a PMC 1 power-up. After that, the voltage source and sense point can be changed after a POR event only. For more information see Voltage Monitors . 0b - The internal LDO supplies the PMC 1, the PMC 1's LVD/HVD sense point is at the supply of the LDO regulator. 1b - The external PMIC supplies the PMC 1; the PMC 1's LVD/HVD sense point is at the pin connected to the PMIC.
30-25 —	Reserved field
24 SRAMF	SRAM Flag This flag indicates if a request to some SRAM entry(or exit) to some power mode is being processed. For more information see Change SRAMs power mode in RUN . **NOTE:** The user must wait for this flag to be cleared to start a new power mode request in the SRAMs. To start a request when this flag is asserted could cause an unxpected behavior in the system. **NOTE:** This bit could be asserted several clocks after to change the PMC0_SRAMCTRL_0 or the PMC0_SRAMCTRL_2 value. 0b - No change request in the SRAMs. 1b - A change mode request is being processed in the SRAMs.
23-9 —	Reserved field
8 COREVLF	Core Regulator Voltage Level Flag This flag is asserted when the PMC is changing the Core Regulator Voltage Level due a different value writing in the fields PMC0_RUN[COREREGVL] (or PMC0_HSRUN[COREREGVL], depending on the PMC 0 mode). The software needs to wait for the end of the voltage level changing (when this flag is cleared) to write a new value in PMC0_RUN[COREREGVL] (or PMC0_HSRUN[COREREGVL]) to start a new change process. For more information see Change regulator voltage in RUN or HSRUN . **NOTE:** Before a power mode transition the user needs to confirm this bit is cleared. **NOTE:** This bit could be asserted several clocks after to change the PMC0_RUN[COREREGVL] value (or PMC0_HSRUN[COREREGVL], depending on the PMC 0 mode). **NOTE:** In case of a voltage level increment, the PMC waits for the internal regulated output to reach the desired voltage, then this flag is cleared. Otherwise, in case of a voltage level decrement, there is a ~3µs timeout by each 21.66mV to wait for the internal regulated output to reach the desired voltage, then this flag is cleared. The timeout will be reached in case the regulator takes a very long time to decrease the voltage level by small loads in the regulator. 0b - Core Regulator Voltage Level is stable 1b - Core Regulator Voltage Level is changing
7-6 —	Reserved field
5 HVDV	1.8V High-Voltage Detector Value This bit is the current value of the 1.8V HVD monitor in the PMC 0. For more information see Voltage Monitors . 0b - High-voltage event was not detected by the 1.8V HVD monitor in the PMC 0 1b - High-voltage event was detected by the 1.8V HVD monitor in the PMC 0

Table continues on the next page...

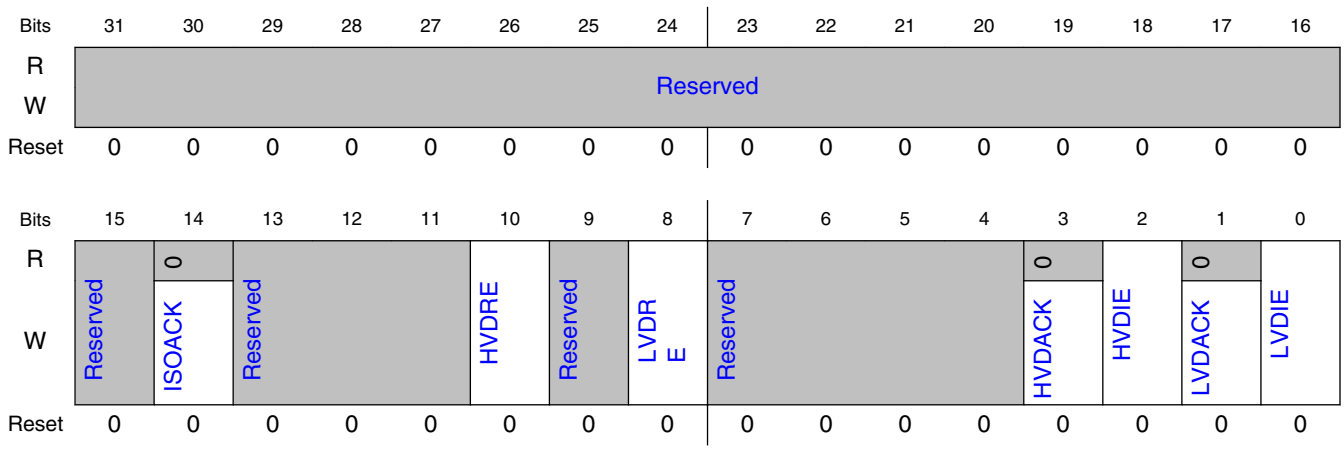
Field	Function
4 HVDF	1.8V High-Voltage Detector Flag This flag is set when a high-voltage event was detected by the 1.8V HVD monitor in the PMC 0. This flag is cleared when 1 is written to the bit HVDACK in the register PMC0_CTRL. For more information see Voltage Monitors . **NOTE:** This bit will only be cleared if HVDV is cleared. 0b - High-voltage event was not detected by the 1.8V HVD monitor in the PMC 0 1b - High-voltage event was detected by the 1.8V HVD monitor in the PMC 0
3-2 —	Reserved field
1 LVDV	1.2V Low-Voltage Detector Value This bit is the current value of the 1.2V LVD monitor in the PMC 0. For more information see Voltage Monitors . 0b - Low-voltage event was not detected by the 1.2V LVD monitor in the PMC 0 1b - Low-voltage event was detected by the 1.2V LVD monitor in the PMC 0
0 LVDF	1.2V Low-Voltage Detector Flag This flag is set when a low-voltage event was detected by the 1.2V LVD monitor in the PMC 0. This flag is cleared when 1 is written to the bit LVDACK in the register PMC0_CTRL. For more information see Voltage Monitors . **NOTE:** This bit will only be cleared if LVDV is cleared. 0b - Low-voltage event was not detected by the 1.2V LVD monitor in the PMC 0 1b - Low-voltage event was detected by the 1.2V LVD monitor in the PMC 0

28.4.22 PMC 1 Control register (CTRL)

28.4.22.1 Offset

Register	Offset
CTRL	24h

28.4.22.2 Diagram



28.4.22.3 Fields

Field	Function
31-16 —	Reserved field
15 —	Reserved field
14 ISOACK	Isolation Acknowledge When 1 is written to this bit, the isolation in the pads is released. For more information see Pads Isolation .
13-11 —	Reserved field
10 HVDRE	1.2V High-Voltage Detector Reset Enable For more information see Voltage Monitors . 0b - 1.2V high-voltage detector reset is disabled 1b - 1.2V high-voltage detector reset is enabled
9 —	Reserved field
8 LVDRE	Low-Voltage Detector Reset Enable For more information see Voltage Monitors . **NOTE:** In case PMIC is being used (PMC0_STATUS[PMC1VSRC] = 0) and it will be turned off during VLLS, LVD reset must be disabled prior a VLLS entry. For more information see Turn off PMC 1 regulator in VLLS **NOTE:** In case PMIC is being used (PMC0_STATUS[PMC1VSRC] = 0) and it is needed to decrease the PMIC voltage below the LVD trip point value, this bit should be cleared. 0b - Low-voltage detector reset is disabled 1b - Low-voltage detector reset is enabled

Table continues on the next page...

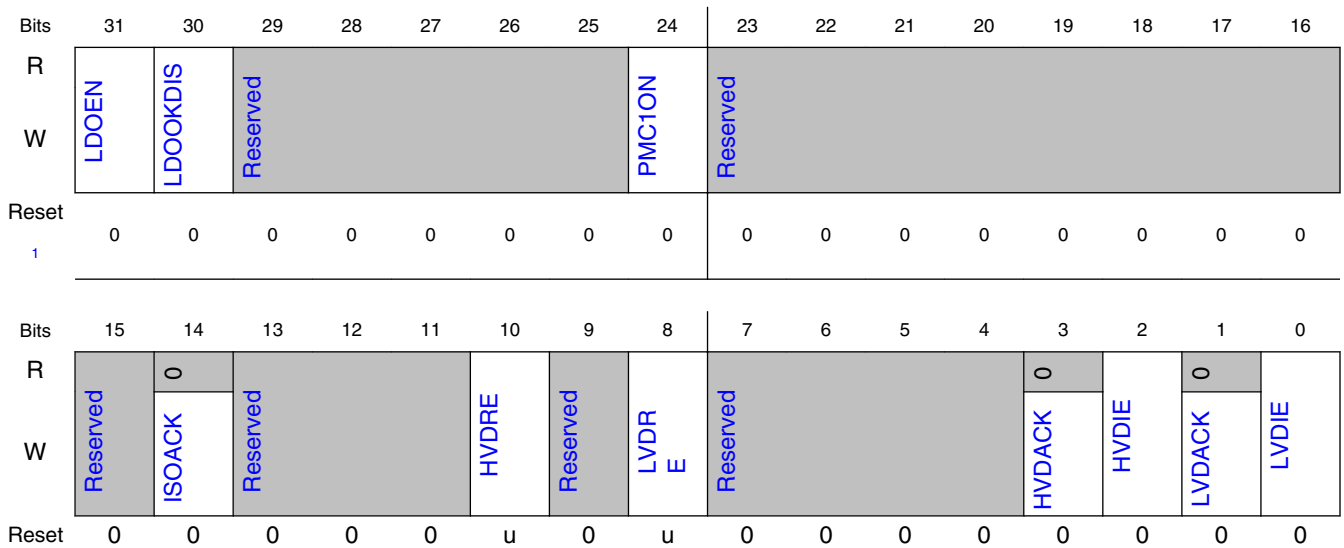
Field	Function
7-4 —	Reserved field
3 HVDACK	1.2V High-Voltage Detector Acknowledge Writing 1 to the bit HVDACK clears the bit HVDF in the register PMC1_STATUS and its interrupt (if the interrupt is enabled).
2 HVDIE	1.2V High-Voltage Detector Interrupt Enable This bit enables the asynchronous and synchronous interrupts by High-Voltage Detector. For more information see Voltage Monitors . **NOTE:** Asynchronous LVD interrupts are masked in VLLS mode. 0b - 1.2V high-voltage detector interrupt is disabled 1b - 1.2V high-voltage detector interrupt is enabled
1 LVDACK	1.2V Low-Voltage Detector Acknowledge Writing 1 to the bit LVDACK clears the bit LVDF in the register PMC1_STATUS and its interrupt (if the interrupt is enabled).
0 LVDIE	Low-Voltage Detector Interrupt Enable This bit enables the asynchronous and synchronous interrupts by Low-Voltage Detector. For more information see Voltage Monitors . **NOTE:** In case PMIC is being used (PMC0_STATUS[PMC1VSRC] = 0) and it will be turned off during VLLS, LVD interrupts must be disabled prior a VLLS entry. For more information see Turn off PMC 1 regulator in VLLS **NOTE:** In case PMIC is being used (PMC0_STATUS[PMC1VSRC] = 0) and it is needed to decrease the PMIC voltage below the LVD trip point value, this bit should be cleared. 0b - Low-voltage detector interrupt is disabled 1b - Low-voltage detector interrupt is enabled

28.4.23 PMC 0 Control register (CTRL)

28.4.23.1 Offset

Register	Offset
CTRL	28h

28.4.23.2 Diagram



1. Undefined reset values will be initialized with fuses values during a POR event.

28.4.23.3 Fields

Field	Function
31 LDOEN	PMC 1 LDO Regulator Enable This bit field selects whether the PMC 1 will be supplied by the internal LDO regulator or a external PMIC during a PMC 1 power up; for more information see Turn on PMC 1 . When a LDO regulator is being used, this bit can also disable the regulator during VLLS; for more information see Turn off PMC 1 regulator in VLLS . **NOTE:** After PMC 1 was powered up using external PMIC, this bitfield value must not be changed (until a POR event). **NOTE:** After PMC 1 was powered up using internal LDO, it can be changed before PMC 1 go to VLLS (to save power), as long as user asserts this bit before PMC 1 exits VLLS. **NOTE:** In case PMC 1 was powered up using internal LDO, this bit must be always kept asserted except in case that PMC 1 is going to be turned-off on a VLLS entry (Turn off PMC 1 regulator in VLLS). 0b - PMC 1 LDO Regulator is disabled. 1b - PMC 1 LDO Regulator is enabled.
30 LDOOKDIS	Disable to Wait LDO OK Signal This bitfield selects whether or not the PMC will check the regulated output from the LDO Regulator during a mode transition, PMC 1 power-up or a voltage level change (PMC 1 in RUN/HSRUN mode). **NOTE:** This bit must keep stable in the middle of a mode transition (including PMC 1 power-up). **NOTE:** In a LDO voltage level change during the PMC 1 RUN/HSRUN mode, this bit must keep stable until LDOVLF (PMC1_STATUS) be cleared. 0b - The PMC will wait for the OK signal from LDO Regulator 1b - The PMC will not wait for the OK signal from LDO Regulator

Table continues on the next page...

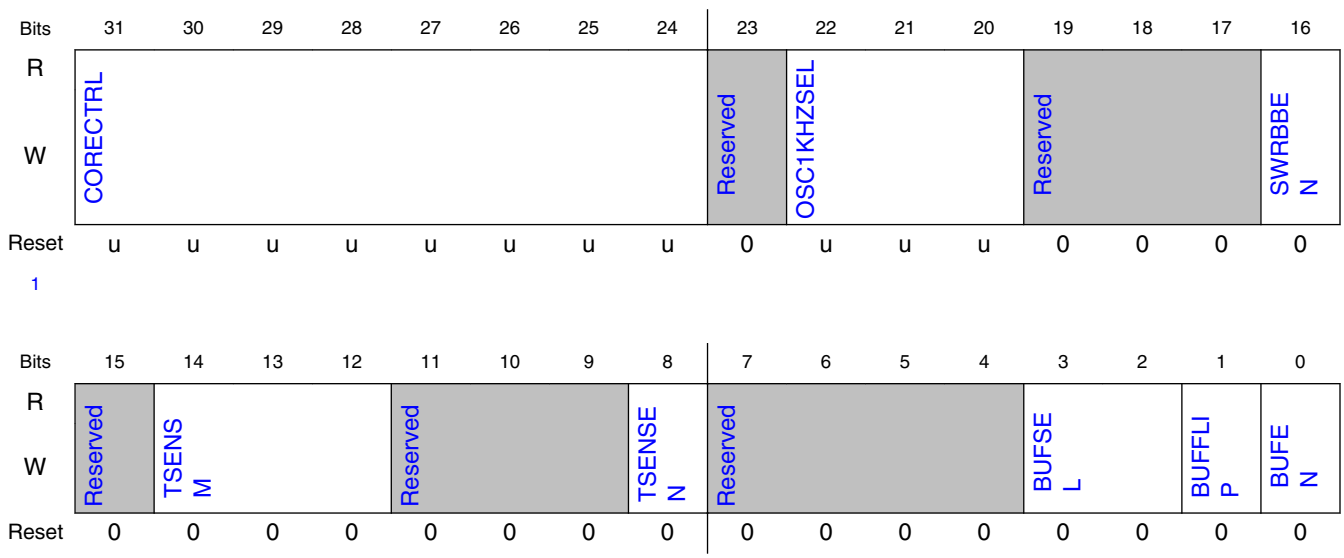
Field	Function
29-25 —	Reserved field
24 PMC1ON	PMC 1 Power On When this bit field is asserted the PMC 1 is powered on. This bit would take action only once. For more information see Turn on PMC 1 . **NOTE:** This bit will be rearmed after a POR event only.
23-15 —	Reserved field
14 ISOACK	Isolation Acknowledge When 1 is written to this bit the isolation in the PADS is released. For more information see Pads Isolation .
13-11 —	Reserved field
10 HVDRE	1.8V High-Voltage Detector Reset Enable For more information see Voltage Monitors . 0b - 1.8V high-voltage detector reset is disabled 1b - 1.8V high-voltage detector reset is enabled
9 —	Reserved field
8 LVDRE	1.2V Low-Voltage Detector Reset Enable For more information see Voltage Monitors . 0b - 1.2V low-voltage detector reset is disabled 1b - 1.2V low-voltage detector reset is enabled
7-4 —	Reserved field
3 HVDACK	1.8V High-Voltage Detector Acknowledge Writing 1 to the bit HVDACK clears the bit HVDF in the register PMC0_STATUS and its interrupt (if the interrupt is enabled).
2 HVDIE	1.8V High-Voltage Detector Interrupt Enable This bit enables the asynchronous and synchronous interrupts by High-Voltage Detector. For more information see Voltage Monitors . 0b - 1.8V high-voltage detector interrupt is disabled 1b - 1.8V high-voltage detector interrupt is enabled
1 LVDACK	1.2V Low-Voltage Detector Acknowledge Writing 1 to the bit LVDACK clears the bit LVDF in the register PMC0_STATUS and its interrupt (if the interrupt is enabled).
0 LVDIE	1.2V Low-Voltage Detector Interrupt Enable This bit enables the asynchronous and synchronous interrupts by Low-Voltage Detector. For more information see Voltage Monitors . 0b - 1.2V low-voltage detector interrupt is disabled 1b - 1.2V low-voltage detector interrupt is enabled

28.4.24 PMC 0 Analog Core Control register (ACTRL)

28.4.24.1 Offset

Register	Offset
ACTRL	30h

28.4.24.2 Diagram



1. Undefined reset values will be initialized with fuses values during a POR event.

28.4.24.3 Fields

Field	Function
31-24 CORECTRL	Controls to Analog PMC Core
23 —	Reserved field
22-20 OSC1KHZSEL	1KHz Oscillator Select
19-17	Reserved field

Table continues on the next page...

Field	Function
—	
16 SWRBBEN	PMC 1 Switch RBB Enable The power switches has a setup to use reverse back bias in its gate and this way reduce the leakage. **NOTE:** Since this circuit when enabled has current consumption it is recommended to use it just in high temperatures when the leakage reduction is much higher than the current consumption. 0b - Switch RBB is disabled. 1b - Switch RBB is enabled.
15 —	Reserved field
14-12 TSENSM	Temperature Sensor Mode Selects the temperature sensor mode.
11-9 —	Reserved field
8 TSENSEN	Temperature Sensor Enable Enables temperature sensor. **NOTE:** The HP bandgap will be kept enabled in any mode when the temperature sensor is enabled. For more information see Bandgap . **NOTE:** This bit must be written in RUN mode only. 0b - The temperature sensor is disabled. 1b - The temperature sensor is enabled.
7-4 —	Reserved field
3-2 BUFSEL	Buffer Selection Selects the buffer output.
1 BUFFLIP	Buffer Flip Flips the buffer input pair for offset cancelling. 0b - Buffer input not flipped. 1b - Buffer input flipped.
0 BUFEN	Buffer Enable Enables the analog buffer. 0b - Analog buffer is disabled. 1b - Analog buffer is enabled.

28.4.25 PMC 1 Biasing Control register (BCTRL)

28.4.25.1 Offset

Register	Offset
BCTRL	34h

28.4.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				FBBPLEVEL				Reserved				FBBNLEVEL			
Reset [1]	0	0	0	0	u	u	u	u	0	0	0	0	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				RBBPLEVEL				Reserved				RBBNLEVEL			
Reset	0	0	0	0	u	u	u	u	0	0	0	0	u	u	u	u

1. Undefined reset values will be initialized with fuses values during a POR event.

28.4.25.3 Fields

Field	Function
31-28 —	Reserved field
27-24 FBBPLEVEL	FBB P-Well Voltage Level Selects the voltage level of the FBB P-Well regulator. Values related to the domain voltage source. Gray coded 50 mV steps. For more information see Enable or Configure FBB . 0000b - No BIAS condition. 0001b - Voltage level at 50mV. 0010b - Voltage level at 150mV. 0011b - Voltage level at 100mV. 0100b - Voltage level at 350mV. 0101b - Voltage level at 300mV. 0110b - Voltage level at 200mV. 0111b - Voltage level at 250mV.
23-20 —	Reserved field
19-16 FBBNLEVEL	FBB N-Well Voltage Level Selects the voltage level of the FBB N-Well regulator. Values related to the domain voltage source. Gray coded 50 mV steps. For more information see Enable or Configure FBB . 0000b - No BIAS condition. 0001b - Voltage level at -50mV. 0010b - Voltage level at -150mV. 0011b - Voltage level at -100mV. 0100b - Voltage level at -350mV. 0101b - Voltage level at -300mV. 0110b - Voltage level at -200mV. 0111b - Voltage level at -250mV.

Table continues on the next page...

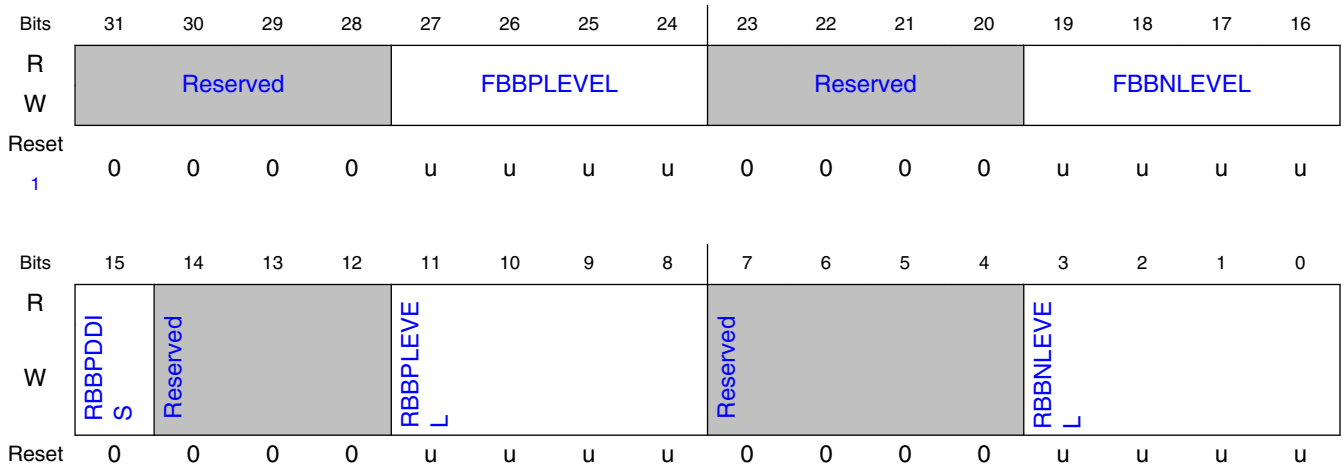
Field	Function
15-12 —	Reserved field
11-8 RBBPLEVEL	RBB P-Well Voltage Level Selects the voltage level of the RBB P-Well regulator. Values related to the domain voltage source. 100 mV steps. For more information see Enable or Configure RBB . 0000b - Voltage level at -0.5V. 0001b - Voltage level at -0.6V. 0010b - Voltage level at -0.7V. 0011b - Voltage level at -0.8V. 0100b - Voltage level at -0.9V. 0101b - Voltage level at -1.0V. 0110b - Voltage level at -1.1V. 0111b - Voltage level at -1.2V. 1000b - Voltage level at -1.3V.
7-4 —	Reserved field
3-0 RBBNLEVEL	RBB N-Well Voltage Level Selects the voltage level of the RBB V-Well regulator. Values related to the domain voltage source. 100 mV steps. For more information see Enable or Configure RBB . 0000b - Voltage level at 0.5V. 0001b - Voltage level at 0.6V. 0010b - Voltage level at 0.7V. 0011b - Voltage level at 0.8V. 0100b - Voltage level at 0.9V. 0101b - Voltage level at 1.0V. 0110b - Voltage level at 1.1V. 0111b - Voltage level at 1.2V. 1000b - Voltage level at 1.3V.

28.4.26 PMC 0 Biasing Control register (BCTRL)

28.4.26.1 Offset

Register	Offset
BCTRL	38h

28.4.26.2 Diagram



1. Undefined reset values will be initialized with fuses values during a POR event.

28.4.26.3 Fields

Field	Function
31-28 —	Reserved field
27-24 FBBPLEVEL	FBB P-Well Voltage Level Selects the voltage level of the FBB P-Well regulator. Values related to the domain voltage source. Gray coded 50 mV steps. For more information see Enable or Configure FBB . 0000b - No BIAS condition. 0001b - Voltage level at 50mV. 0010b - Voltage level at 150mV. 0011b - Voltage level at 100mV. 0100b - Voltage level at 350mV. 0101b - Voltage level at 300mV. 0110b - Voltage level at 200mV. 0111b - Voltage level at 250mV.
23-20 —	Reserved field
19-16 FBBNLEVEL	FBB N-Well Voltage Level Selects the voltage level of the FBB N-Well regulator. Values related to the domain voltage source. Gray coded 50 mV steps. For more information see Enable or Configure FBB . 0000b - No BIAS condition. 0001b - Voltage level at -50mV. 0010b - Voltage level at -150mV. 0011b - Voltage level at -100mV. 0100b - Voltage level at -350mV. 0101b - Voltage level at -300mV. 0110b - Voltage level at -200mV.

Table continues on the next page...

Field	Function
	0111b - Voltage level at -250mV.
15 RBBPDDIS	RBB Pull-down Disable Selects if RBB pull-down will be disabled in a transition to VLPR. 0b - RBB pull-down is enabled. 1b - RBB pull-down is disabled.
14-12 —	Reserved field
11-8 RBBPLEVEL	RBB P-Well Voltage Level Selects the voltage level of the RBB P-Well regulator. Values related to the domain voltage source. 100 mV steps. For more information see Enable or Configure RBB . 0000b - Voltage level at -0.5V. 0001b - Voltage level at -0.6V. 0010b - Voltage level at -0.7V. 0011b - Voltage level at -0.8V. 0100b - Voltage level at -0.9V. 0101b - Voltage level at -1.0V. 0110b - Voltage level at -1.1V. 0111b - Voltage level at -1.2V. 1000b - Voltage level at -1.3V.
7-4 —	Reserved field
3-0 RBBNLEVEL	RBB N-Well Voltage Level Selects the voltage level of the RBB N-Well regulator. Values related to the domain voltage source. 100 mV steps. For more information see Enable or Configure RBB . 0000b - Voltage level at 0.5V. 0001b - Voltage level at 0.6V. 0010b - Voltage level at 0.7V. 0011b - Voltage level at 0.8V. 0100b - Voltage level at 0.9V. 0101b - Voltage level at 1.0V. 0110b - Voltage level at 1.1V. 0111b - Voltage level at 1.2V. 1000b - Voltage level at 1.3V.

28.4.27 PMC 1 SRAMs Control register (SRAMCTRL)

28.4.27.1 Offset

Register	Offset
SRAMCTRL	44h

28.4.27.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								SRAM_STDY							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.27.3 Fields

Field	Function
31-8 —	Reserved field
7-0 SRAM_STDY	PMC 1 SRAM Bank in Standby Mode The bit i controls the PMC 1 SRAM bank i. SRAM_STDY[i] = 0 — PMC 1 SRAM bank i is in RETENTION mode during LLS power mode, in POWER_DOWN mode during VLLS power mode, or in RUN mode during any other power mode. SRAM_STDY[i] = 1 — PMC 1 SRAM bank i is in STANDBY mode during any power mode except VLLS and LLS; in RETENTION mode during LLS power mode ; or in POWER_DOWN mode during VLLS power mode. For more information see SRAM Banks . **NOTE:** This field will be reset by a PMC 1's domain warm reset.

28.4.28 PMC 0 SRAMs Control 0 register (SRAMCTRL_0)

28.4.28.1 Offset

Register	Offset
SRAMCTRL_0	48h

28.4.28.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SRAM_PD				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.28.3 Fields

Field	Function
31-5 —	Reserved field
4-0 SRAM_PD	PMC 0 SRAM Bank Power Down The bit i controls the power mode of the PMC 0 SRAM bank i. PMC0_SRAM_PD[i] = 0 — PMC 0 SRAM bank i is in RETENTION mode during LLS power mode, in DEEP_SLEEP mode during VLLS power mode, or in RUN mode during any other power mode. PMC0_SRAM_PD[i] = 1 — PMC 0 SRAM bank i is in ASD or ARRAY_SHUTDOWN during any power mode except VLLS; or in POWER_DOWN mode during VLLS power mode. For more information see SRAM Banks . **NOTE:** The field PMC0_SRAM_PD[i] has higher priority over the fields PMC0_SRAM_PDS[i] (in the register PMC0_SRAMCTRL_1) and PMC0_SRAM_STDY[i] (in the register PMC0_SRAMCTRL_2). **NOTE:** This bitfield must be cleared before a VLLS entry with Array Regulator turned off.

28.4.29 PMC 0 SRAMs Control 1 register (SRAMCTRL_1)

28.4.29.1 Offset

Register	Offset
SRAMCTRL_1	4Ch

28.4.29.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SRAM_PDS				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.29.3 Fields

Field	Function
31-5 —	Reserved field
4-0 SRAM_PDS	PMC 0 SRAM Bank Power Down in Stop Modes The bit i controls the power mode of the PMC 0 SRAM bank i. PMC0_SRAM_PDS[i] = 0 — PMC 0 SRAM bank i is in RETENTION mode during LLS power mode, in DEEP_SLEEP mode during VLLS power mode, or in RUN mode during any other power mode. PMC0_SRAM_PDS[i] = 1 — PMC 0 SRAM bank i is in ASD or ARRAY_SHUTDOWN power mode during STOP , LLS or VLPS modes; in RUN SRAM mode during RUN, VLPR or HSRUN power modes; or in POWER_DOWN mode during VLLS power mode. For more information see SRAM Banks . **NOTE:** The field PMC0_SRAM_PDS[i] has higher priority over the field PMC0_SRAM_STDY[i] (in the register PMC0_SRAMCTRL_2). **NOTE:** This bitfield must be cleared before a VLLS entry with Array Regulator turned off.

28.4.30 PMC 0 SRAMs Control 2 register (SRAMCTRL_2)

28.4.30.1 Offset

Register	Offset
SRAMCTRL_2	50h

28.4.30.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											SRAM_STDY				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.4.30.3 Fields

Field	Function
31-5 —	Reserved field
4-0 SRAM_STDY	PMC 0 SRAM Bank in Standby Mode The bit i controls the PMC 0 SRAM bank i. PMC0_SRAM_STDY[i] = 0 — PMC 0 SRAM bank i is in RETENTION mode during LLS power mode, in DEEP_SLEEP mode during VLLS power mode, or in RUN mode during any other power mode. PMC0_SRAM_STDY[i] = 1 — PMC 0 SRAM bank i is in STANDBY mode during any power mode except VLLS and LLS; in RETENTION mode during LLS power mode ; or in DEEP_SLEEP mode during VLLS power mode. For more information see SRAM Banks . **NOTE:** This field will be reset by a PMC 0's domain warm reset.

28.5 Functional Description

28.5.1 Power-on Reset

The digital and analog PMC supply is constantly monitored by a 1.8V Power-On Reset (POR) monitor. This monitor is always enabled and if the supply is out of the range (around 1.8V) then a power-on reset is generated.

After a POR event, only the PMC 0 domain will be turned on, the PMC 1 domain should be turned on by the procedure described in Procedures.

28.5.2 Regulators

28.5.2.1 Core and Array Regulator

The PMC 0 has the following regulators:

- Core Regulator, supplies the PMC 0 SoG and the periphery of the PMC 0 SRAM memories. It can operate in High Power (HP) or Low Power (LP) modes.
- Array Regulator, supplies the array of the PMC 0 SRAM memories. It can operate in High Power (HP) and Low Power (LP) modes.

The mode of the regulator depends on the PMC 0 mode and the COREREG or ARRAYREG fields in the PMC 0 Registers as shown in Table 28-5. In the RUN, HSRUN and STOP modes, both regulators are in HP mode. In VLLS, the Core Regulator is turned off and the Array Regulator can be turned off optionally. After a POR event, the PMC 0 power mode is in RUN mode.

NOTE

In case the Array Regulator will be turned off, the content of all the memories will be lost.

Table 28-5. PMC 0 regulators

PMC 0 Mode	Core Regulator Mode (COREREG)	Array Regulator Mode (ARRAYREG)	Core Regulator Voltage Level (COREREGVL) *	Array Regulator Voltage Level *
HSRUN (High Speed Run)	HP	HP	0.90V	1.0V
RUN	HP	HP	0.90V	1.0V
VLPR (Very Low Power Run	HP/LP*	HP/LP*	0.70V	1.0V
STOP	HP	HP	0.90V	1.0V
VLPS (Very Low Power Stop)	HP/LP*	HP/LP*	0.70V	1.0V
LLS (Low Leakage Stop)	HP/LP*	HP/LP*	0.70V	0.75V
VLLS (Very Low Leakage Stop)	OFF	HP/LP*/OFF	-	0.75V

In Table 28-5, the cells marked with * are the default values after a POR event.

The level voltage of the Core Regulator can be changed by mode through the COREREGVL field. To change this value in RUN mode see Core Regulator (PMC 0). In the Array Regulator case, the voltage level will be changed automatically to predefined values as shown in Table 28-5.

28.5.2.2 LDO Regulator

The PMC 1 is turned on by the PMC 0 (see Procedures). The PMC 1 can use the internal LDO Regulator or the external PMIC to supply the PMC 1 SoG and SRAMs.

The LDO Regulator will be turned on in the PMC 1 initialization sequence only when the bit LDOEN in PMC 0 registers is asserted (see Using internal LDO regulator). This regulator can operate in High Power (HP) and Low Power (LP) modes. The operation mode depends on the PMC 1 power mode and the LDOM field in the PMC 1 registers as shown in Table 28-6.

The LDO Regulator can be turned off in a mode transition to VLLS by the procedure described in Using internal LDO regulator.

The bit field LDOLV in PMC 1 registers selects the voltage level generated by the LDO regulator, see LDO Regulator (PMC 1) to change the voltage level value in RUN mode.

Table 28-6. PMC 1 regulators

PMC 1 Mode	LDO Regulator Mode (LDOM)	LDO Regulator Voltage Level (LDOVL) *
HSRUN (High Speed Run)	HP	1.0V
RUN	HP	1.0V
VLPR (Very Low Power Run	HP/LP*	0.70V
STOP	HP	1.0V
VLPS (Very Low Power Stop)	HP/LP*	0.70V
LLS (Low Leakage Stop)	HP/LP*	0.70V
VLLS (Very Low Leakage Stop)	HP/LP*	The same as RUN

In Table 28-6, the cells marked with * are the default values after a POR event.

NOTE

In VLLS mode of the PMC 1, althought the LDO regulator is turned on, the PMC 1 SoG and SRAMs aren't supplied, due to the internal supply switches are opened.

28.5.3 Voltage Monitors

The PMC 0 has the following voltage monitors:

- 1.2V Low Voltage Detector (LVD);
- 1.8V High Voltage Detector (HVD).

The LVD can operate in High Power (HP) or Low Power (LP) modes and can not be disabled, but the HVD can operate in HP only and can be disabled. The operation mode depends on PMC 0 mode and the fields MONLVD and MONHVD in the PMC 0 registers as shown in Table 28-7.

The sense point of the LVD/HVD is in the supply of the Core regulator.

These monitors can be configured to generate resets (PMC0_CTRL register - bits LVDRE and HVDRE), asynchronous and synchronous interrupts (PMC0_CTRL register - bits LVDIE and HVDIE) or none of them.

Table 28-7. PMC 0 monitors

PMC 0 Mode	LVD Mode (MONLVD)	HVD Mode (MONHVD)
HSRUN (High Speed Run)	HP	HP
RUN	HP	HP
VLPR (Very Low Power Run	HP/LP*	HP/OFF*
STOP	HP	HP
VLPS (Very Low Power Stop)	HP/LP*	HP/OFF*
LLS (Low Leakage Stop)	HP/LP*	HP/OFF*
VLLS (Very Low Leakage Stop)	HP/LP*	HP/OFF*

In Table 28-7, the cells marked with * are the default values after a POR event.

The PMC 1 has the following voltage monitors:

- Low Voltage Detector (LVD);
- 1.2V High Voltage Detector (HVD).

Like PMC 0, the LVD can operate in High Power (HP) or Low Power (LP) modes and can not be disabled, but the HVD can operate in HP only and can be disabled. After a POR event, the LVD is in Low Power mode and the HVD is disabled. After PMC 1 power up, the monitors operation mode depends on the PMC 1 mode and the fields MONLVD and MONHVD in the PMC 1 registers as is shown in Table 28-8.

The sense point of the LVD/HVD is configurable, it can be placed at the supply of the LDO regulator or at the pin connected to the external PMIC. This sense point is configured by the LDOEN bit (in PMC0_CTRL register) during a PMC1 power up, then this configuration can be changed after a POR event only. When the PMC 1 is powered up using the internal LDO (LDOEN=1, see Using internal LDO regulator), the sense

point is placed at the supply of the LDO regulator; otherwise, when the PMC 1 is powered up with the external PMIC (LDOEN=0, see Using external PMIC), the sense point is placed at the pin connected to the external PMIC. The actual sense point configured is indicated by the PMC1VSRC (in PMC0_STATUS register).

When the sense point is placed at the supply of the LDO regulator, the trip point of the LVD is around 1.2V.

These monitors can be configured to generate reset (PMC1_CTRL register - bits LVDRE and HVDRE), asynchronous and synchronous interrupts (PMC1_CTRL register - bits LVDIE and HVDIE) or none of them.

Table 28-8. PMC 1 monitors

PMC 1 Mode	LVD Mode (MONLVD)	HVD Mode (MONHVD)
HSRUN (High Speed Run)	HP	HP
RUN	HP	HP
VLPR (Very Low Power Run	HP/LP*	HP/OFF*
STOP	HP	HP
VLPS (Very Low Power Stop)	HP/LP*	HP/OFF*
LLS (Low Leakage Stop)	HP/LP*	HP/OFF*
VLLS (Very Low Leakage Stop)	HP/LP*	HP/OFF*

In Table 28-8, the cells marked with * are the default values after a POR event.

28.5.4 Biasing

The PMC 0 and PMC 1 have Forward Back Bias (FBB) and Reverse Back Bias (RBB) functionalities. RBB provides core leakage and total SoC consumption reduction. FBB provides SoC core performance enhancement by increasing device speed (at a cost of higher leakage consumption). RBB and FBB functionalities are not available at same time in the same PMC domain. The FBB could be enabled in the HSRUN mode only. The RBB can be enabled in low power modes only (VLPR, LLS and VLPS) as shown in Table 28-9.

Table 28-9. PMC Biasing

PMC 0/1 Mode	Forward Back Bias	Reverse Back Bias
HSRUN (High Speed Run)	ON/OFF*	OFF
RUN	OFF	OFF
VLPR (Very Low Power Run)	OFF	ON/OFF*
STOP	OFF	OFF

Table continues on the next page...

Table 28-9. PMC Biasing (continued)

PMC 0/1 Mode	Forward Back Bias	Reverse Back Bias
VLPS (Very Low Power Stop)	OFF	ON/OFF*
LLS (Low Leakage Stop)	OFF	ON/OFF*
VLLS (Very Low Leakage Stop)	OFF	OFF

In Table 28-9, the cells marked with * are the default values after a POR event.

NOTE

RBB is not avaliable in the PMC 1's VLPR mode.

28.5.5 Bandgap

The PMC has a HP and a LP bandgap to provide voltage references to analog blocks. The LP bandgap is always turned on. The HP bandgap will be enabled when the Core Regulator is in HP mode or the Array Regulator is in HP mode or the LDO Regulator is turned on or the Temperature Sensor is enabled or the FBB is enabled or some monitor is in HP or the FBGHP (PMC 0 registers) is asserted in the current mode. In other cases the HP bandgap will be disabled to reduce the power consumption i.e. the references will be provided by the LP bandgap only.

28.5.6 SRAM Banks

The PMC controls the power modes of the SRAM banks. The power modes are shown in Table 28-10 and are described in the following:

- RUN mode is the default power mode, the periphery and array are supplied with the operation voltage. Memory accesses are only allowed in this mode.
- In STANDBY mode, the periphery power consumption is reduced. The array is turned on at the operation voltage.
- In ASD or ARRAY-SHUTDOWN mode, the array is shutdown. The periphery is turned on at the operation voltage.
- In RETENTION mode, the periphery and array are supplied with the retention voltage.
- In DEEP_SLEEP mode, the periphery is shutdown and the array is supplied with the retention voltage.
- In POWER_DOWN mode, the periphery and array are shutdown.

Table 28-10. Power Modes

SRAM Mode	Periphery State	Array State
RUN	ON at Operation Voltage	ON at Operation Voltage
STANDBY	ON at Operation Voltage	ON at Operation Voltage
ASD	ON at Operation Voltage	OFF
RETENTION	ON at Retention Voltage	ON at Retention Voltage
DEEP_SLEEP	OFF	ON at Retention Voltage
POWER_DOWN	OFF	OFF

It is important to remark that the ASD mode has higher priority over other modes and STANDBY has priority over the RUN mode.

In the PMC 0 domain there are two memory groups: TCM (Tightly-Coupled Memories) and No-TCM. The PMC 0 supports the following power modes for the TCM: RUN, ASD, STANDBY, RETENTION and POWER_DOWN as shown in Table 28-11.

The PMC0_SRAM_PD[i] bit (PMC0_SRAMCTRL_0) or the PMC0_SRAM_PDS[i] (PMC0_SRAMCTRL_1) should be used to the bank 'i' enter to ASD mode. The PMC0_SRAM_STDY[i] bit (PMC0_SRAMCTRL_2) should be used to the bank 'i' enter to STANDBY mode.

Table 28-11. PMC 0 TCM

PMC 0 Mode	SRAM Modes (PMC0_SRAM_PD[i] , PMC0_SRAM_PDS[i] and PMC0_SRAM_STDY[i])
HSRUN (High Speed Run)	RUN*/STANDBY/ASD
RUN	RUN*/STANDBY/ASD
VLPR (Very Low Power Run)	RUN*/STANDBY/ASD
STOP	RUN*/STANDBY/ASD
VLPS (Very Low Power Stop)	RUN*/STANDBY/ASD
LLS (Low Leakage Stop)	RETENTION*/ASD
VLLS (Very Low Leakage Stop)	DEEP_SLEEP*/POWER_DOWN

In Table 28-11 the cells marked with * are the default values after a POR event.

The No-TCM group only supports RUN, RETENTION and POWER_DOWN modes as shown in Table 28-12.

Table 28-12. PMC 0 No-TCM

PMC 0 Mode	SRAM Modes
HSRUN (High Speed Run)	RUN
RUN	RUN

Table continues on the next page...

Table 28-12. PMC 0 No-TCM (continued)

PMC 0 Mode	SRAM Modes
VLPR (Very Low Power Run)	RUN
STOP	RUN
VLPS (Very Low Power Stop)	RUN
LLS (Low Leakage Stop)	RETENTION
VLLS (Very Low Leakage Stop)	POWER_DOWN

NOTE

In the PMC 0, all SRAMs will be in POWER_DOWN when the Array Regulator is turned off (ARRAYREG bitfield in the PMC0_VLLS register).

Since the Array Regulator supplies only to the PMC 0 SRAM's arrays, in the PMC 1, the array and periphery share the same supply. The PMC 1 supports four power modes: RUN, STANDBY , RETENTION and POWER_DOWN as shown in Table 28-13.

Table 28-13. PMC 1 SRAMs

PMC 1 Mode	SRAM Modes (PMC1_SRAM_STDY[i])
HSRUN (High Speed Run)	RUN*/STANDBY
RUN	RUN*/STANDBY
VLPR (Very Low Power Run)	RUN*/STANDBY
STOP	RUN*/STANDBY
VLPS (Very Low Power Stop)	RUN*/STANDBY
LLS (Low Leakage Stop)	RETENTION
VLLS (Very Low Leakage Stop)	POWER_DOWN

In Table 28-13, the cells marked with * are the default values after a POR event.

28.5.7 Pads Isolation

Some pads are isolated during LLS/VLLS mode entry. On LLS mode exit, the isolation in pads is released automatically. On VLLS mode exit, the isolation in pads in each power domain (PMC 0 or PMC 1) can be released by writing 1 in the respective ISOACK bit (PMC0_CTRL or PMC1_CTRL registers) or by the respective system warm reset.

28.5.8 System Reset

On a system reset during HSRUN mode or some Stop mode, it is expected that the MSMC will start an exit sequence to RUN mode. During this transition the respective regulator voltage value will be adjusted to its POR value.

28.5.9 Procedures

28.5.9.1 Turn on PMC 1

NOTE

The PMC 1 is only once configured during a PMC 1 power up for using either the internal LDO or the external PMIC. This configuration can be changed only after a POR event.

28.5.9.1.1 Using internal LDO regulator

After a POR event, when the PMC 0 is running in RUN mode and the PMC 1 is turned off, the process to turn on the PMC 1 using the internal LDO regulator is as follows:

- Assert the LDOEN bit (PMC0_CTRL).
- Assert the LDOOKDIS bit (PMC0_CTRL) if required.
- Assert the PMC1ON bit (PMC0_CTRL).

NOTE

The LDOOKDIS and LDOEN bitfields must be kept stable until the PMC1 finishes its RUN mode entry.

28.5.9.1.2 Using external PMIC

After a POR event, when the PMC 0 is running in RUN mode and the PMC 1 is turned off, the process to turn on the PMC 1 using the external PMIC is as follows:

- Clear the LDOEN bit (PMC0_CTRL).
- Assert the PMC1ON bit (PMC0_CTRL).

NOTE

The LDOEN bitfield must be kept cleared until PMC 1 finishes its entry to RUN mode.

28.5.9.2 Turn off PMC 1 regulator in VLLS

28.5.9.2.1 Using internal LDO regulator

When the internal LDO regulator is being used (PMC1VSRC=0) and the PMC 1 is running in RUN mode, the LDO Regulator can be programmed to be turned off in the next transition from RUN to VLLS power mode. The procedure is:

- Clear the LDOEN bit (PMC0_CTRL).
- Execute the command to transition the PMC 1 to VLLS mode (see MSMC).

NOTE

The LDO Regulator is not turned off immediately. It is turned off only when a RUN to VLLS power mode transition is requested.

28.5.9.2.2 Using external PMIC

When the external PMIC is being used (PMC1VSRC=1) , it is possible to turn off the PMIC when the PMC1 is in VLLS mode. The procedure is:

- Clear the LVDIE and LVDRE bits (PMC1_CTRL).
- Execute the command to transition the PMC 1 to VLLS mode (see MSMC).
- Turn off the PMIC (see PMIC specification).

NOTE

The PMIC must be turned off only when the PMC 1 domain is effectively in VLLS mode.

28.5.9.3 Turn on PMC 1 regulator in a VLLS wake-up

28.5.9.3.1 Using internal LDO regulator

When the internal LDO regulator is being used (PMC1VSRC=0), the PMC 1 is running in VLLS mode and the LDO regulator is turned off; the LDO regulator must be turned on in a transition to RUN mode following the next procedure:

- Execute the command to transition the PMC 1 to RUN mode (see MSMC).
- Once in RUN mode, assert the LDOEN bit (PMC0_CTRL).

NOTE

This procedure is strictly neccesary to transition back PMC 1 to RUN mode.

NOTE

The LDOOKDIS and LDOEN bitfields must be kept stable until the PMC1 finishes its entry to RUN mode (see MSMC).

28.5.9.3.2 Using external PMIC

When the external PMIC is being used (PMC1VSRC=1), the PMC 1 is running in VLLS mode and the PMIC is turned off; the PMIC must be turned on following the next procedure:

- Turn on the PMIC (see PMIC specification).
- Execute the command to transition the PMC 1 to RUN mode (see MSMC).

NOTE

This procedure is strictly neccesary to transition back PMC 1 to RUN mode.

28.5.9.4 Change regulator voltage in RUN or HSRUN

28.5.9.4.1 Core Regulator (PMC 0)

The procedure to change the Core Regulator voltage level when the PMC 0 is in RUN mode is:

- Confirm COREVLF (PMC0_STATUS) is cleared.
- Change the value of the COREREGVL bitfield (PMC0_RUN).
- Wait for COREVLF (PMC0_STATUS) to be cleared.

The procedure to change the Core Regulator voltage level when the PMC 0 is in HSRUN mode is:

- Confirm COREVLF (PMC0_STATUS) is cleared.
- Change the value of the COREREGVL bitfield (PMC0_HSRUN).
- Wait for COREVLF (PMC0_STATUS) to be cleared.

28.5.9.4.2 LDO Regulator (PMC 1)

The procedure to change the LDO Regulator voltage level when the PMC 1 is in RUN mode is:

- Confirm LDOVLF (PMC1_STATUS) is cleared.
- Assert the LDOOKDIS bit (PMC0_CTRL) if required.
- Change the value of the LDOVL bitfield (PMC1_RUN).
- Wait for LDOVLF (PMC1_STATUS) to be cleared.

The procedure to change the LDO Regulator voltage level when the PMC 1 is in HSRUN mode is:

- Confirm LDOVLF (PMC1_STATUS) is cleared.
- Assert the LDOOKDIS bit (PMC0_CTRL) if required.
- Change the value of the LDOVL bitfield (PMC1_HSRUN).
- Wait for LDOVLF (PMC1_STATUS) to be cleared.

NOTE

The LDOOKDIS bit must be kept stable until the LDOVLF bitfield be cleared.

28.5.9.5 Change SRAMs power mode in RUN

28.5.9.5.1 PMC 0 SRAMs

The procedure to change the SRAMs power mode during the PMC 0 RUN mode is:

- Confirm SRAMF (PMC0_STATUS) is cleared.
- Change the value of PMC0_SRAM_PD (PMC0_SRAMCTRL_0) or PMC0_SRAM_STDY (PMC0_SRAMCTRL_2).
- Wait for SRAMF (PMC0_STATUS) to be cleared.

28.5.9.5.2 PMC 1 SRAMs

The procedure to change the SRAMs power mode during the PMC 1 RUN mode is:

- Confirm SRAMF (PMC1_STATUS) is cleared.
- Change the value of the PMC1_SRAM_STDY bitfield (PMC1_SRAMCTRL).
- Wait for SRAMF (PMC1_STATUS) to be cleared.

28.5.9.6 Enable or Configure RBB

To enable or configure RBB, i.e. to change the voltage level or enable/disable pull-down, the procedure is following:

- Go to RUN mode.
- Change the value of RBBNLEVEL or RBBPLEVEL bitfields in PMC0_BCTRL register if required.
- Assert RBBPDDIS bitfield in PMC0_BCTRL register if required.
- Assert RBBEN bit of the respective low power mode register (PMCx_VLPR, PMCx_LLS or PMCx_VLPS) if it has not been not asserted previously.
- Go to the required low power mode (VLPR, LLS or VPLS).

The RBB regulators will be enabled in the mode transition.

28.5.9.7 Enable or Configure FBB

To enable or configure FBB, i.e. to change the voltage level, the procedure is following:

- Go to RUN mode.
- Change the value of FBBNLEVEL or FBBPLEVEL bitfields in PMC0_BCTRL register if required.

- Assert FBBEN bit in the PMCx_HSRUN register if it has not been asserted previously.
- Go to HSRUN mode.

The FBB regulators will be enabled in the mode transition.

Chapter 29
Asynchronous Wakeup Interrupt Controller (AWIC)

29.1 Asynchronous Wake-up Interrupt Controller (AWIC)

When either processor enters deep sleep mode, the power management unit in the system can power down most of that processor domain. When the Asynchronous Wake-up Interrupt Controller (AWIC) detects an interrupt belongs to that processor, it wakes up the processor and restores its state, before the processor can process the interrupt. There is one instance of AWIC used on M4 side.

Table 29-1. AWIC configuration

Parameter	Description
Name	Asynchronous Wakeup Interrupt Controller
Instances	1
Configurable features	NA
Interface speed	NA
External I/O pins	NA

Table 29-2. AWIC wake-up sources

AWIC Channel	Interrupt Sources	NVIC IRQ #
0	Reserved	
1	NMI (RMC/MU_A)	
2	Reserved	0
3	Reserved	
4	DMA0 Channel 0, 4 Transfer Complete	1
5	DMA0 Channel 1, 5 Transfer Complete	2
6	DMA0 Channel 2, 6 Transfer Complete	3
7	DMA0 Channel 3, 7 Transfer Complete	4
8	DMA0 Channel 8, 12 Transfer Complete	5
9	DMA0 Channel 9, 13 Transfer Complete	6

Table continues on the next page...

Table 29-2. AWIC wake-up sources (continued)

AWIC Channel	Interrupt Sources	NVIC IRQ #
10	DMA0 Channel 10, 14 Transfer Complete	7
11	DMA0 Channel 11, 15 Transfer Complete	8
12	DMA0 Channel 16, 20 Transfer Complete	9
13	DMA0 Channel 17, 21 Transfer Complete	10
14	DMA0 Channel 18, 22 Transfer Complete	11
15	DMA0 Channel 19, 23 Transfer Complete	12
16	DMA0 Channel 24, 28 Transfer Complete	13
17	DMA0 Channel 25, 29 Transfer Complete	14
18	DMA0 Channel 26, 30 Transfer Complete	15
19	DMA0 Channel 27, 31 Transfer Complete	16
20	DMA0 Error Interrupt - All Channels	17
21	Reserved	18
22	Reserved	19
23	LLWU0	20
24	Reserved	21
25	MU_A	22
26	Reserved	23
27	Reserved	24
28	Reserved	25
29	WDOG0	26
30	SCG0	27
31	Reserved	28
32	Reserved	29
33	Reserved	30
34	SNVS	31
35	Reserved	32
36	LPIT0	33
37	PMC0	34
38	CMC0	35
39	LPTMR0	36
40	LPTMR1	37
41	TPM0	38

Table continues on the next page...

Table 29-2. AWIC wake-up sources (continued)

AWIC Channel	Interrupt Sources	NVIC IRQ #
42	TPM1	39
43	TPM2	40
44	TPM3	41
45	FlexIO0	42
46	LPI2C0	43
47	LPI2C1	44
48	LPI2C2	45
49	LPI2C3	46
50	SAI0	47
51	SAI1	48
52	LPSPI0	49
53	LPSPI1	50
54	LPUART0	51
55	LPUART1	52
56	LPUART2	53
57	LPUART3	54
58	Reserved	55
59	PCTLA	56
60	PCTLB	57
61	ADC0	58
62	ADC1	59
63	CMP0	60
64	CMP1	61
65	DAC0	62
66	DAC1	63
67	A7 WDOG1	64
68	A7 USB0	65
69	A7 USB1	66
70	Reserved	67
71	A7 WDOG2	68
72	A7 USB PHY	69
73	A7 CMC1	70
74	Reserved	71
75	X-Domain DMA interrupt	72
76	X-Domain Master interrupt	73
77	Reserved	74
78	A7 GPU-3D	75
79	A7 GPU-2D	76
80-127	Reserved	NA

Chapter 30
Multicore System Mode Controller (MSMC)

30.1 Chip-specific MSMC information

Table 30-1. Reference links to related information

Topic	Related module	Reference
Full description	MSMC	MSMC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

30.1.1 Multicore System Mode Controller (MSMC)

NOTE

MSMC, CMC, and RMC are three separate but tightly coupled blocks. The memory map documented in this chapter encompasses features related to all 3 blocks, being actually implemented into the CMC block alone

NOTE

References in this chapter to "Core 0" correspond to the Cortex M4 core and references to "Core 1" correspond to the Cortex A7 core.

Table 30-2. MSMC configuration

Parameter	Description
Name	Multicore System Mode Controller (MSMC)
Instances	1
Configurable features	• SMC0 and SMC1: Supports dual cores and dual power domains

Table continues on the next page...

Table 30-2. MSMC configuration (continued)

Parameter	Description
Interface speed	NA
External I/O pins	NA

30.1.2 LLS mode operation

VLPS and LLS modes work in a very similar way, except that LLS causes the memories to enter RETENTION mode and their supply to be reduced (supply reduction applies to M4 domain only). Besides, pins remain isolated while the domain is in LLS mode.

30.1.3 Power mode status through PMSTAT registers

Power Mode Status register (PMSTAT) registers in SMC0 and SMC1 memory map: With the exception of HSRUN mode, PMSTAT[PMSTAT] changes upon power mode entry request, while a power mode is effectively entered only after that request is acknowledged. For accurate power mode status information,PMC 0 Power Mode Status register (PM_STAT) register in the PMC chapter, can be used.

NOTE

It must be noted that Core1 (CA7) is not able to check Core0 (CM4)’s power mode status through PMC0 whenever the platform clock is gated in the latter’s domain. Under such scenario, Core0’s power mode status is only available via MU.

30.1.4 HSRUN mode restriction

i.MX 7ULP doesn't have on-chip flash programming, so any restriction related to flash programming/erasing for entering HSRUN mode mentioned in the chapter, must be ignored.

30.1.5 SLEEPDEEP mode

While CM4 core supports both Sleep and Deep Sleep low power modes, CA7 core supports Standby low power mode only. On this device, CA7 Standby mode affects the application domain (APD) in a similar way CM4 Deep Sleep acts on real-time domain

(RTD), therefore causing WAIT mode to be not supported by APD. In this sense, all references to SLEEPDEEP=0 in the CA7 (or Core1) context throughout this chapter must be ignored. PSTOP3 can be used to emulate WAIT mode on APD, since both modes work in a very similar manner.

30.1.6 Loss of lock (LOL) reset

LOL faults are not supported on this device. All references to loss of lock (LOL) in SRS[SCG] bitfield (See SCG and SCG) and throughout the chapter should be ignored.

30.1.7 WFE/SEV instructions support

WFE/SEV instructions are not supported in i.MX 7ULP, so any reference to these instructions throughout this document must be ignored.

30.1.8 Power mode trigger conditions on i.MX 7ULP

On this device, power mode transition triggers are described by the following tables.

Table 30-3. Power mode transition triggers for Core 0

Transition #	From	To	Trigger conditions
1	RUN	WAIT	Sleep-now or sleep-on-exit executed with SLEEPDEEP=0 by Core 0. See note.[1]
	WAIT	RUN	Core 0 interrupt or reset[2]
2	RUN	STOP	Core 0 STOPM=STOP[3] and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 0. See note.[1]
	STOP	RUN	Core 0 interrupt or reset.[2]
3	RUN	VLPR	Core 0 RUNM=VLPR.
	VLPR	RUN	Core 0 RUNM=RUN or reset.[2]
4	VLPR	VLPW	Sleep-now or sleep-on-exit executed with SLEEPDEEP=0 by Core 0. See note.[1]
	VLPW	VLPR	Core 0 Interrupt.
5	VLPW	RUN	reset.[2]

Table continues on the next page...

Table 30-3. Power mode transition triggers for Core 0 (continued)

Transition #	From	To	Trigger conditions
6	VLPR	VLPS	Core 0 STOPM=VLPS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 0. See note.[1]
	VLPS	VLPR	Interrupt and VLPS mode was entered directly from VLPR.
7	RUN	VLPS	Core 0 STOPM=VLPS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 0. See note.[1]
	VLPS	RUN	Interrupt and VLPS mode was entered directly from RUN or reset.[2]
8	RUN	VLLS	Core 0 STOPM=VLLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 0. See note.[1]
	VLLS	RUN	LLWU wakeup, NMI (PTA9)[4], or reset[5].
9	VLPR	VLLS	Core 0 STOPM=VLLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 0. See note.[1]
10	RUN	LLS	Core 0 STOPM=LLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 0. See note.[1]
	LLS	RUN	Interrupt and LLS mode was entered directly from RUN or RESET0_b.[2]
11	VLPR	LLS	Core 0 STOPM=LLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 0. See note.[1]
	LLS	VLPR	Interrupt and LLS mode was entered directly from VLPR.
12	RUN	HSRUN	Core 0 RUNM=HSRUN
	HSRUN	RUN	Core 0 RUNM=RUN or reset. [2]

1. If debug is enabled, the core clock remains to support debug.
2. M4 reset sources or A7 reset sources (when A7_TO_M4_RESET_EN fuse is blown).
3. If PMCTRL[STOPM]=STOP and PMCTRL[PSTOPO]!=00, then only a Partial Stop mode is entered instead of STOP
4. SIM_DGO_GP11[NMI_VLLS_WAKEUP_EN] bitfield must be set prior to entering VLLS mode when using NMI pin as (VLLS) wake-up source.
5. RESET0_b pin, or RESET1_b pin (when Core 1 is also in VLLS mode)

Table 30-4. Power mode transition triggers for Core 1

Transition #	From	To	Trigger conditions
1	RUN	WAIT	Sleep-now or sleep-on-exit executed with SLEEPDEEP=0 by Core 1. See note.[1]
	WAIT	RUN	Core 1 interrupt or reset[2]
2	RUN	STOP	Core 1 STOPM=STOP[3] and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 1. See note.[1]
	STOP	RUN	Core 1 interrupt or reset.[2]
3	RUN	VLPR	Core 1 RUNM=VLPR.
	VLPR	RUN	Core 1 RUNM=RUN or reset.[2]
4	VLPR	VLPW	Sleep-now or sleep-on-exit executed with SLEEPDEEP=0 by Core 1. See note.[1]
	VLPW	VLPR	Core 1 Interrupt.
5	VLPW	RUN	reset[2]
6	VLPR	VLPS	Core 1 STOPM=VLPS/STOP and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 1. See note.[1]
	VLPS	VLPR	Interrupt and VLPS mode was entered directly from VLPR.
7	RUN	VLPS	Core 1 STOPM=VLPS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 1. See note.[1]
	VLPS	RUN	Interrupt and VLPS mode was entered directly from RUN or reset.[2]
8	RUN	VLLS	Core 1 STOPM=VLLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 1. See note.[1]
	VLLS	RUN	MU_A NMI/Reset wakeup or reset.[2]
9	VLPR	VLLS	Core 1 STOPM=VLLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 1. See note.[1]
10	RUN	LLS	Core 1 STOPM=LLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 1. See note.[1]

Table continues on the next page...

Table 30-4. Power mode transition triggers for Core 1 (continued)

Transition #	From	To	Trigger conditions
	LLS	RUN	Interrupt and LLS mode was entered directly from RUN or reset[2].
11	VLPR	LLS	Core 1 STOPM=LLS and Sleep-now or sleep-on-exit with SLEEPDEEP=1 executed by Core 1. See note.[1]
	LLS	VLPR	Interrupt and LLS mode was entered directly from VLPR.
12	RUN	HSRUN	Core 1 RUNM=HSRUN
	HSRUN	RUN	Core 1 RUNM=RUN or reset[2].

1. If debug is enabled, the core clock remains to support debug.
2. A7 reset sources or M4 reset sources.
3. If PMCTRL[STOPM]=STOP and PMCTRL[PSTOPO]!=00 for MSMC1, then only a Partial Stop mode is entered instead of STOP

30.1.9 Debug in Low Power modes

It is required that the system provides control so that debugger can have control of the device when entering and exiting low power modes. During low power modes, the clock and power (based on modes) will not be available to multiple components, the following handshake allows the debugger to maintain connection during low power. There are two modes supported:

- Clock gated mode
- Power gated mode.

30.2 Introduction

The Multi-System Mode Controller (MSMC) is responsible for sequencing the MCU into and out of all stop and run power modes.

Specifically, it monitors events to trigger transitions between power modes while controlling the power, clocks, and memories of the MCU to achieve the power consumption and functionality of that mode.

This chapter describes all the available power modes, the sequence followed to enter/exit each mode, and the functionality available while in each of the modes.

The following diagram illustrates the connections of the MSMC with other components in the system that are necessary to sequence the system through all power modes.

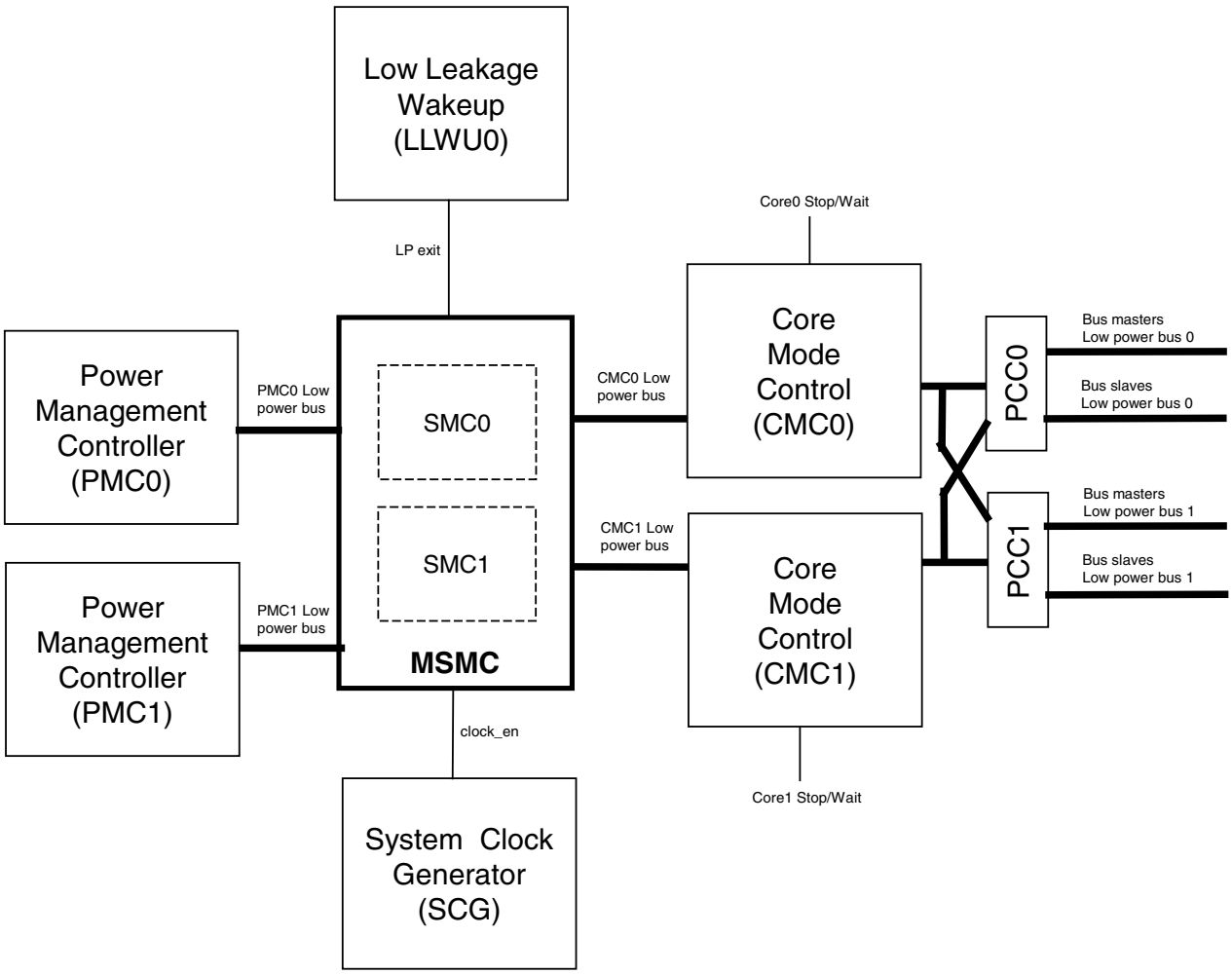


Figure 30-1. Power management system components and connections

30.3 Modes of operation

An Arm Cortex-M CPU has three primary modes of operation:

- Run
- Sleep
- Deep Sleep

An Arm Cortex-A CPU has two primary modes of operation:

- Run
- Standby

The WFI or WFE instruction is used to invoke a Sleep, Deep Sleep or Standby modes for either Cortex family. Run, Wait, and Stop are the common terms used for the primary operating modes of the MSMC.

The following table shows the translation between the Arm CPU modes and MSMC power modes:

Arm CPU mode	MSMC mode
Sleep	Wait
Deep Sleep or Standby	Stop

In addition, the MSMC also augments Stop, Wait, and Run modes in a number of ways. Depending on the needs of the user application, a variety of run and stop modes are available that allow the user to adjust functionality, performance and power consumption.

The following table describes the power modes available to each core within the MCU.

Table 30-5. Power modes

Mode	Description
Run (RUN)	The core can be run at full frequency and all functionality is available. This mode is also referred to as Normal Run mode.
High Speed Run (HSRUN)	Provides the highest performance mode at the fastest supported frequency. The core, system, bus, and flash clock maximum frequencies are unrestricted in this mode.[1]
Wait (WAIT)	Allows a core to enter a static, low power state with instant wakeup time, while still allowing peripherals to operate with full functionality. The core clock is gated off. The system clock continues to operate. Bus clocks, if enabled, continue to operate.
Stop (STOP)	Allows a core and its peripherals to enter a static, low power state with relatively fast wakeup times, while still allowing peripherals to operate with full asynchronous functionality. The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.
Very Low Power Run (VLPR)	Provides the lowest power operating mode to the core at a reduced frequency. The core, system, bus, and flash clock maximum frequencies are restricted in this mode.[1]
Very Low Power Wait (VLPW)	Allows a core to enter a static, very low power state, while still allowing peripherals to operate with full functionality at a reduced frequency. The core clock is gated off. The system, bus, and flash clocks continue to operate, although their maximum frequency is restricted.[1]
Very Low Power Stop (VLPS)	Allows a core and its peripherals to enter a static, very low power state with relatively fast wakeup times, while still allowing peripherals to operate with limited asynchronous functionality. The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.
Low Leakage Stop (LLS)	Allows a core and its peripherals to enter a low leakage, power gated state with relatively fast wakeup times, while still retaining the state of all logic. The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. Logic and I/O are retained, with wakeup only supported via the LLWU, NMI or Reset pins.
Very Low Leakage Stop (VLLS)	Allows a core and its peripherals to enter a very low leakage, power gated state.The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. Logic is power gated, with wakeup only supported via the LLWU, NMI or Reset pins. I/O states are latched.

1. See the devices' chip configuration details for the maximum allowable frequencies.

30.4 Memory map and register descriptions

Information about the registers related to the system mode controller can be found here.

30.4.1 SMC register descriptions

Different MSMC registers reset on different reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

NOTE

The MSMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

NOTE

Before executing the WFI instruction, the last register written to must be read back. This ensures that all register writes associated with setting up the low power mode being entered have completed before the MCU enters the low power mode. Failure to do this may result in the low power mode not being entered correctly.

30.4.1.1 MSMC Memory map

MSMC0 base address: 410A_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_00AAh
4h	Parameter Register (PARAM)	32	RO	0000_0001h
8h	Power Mode Protection register (PMPROT)	32	RW	See description.
10h	Power Mode Control register (PMCTRL)	32	RW	See description.
18h	Power Mode Status register (PMSTAT)	32	RO	0000_0001h
20h	System Reset Status (SRS)	32	RO	0000_0006h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
24h	Reset Pin Control (RPC)	32	RW	0000_0000h
28h	Sticky System Reset Status (SSRS)	32	W1C	0000_0006h
2Ch	System Reset Interrupt Enable (SRIE)	32	RW	0000_0000h
30h	System Reset Interrupt Flag (SRIF)	32	W1C	0000_0000h
34h	Core Software Reset Enable (CSRE)	32	RW	0000_0000h
40h	Mode Register (MR)	32	RO	0000_0000h
60h	Force Mode Register (FM)	32	RW	0000_0000h

30.4.1.2 Version ID Register (VERID)

30.4.1.2.1 Offset

Register	Offset
VERID	0h

30.4.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0

30.4.1.2.3 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.

Table continues on the next page...

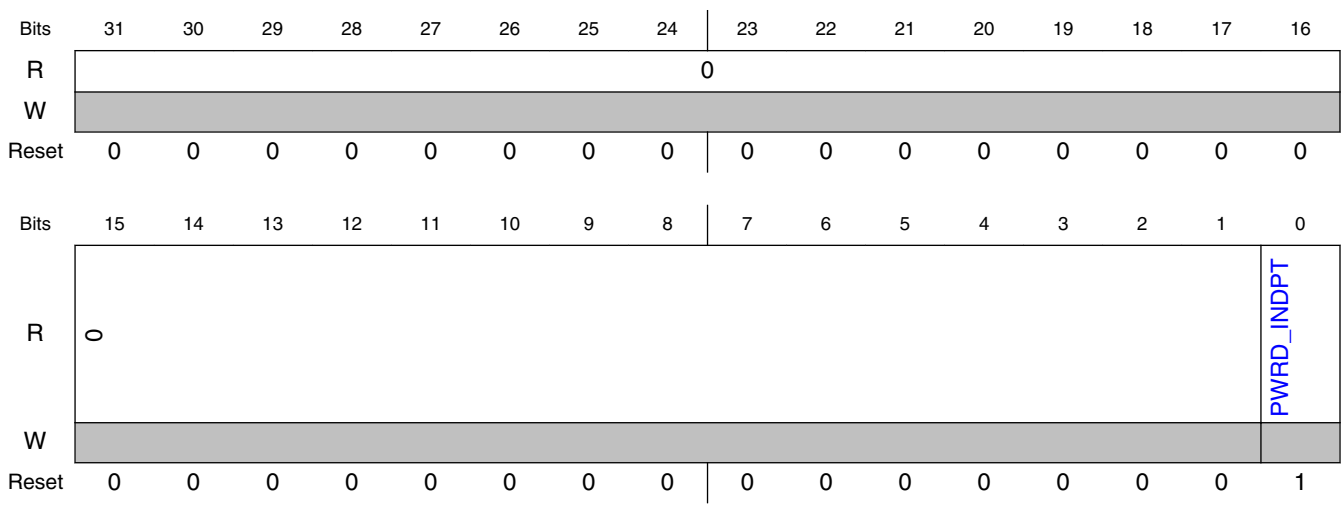
Field	Function
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000010101010b - Default features supported

30.4.1.3 Parameter Register (PARAM)

30.4.1.3.1 Offset

Register	Offset
PARAM	4h

30.4.1.3.2 Diagram



30.4.1.3.3 Fields

Field	Function
31-1 —	Reserved
0 PWRD_INDPT	Power Domains Independent When set, means each core is able to enter power modes independently of the other. When clear, means certain dependancies are enforced by the MSMC before each core is able to enter some power modes. See "Power mode transitions" section for detailed requirements for each core to enter a power mode.

30.4.1.4 Power Mode Protection register (PMPROT)

30.4.1.4.1 Offset

Register	Offset
PMPROT	8h

30.4.1.4.2 Function

This register provides protection for entry into any run or stop power mode. The enabling of the run or stop mode occurs by configuring the Power Mode Control register (PMCTRL).

The PMPROT register can be written only once after any system reset.

If a core is configured for a disallowed or reserved power mode, the core remains in its current power mode. For example, if the core is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and PMCTRL[RUNM] remains 00b, indicating the core is still in Normal Run mode.

NOTE

This register is reset on Chip Reset and by reset sources that trigger Chip Reset (see the Reset section for more information).

30.4.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								AHSRUN	0	AVLP	0	ALLS	0	AVLLS	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	0

1. The reset value is 1 on VLLS wakeup, otherwise it is 0.

30.4.1.4.4 Fields

Field	Function
31-8 —	Reserved
7 AHSRUN	Allow High Speed Run mode When set, allows the PMCTRL[RUNM] field to be written to HSRUN. 0b - HSRUN is not allowed 1b - HSRUN is allowed
6 —	Reserved
5 AVLP	Allow Very-Low-Power Modes When set, allows the PMCTRL[RUNM] field to be written to VLPR and/or the PMCTRL[STOPM] field to be written with VLPS. 0b - VLPR, VLPW, and VLPS are not allowed. 1b - VLPR, VLPW, and VLPS are allowed.
4 —	Reserved
3 ALLS	Allow Low-Leakage Stop Mode When set, allows the PMCTRL[STOPM] field to be written to LLS. 0b - LLS is not allowed 1b - LLS is allowed
2 —	Reserved
1 AVLLS	Allow Very-Low-Leakage Stop Mode When set, allows the PMCTRL[STOPM] field to be written to VLLS. 0b - VLLS mode is not allowed 1b - VLLS mode is allowed
0 —	Reserved

30.4.1.5 Power Mode Control register (PMCTRL)

30.4.1.5.1 Offset

Register	Offset
PMCTRL	10h

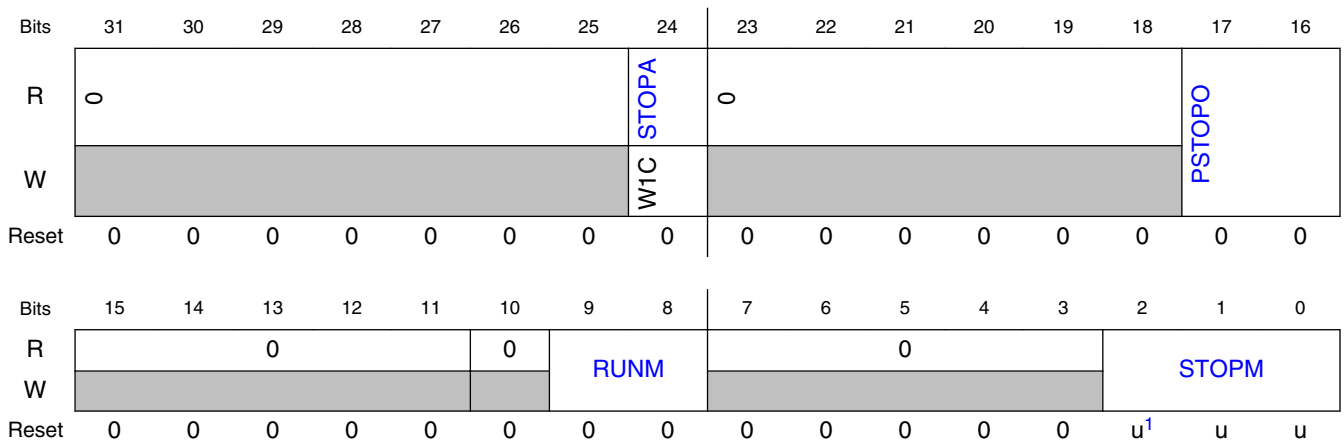
30.4.1.5.2 Function

The PMCTRL register controls entry into low-power Run and Stop modes, provided that the selected power mode is allowed via an appropriate setting of the protection (PMPROT) register.

NOTE

This register is reset on Chip POR and by reset types that trigger Chip POR. See the Reset section details for more information.

30.4.1.5.3 Diagram



1. The reset value is 100 on VLLS wakeup , otherwise it is 000.

30.4.1.5.4 Fields

Field	Function
31-25 —	Reserved
24 STOPA	Stop Abort Flag
23-18 —	Reserved
17-16 PSTOPO	Partial Stop Option These bits control whether a Partial Stop mode is entered when STOPM=STOP. When entering a Partial Stop mode from RUN mode, the PMC and SCG remain fully powered, allowing the device to wakeup almost instantaneously at the expense of higher power consumption. In PSTOP2, only system clocks are gated allowing peripherals running on bus clock to remain fully functional. In PSTOP1, both system and bus clocks are gated. 00b - STOP - Normal Stop mode 01b - PSTOP1 - Partial Stop with system and bus clock disabled

Table continues on the next page...

Field	Function
	10b - PSTOP2 - Partial Stop with system clock disabled and bus clock enabled 11b - PSTOP3 - Partial Stop with system clock enabled and bus clock enabled
15-11 —	Reserved
10 —	Reserved
9-8 RUNM	Run Mode Control When written, causes entry into the selected run mode. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. **NOTE:** Stop mode entry should not be attempted while RUNM=HSRUN or PMSTAT=HSRUN. 00b - Normal Run mode (RUN) 01b - Reserved 10b - Very-Low-Power Run mode (VLPR) 11b - High Speed Run mode (HSRUN)
7-3 —	Reserved
2-0 STOPM	Stop Mode Control Selects the desired stop mode when a core enters deepsleep. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. After any system reset, this field is cleared by hardware on any successful write to the PMPROT register. **NOTE:** When set to STOP, the PSTOPO bits can be used to select a Partial Stop mode if desired. 000b - Normal Stop (STOP) 001b - Reserved 010b - Very-Low-Power Stop (VLPS) 011b - Low-Leakage Stop (LLS) 100b - Very-Low-Leakage Stop (VLLS) 101b - Reserved 110b - Reserved 111b - Reserved

30.4.1.6 Power Mode Status register (PMSTAT)

30.4.1.6.1 Offset

Register	Offset
PMSTAT	18h

30.4.1.6.2 Function

PMSTAT is a read-only, one-hot register which indicates the current power mode of the system.

30.4.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								PMSTAT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

30.4.1.6.4 Fields

Field	Function
31-24 —	Reserved
23-8 —	Reserved
7-0 PMSTAT	Power Mode Status **NOTE:** When debug is enabled, the PMSTAT will not update to STOP or VLPS **NOTE:** When a Partial Stop Option is enabled, the PMSTAT will not update to STOP or VLPS 00000001b - Current power mode is RUN. 00000010b - Current power mode is any STOP mode. 00000100b - Current power mode is VLPR. 10000000b - Current power mode is HSRUN

30.4.1.7 System Reset Status (SRS)

30.4.1.7.1 Offset

Register	Offset
SRS	20h

30.4.1.7.2 Function

This register includes read-only status flags to indicate the source of the most recent reset.

NOTE

The reset state of this register depends on the reset source:

- POR = 0x0000_0006
- LVD = 0x0000_0004
- VLLS Wakeup = 0x0000_0001
- VLLS Reset Wakeup = 0x0000_0111

30.4.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TAMPER	SECVIO	0	JTAG	0			VBAT	0						0	CORE1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCKUP	SW	WDOG	SCG	STOPACK	RSTACK	MDM	PIN	CORE	0	FATAL	WARM	HVD	LVD	POR	WAKEUP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

30.4.1.7.4 Fields

Field	Function
31 TAMPER	Tamper Reset Indicates a reset has been caused by Tamper Detection logic. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by tamper detection. 1b - Reset generated by tamper detection.
30 SECVIO	Security Violation Reset Indicates a reset has been caused by a Security Violation logic. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by security violation. 1b - Reset generated by security violation.
29 —	Reserved
28 JTAG	JTAG System Reset Indicates a reset has been caused by a JTAG system reset request. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by JTAG system reset.

Table continues on the next page...

Field	Function
	1b - Reset generated by JTAG system reset.
27-25 —	Reserved
24 VBAT	VBAT System Reset VBAT POR will generate a system reset. 0b - Reset not generated by VBAT system reset. 1b - Reset generated by VBAT system reset.
23-18 —	Reserved
17 —	Reserved
16 CORE1	Core1 System Reset Indicates the reset source was generated by a Core1 specific reset. 0b - Reset not generated from Core1 system reset source. 1b - Reset generated from Core1 system reset source.
15 LOCKUP	Lockup Reset Indicates a reset has been caused by the Arm core indication of a LOCKUP event. 0b - Reset not generated by core lockup or exception. 1b - Reset generated by core lockup or exception.
14 SW	Software Reset Indicates a reset has been caused by a software reset request from the Arm core (SYSRESETREQ). 0b - Reset not generated by software request from core. 1b - Reset generated by software request from core.
13 WDOG	Watchdog Reset Indicates a reset has been caused by a WatchDog timeout. 0b - Reset is not generated from the WatchDog timeout. 1b - Reset is generated from the WatchDog timeout.
12 SCG	SCG Reset Indicates a reset has been caused by a loss-of-clock or loss-of-lock event in the SCG. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset is not generated from an SCG loss of lock or loss of clock. 1b - Reset is generated from an SCG loss of lock or loss of clock.
11 STOPACK	Stop Timeout Reset Indicates a reset has been caused by a timeout in the Stop mode entry logic. This timeout is generated if a peripheral does not acknowledge entry into any Stop mode within approximately one second. 0b - Reset not generated by Stop Controller Timeout. 1b - Reset generated by Stop Controller Timeout.
10 RSTACK	Reset Timeout Indicates a reset has been caused by a timeout in the system reset generaton logic. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated from Reset Controller Timeout. 1b - Reset generated from Reset Controller Timeout.
9 MDM	MDM Reset Indicates a reset has been caused by a reset request from the Miscellaneous Debug Module (MDM). 0b - Reset was not generated from the MDM reset request. 1b - Reset was generated from the MDM reset request.

Table continues on the next page...

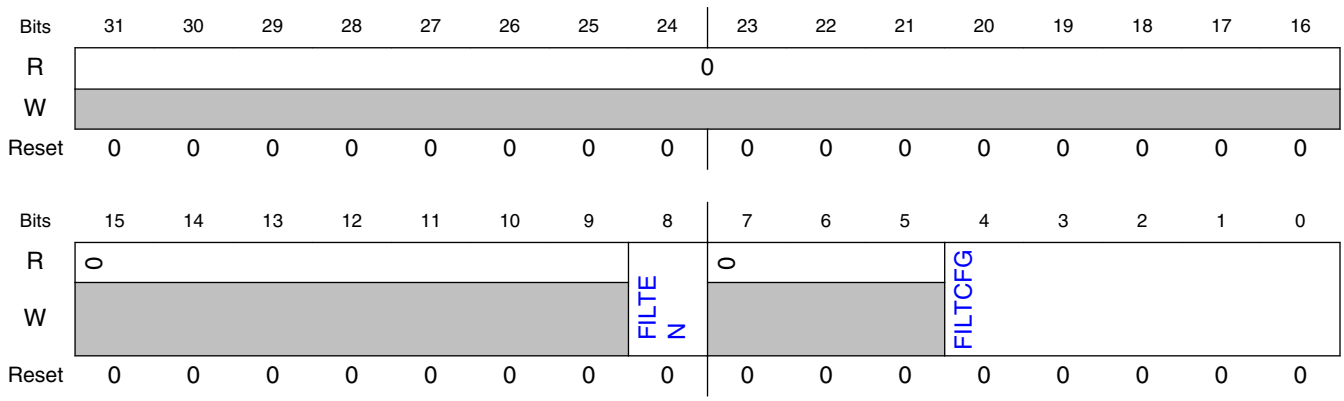
Field	Function
8 PIN	Pin Reset Indicates a reset has been caused by external assertion of the RESET_b pin. 0b - Reset was not generated from the assertion of RESET_B pin. 1b - Reset was generated from the assertion of RESET_B pin.
7 CORE	Core Reset Indicates the last reset was a reset that only reset the core. 0b - Reset source was not core only reset. 1b - Reset source was core reset and reset the core only.
6 —	Reserved
5 FATAL	Fatal Reset Fatal Reset will assert if the system reset source was fatal. SRAM contents cannot be guaranteed following a fatal reset source. 0b - Reset was not generated by a fatal reset source. 1b - Reset was generated by a fatal reset source.
4 WARM	Warm Reset Warm Reset flag will assert whenever any of the system reset sources in this register assert (SRS[31:8]). 0b - Reset not generated by Warm Reset source. 1b - Reset generated by Warm Reset source.
3 HVD	HVD Reset Indicates a reset has been caused by a High Voltage Detect. SRAM contents cannot be guaranteed following a HVD reset source. 0b - Reset not generated by HVD. 1b - Reset generated by HVD.
2 LVD	LVD Reset Indicates a reset has been caused by a Low Voltage Detect. SRAM contents cannot be guaranteed following a LVD reset source. 0b - Reset not generated by LVD. 1b - Reset generated by LVD.
1 POR	POR Reset Indicates a reset has been caused by Power On Reset detection logic. SRAM contents cannot be guaranteed following a POR reset source. 0b - Reset not generated by POR. 1b - Reset generated by POR.
0 WAKEUP	Wakeup Reset Indicates a reset has been caused by a wakeup from VLLS mode. 0b - Reset not generated by wakeup from VLLS mode. 1b - Reset generated by wakeup from VLLS mode.

30.4.1.8 Reset Pin Control (RPC)

30.4.1.8.1 Offset

Register	Offset
RPC	24h

30.4.1.8.2 Diagram



30.4.1.8.3 Fields

Field	Function
31-9 —	Reserved
8 FILTEN	Filter Enable 0b - Slow clock reset pin filter disabled. 1b - Slow clock reset pin filter enabled in Run modes.
7-5 —	Reserved
4-0 FILTCFG	Reset Filter Configuration Configures the width of the reset pin filter from 1 to 32 slow clock cycles.

30.4.1.9 Sticky System Reset Status (SSRS)

30.4.1.9.1 Offset

Register	Offset
SSRS	28h

30.4.1.9.2 Function

The SSRS are only reset on POR/LVD/HVD/WAKEUP and remain set until cleared by software. The SSRS stores all system reset sources that have generated a system reset, since the last POR/LVD/HVD/WAKEUP, that have not been cleared by software. The SSRS does not update following a core software reset.

30.4.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TAMPER	SECVIO	0	JTAG	0			VBAT	0						0	CORE1
W	W1C	W1C		W1C				W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LOCKUP	SW	WDOG	SCG	STOPACK	RSTACK	MDM	PIN	0		FATAL	WARM	HVD	LVD	POR	WAKEUP
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

30.4.1.9.4 Fields

Field	Function
31 TAMPER	Tamper Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by tamper detection. 1b - Reset generated by tamper detection.
30 SECVIO	Security Violation Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by Security Violation detection. 1b - Reset generated by Security Violation detection.
29 —	Reserved
28 JTAG	JTAG System Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by JTAG system reset. 1b - Reset generated by JTAG system reset.

Table continues on the next page...

Field	Function
27-25 —	Reserved
24 VBAT	VBAT System Reset VBAT POR will generate a system reset. 0b - Reset not generated by VBAT system reset. 1b - Reset generated by VBAT system reset.
23-18 —	Reserved
17 —	Reserved
16 CORE1	Core1 System Reset Indicates the reset source was generated by a Core1 specific reset. 0b - Reset not generated from Core1 system reset source. 1b - Reset generated from Core1 system reset source.
15 LOCKUP	Lockup Reset 0b - Reset not generated by core lockup. 1b - Reset generated by core lockup.
14 SW	Software Reset 0b - Reset not generated by software request from core. 1b - Reset generated by software request from core.
13 WDOG	Watchdog Reset 0b - Reset is not generated from the WatchDog timeout. 1b - Reset is generated from the WatchDog timeout.
12 SCG	SCG Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset is not generated from an SCG loss of lock or loss of clock. 1b - Reset is generated from an SCG loss of lock or loss of clock.
11 STOPACK	Stop Timeout Reset 0b - Reset not generated by Stop Controller Timeout. 1b - Reset generated by Stop Controller Timeout.
10 RSTACK	Reset Timeout This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated from Reset Controller Timeout. 1b - Reset generated from Reset Controller Timeout.
9 MDM	MDM Reset 0b - Reset was not generated from the MDM reset request. 1b - Reset was generated from the MDM reset request.
8 PIN	Pin Reset 0b - Reset was not generated from the RESET_B pin. 1b - Reset was generated from the RESET_B pin.
7-6 —	Reserved
5 FATAL	Fatal Reset 0b - Reset was not generated by a fatal reset source. 1b - Reset was generated by a fatal reset source.
4	Warm Reset 0b - Reset not generated by system reset source.

Table continues on the next page...

Field	Function
WARM	1b - Reset generated by system reset source.
3 HVD	HVD Reset 0b - Reset not generated by HVD. 1b - Reset generated by HVD.
2 LVD	LVD Reset 0b - Reset not generated by LVD. 1b - Reset generated by LVD.
1 POR	POR Reset 0b - Reset not generated by POR. 1b - Reset generated by POR.
0 WAKEUP	Wakeup Reset 0b - Reset not generated by wakeup from VLLS mode. 1b - Reset generated by wakeup from VLLS mode.

30.4.1.10 System Reset Interrupt Enable (SRIE)

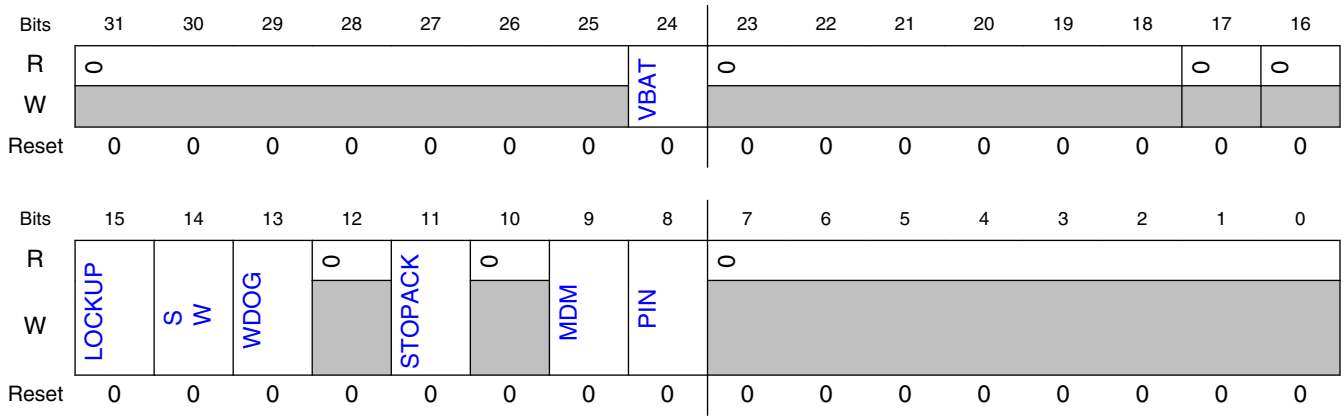
30.4.1.10.1 Offset

Register	Offset
SRIE	2Ch

30.4.1.10.2 Function

This registers delays the assertion of a system reset for between 9 and 10 LPO clock cycles while an interrupt is generated. This allows software to perform a graceful shutdown or to abort the system reset provided the pending reset source is cleared by resetting the source of the reset and then clearing the pending flag. A Chip POR or fatal reset source cannot be delayed by this feature. The SRS will only update after the system reset occurs.

30.4.1.10.3 Diagram



30.4.1.10.4 Fields

Field	Function
31-25 —	Reserved
24 VBAT	VBAT System Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
23-18 —	Reserved
17 —	Reserved
16 —	Reserved.
15 LOCKUP	Lockup Reset Note that interrupts cannot be serviced by a core in the lockup state. 0b - Interrupt disabled. 1b - Interrupt enabled.
14 SW	Software Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
13 WDOG	Watchdog Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
12 —	Reserved
11 STOPACK	Stop Timeout Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
10	Reserved

Table continues on the next page...

Field	Function
—	
9 MDM	MDM Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
8 PIN	Pin Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
7-0 —	Reserved

30.4.1.11 System Reset Interrupt Flag (SRIF)

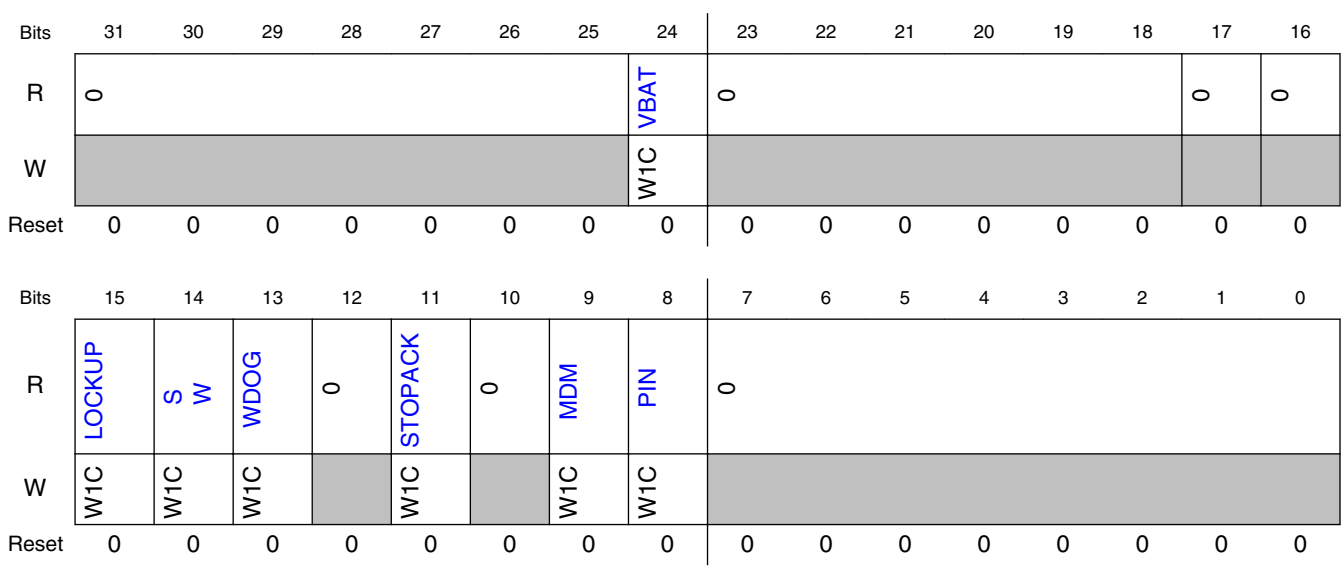
30.4.1.11.1 Offset

Register	Offset
SRIF	30h

30.4.1.11.2 Function

This registers returns the source of the reset interrupt. The pending reset source can be cleared by resetting the source of the reset and then clearing the pending flag.

30.4.1.11.3 Diagram



30.4.1.11.4 Fields

Field	Function
31-25 —	Reserved
24 VBAT	VBAT System Reset 0b - Reset source not pending. 1b - Reset source pending.
23-18 —	Reserved
17 —	Reserved
16 —	Reserved.
15 LOCKUP	Lockup Reset 0b - Reset source not pending. 1b - Reset source pending.
14 SW	Software Reset 0b - Reset source not pending. 1b - Reset source pending.
13 WDOG	Watchdog Reset 0b - Reset source not pending. 1b - Reset source pending.
12 —	Reserved
11 STOPACK	Stop Timeout Reset 0b - Reset source not pending. 1b - Reset source pending.
10 —	Reserved
9 MDM	MDM Reset 0b - Reset source not pending. 1b - Reset source pending.
8 PIN	Pin Reset 0b - Reset source not pending. 1b - Reset source pending.
7-0 —	Reserved

30.4.1.12 Core Software Reset Enable (CSRE)

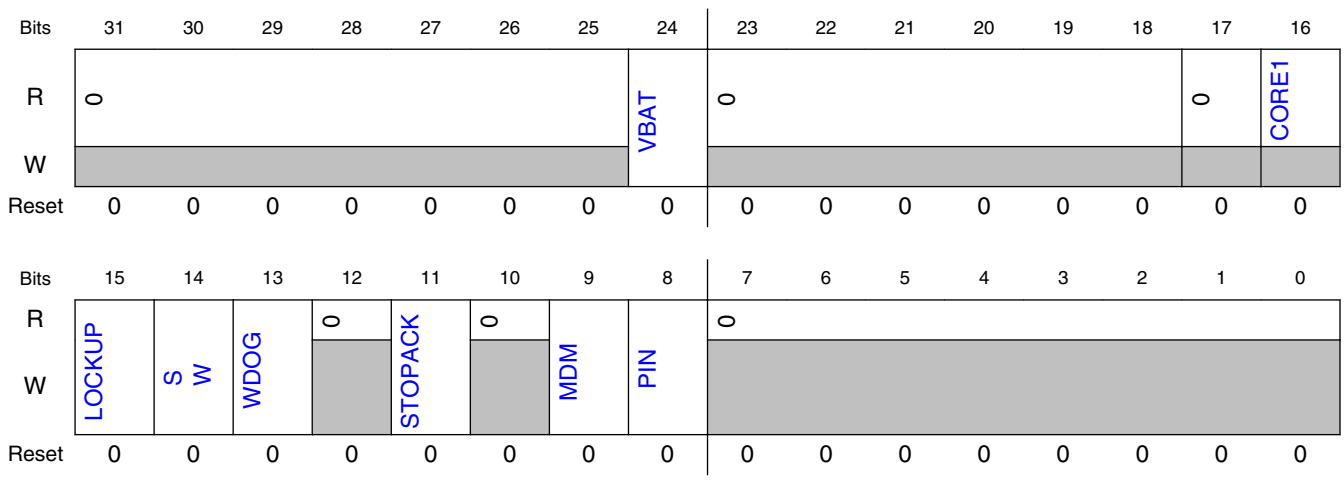
30.4.1.12.1 Offset

Register	Offset
CSRE	34h

30.4.1.12.2 Function

This registers delays the assertion of a system reset for between 9 and 10 LPO clock cycles while a core software reset is generated. This allows software to recover without resetting the entire system, provided the pending reset source is cleared by resetting the source of the reset and then clearing the pending flag. A Chip POR or fatal reset source cannot be delayed by this feature.

30.4.1.12.3 Diagram



30.4.1.12.4 Fields

Field	Function
31-25 —	Reserved
24 VBAT	VBAT System Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
23-18 —	Reserved
17 —	Reserved
16	Core1 System Reset

Table continues on the next page...

Field	Function
CORE1	Indicates the reset source was generated by a Core1 specific reset. 0b - Reset not generated from Core1 system reset source. 1b - Reset generated from Core1 system reset source.
15 LOCKUP	Lockup Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
14 SW	Software Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
13 WDOG	Watchdog Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
12 —	Reserved
11 STOPACK	Stop Timeout Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
10 —	Reserved
9 MDM	MDM Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
8 PIN	Pin Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
7-0 —	Reserved

30.4.1.13 Mode Register (MR)

30.4.1.13.1 Offset

Register	Offset
MR	40h

30.4.1.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BOOTCFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BOOTCFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

30.4.1.13.3 Fields

Field	Function
31-0 BOOTCFG	Boot Configuration The definition of this field is device specific.

30.4.1.14 Force Mode Register (FM)

30.4.1.14.1 Offset

Register	Offset
FM	60h

30.4.1.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FORCECFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FORCECFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

30.4.1.14.3 Fields

Field	Function
31-0 FORCECFG	Boot Configuration This register can force the corresponding bit in the Mode Register to assert on next system reset. 00000000000000000000000000000000b - No effect. 00000000000000000000000000000001b - Assert corresponding bit in Mode Register on next system reset.

30.4.2 SMC register descriptions

Different MSMC registers reset on different reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

NOTE

The MSMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

NOTE

Before executing the WFI instruction, the last register written to must be read back. This ensures that all register writes associated with setting up the low power mode being entered have completed before the MCU enters the low power mode. Failure to do this may result in the low power mode not being entered correctly.

30.4.2.1 MSMC Memory map

MSMC1 base address: 4041_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_00AAh
4h	Parameter Register (PARAM)	32	RO	0000_0001h
8h	Power Mode Protection register (PMPROT)	32	RW	See description.
10h	Power Mode Control register (PMCTRL)	32	RW	See description.

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
18h	Power Mode Status register (PMSTAT)	32	RO	0000_0001h
20h	System Reset Status (SRS)	32	RO	0000_0006h
24h	Reset Pin Control (RPC)	32	RW	0000_0000h
28h	Sticky System Reset Status (SSRS)	32	W1C	0000_0006h
2Ch	System Reset Interrupt Enable (SRIE)	32	RW	0000_0000h
30h	System Reset Interrupt Flag (SRIF)	32	W1C	0000_0000h
34h	Core Software Reset Enable (CSRE)	32	RW	0000_0000h
40h	Mode Register (MR)	32	RO	0000_0000h
60h	Force Mode Register (FM)	32	RW	0000_0000h

30.4.2.2 Version ID Register (VERID)

30.4.2.2.1 Offset

Register	Offset
VERID	0h

30.4.2.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0

30.4.2.2.3 Fields

Field	Function
31-24	Major Version Number
MAJOR	This read only field returns the major version number for the module specification.

Table continues on the next page...

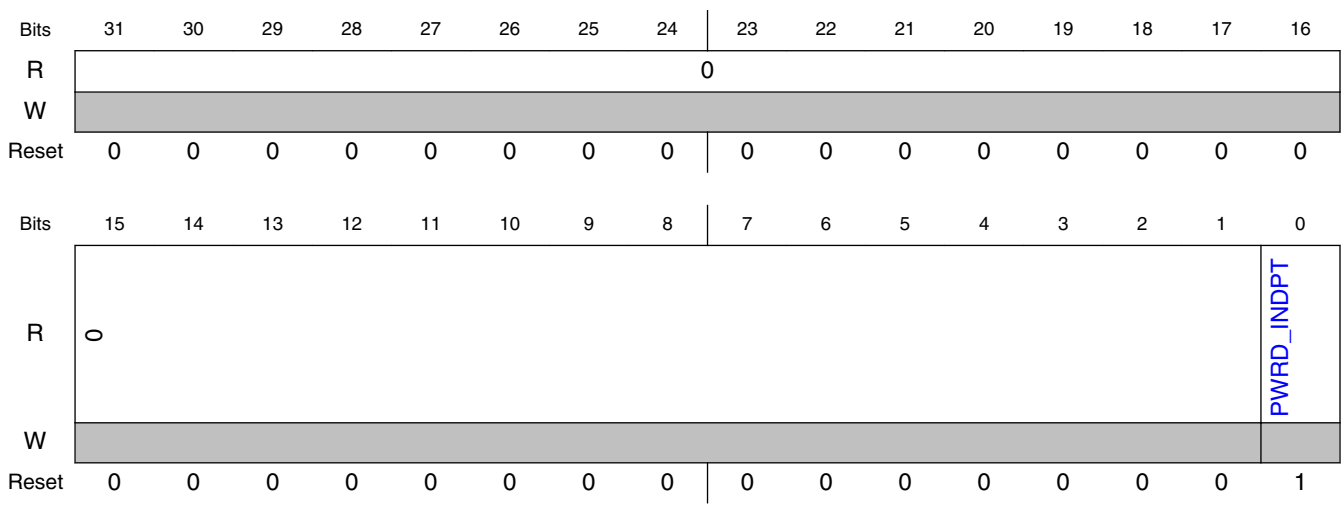
Field	Function
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000010101010b - Default features supported

30.4.2.3 Parameter Register (PARAM)

30.4.2.3.1 Offset

Register	Offset
PARAM	4h

30.4.2.3.2 Diagram



30.4.2.3.3 Fields

Field	Function
31-1 —	Reserved
0 PWRD_INDPT	Power Domains Independent

Field	Function
	When set, means each core is able to enter power modes independently of the other. When clear, means certain dependancies are enforced by the MSMC before each core is able to enter some power modes. See "Power mode transitions" section for detailed requirements for each core to enter a power mode.

30.4.2.4 Power Mode Protection register (PMPROT)

30.4.2.4.1 Offset

Register	Offset
PMPROT	8h

30.4.2.4.2 Function

This register provides protection for entry into any run or stop power mode. The enabling of the run or stop mode occurs by configuring the Power Mode Control register (PMCTRL).

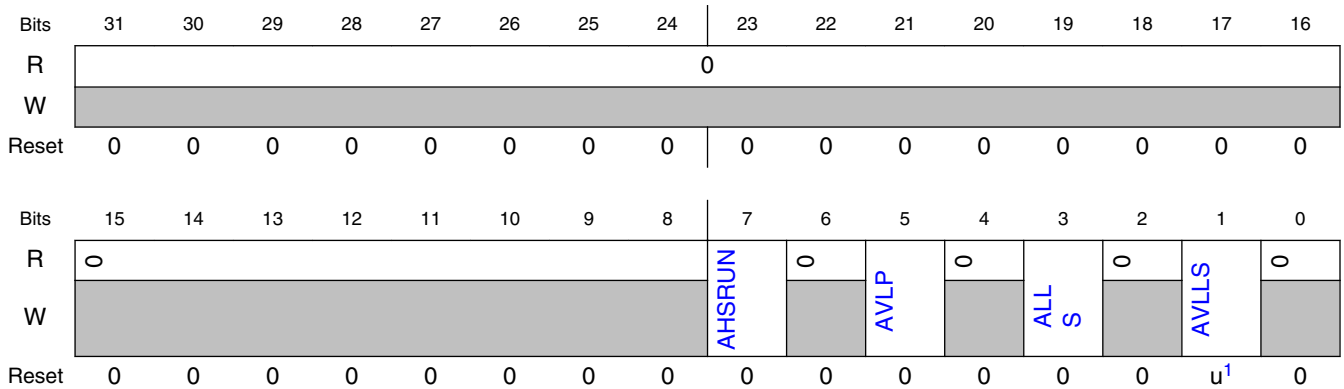
The PMPROT register can be written only once after any system reset.

If a core is configured for a disallowed or reserved power mode, the core remains in its current power mode. For example, if the core is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and PMCTRL[RUNM] remains 00b, indicating the core is still in Normal Run mode.

NOTE

This register is reset on Chip Reset and by reset sources that trigger Chip Reset (see the Reset section for more information).

30.4.2.4.3 Diagram



1. The reset value is 1 on VLLS wakeup, otherwise it is 0.

30.4.2.4.4 Fields

Field	Function
31-8 —	Reserved
7 AHSRUN	Allow High Speed Run mode When set, allows the PMCTRL[RUNM] field to be written to HSRUN. 0b - HSRUN is not allowed 1b - HSRUN is allowed
6 —	Reserved
5 AVLP	Allow Very-Low-Power Modes When set, allows the PMCTRL[RUNM] field to be written to VLPR and/or the PMCTRL[STOPM] field to be written with VLPS. 0b - VLPR, VLPW, and VLPS are not allowed. 1b - VLPR, VLPW, and VLPS are allowed.
4 —	Reserved
3 ALLS	Allow Low-Leakage Stop Mode When set, allows the PMCTRL[STOPM] field to be written to LLS. 0b - LLS is not allowed 1b - LLS is allowed
2 —	Reserved
1 AVLLS	Allow Very-Low-Leakage Stop Mode When set, allows the PMCTRL[STOPM] field to be written to VLLS. 0b - VLLS mode is not allowed 1b - VLLS mode is allowed
0 —	Reserved

30.4.2.5 Power Mode Control register (PMCTRL)

30.4.2.5.1 Offset

Register	Offset
PMCTRL	10h

30.4.2.5.2 Function

The PMCTRL register controls entry into low-power Run and Stop modes, provided that the selected power mode is allowed via an appropriate setting of the protection (PMPROT) register.

NOTE

This register is reset on Chip POR and by reset types that trigger Chip POR. See the Reset section details for more information.

30.4.2.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							STOPA	0						PSTOPO	
W								W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					0	RUNM		0					STOPM		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	u	u

1. The reset value is 100 on VLLS wakeup , otherwise it is 000.

30.4.2.5.4 Fields

Field	Function
31-25	Reserved

Table continues on the next page...

Field	Function
—	
24 STOPA	Stop Abort Flag
23-18 —	Reserved
17-16 PSTOPO	Partial Stop Option These bits control whether a Partial Stop mode is entered when STOPM=STOP. When entering a Partial Stop mode from RUN mode, the PMC and SCG remain fully powered, allowing the device to wakeup almost instantaneously at the expense of higher power consumption. In PSTOP2, only system clocks are gated allowing peripherals running on bus clock to remain fully functional. In PSTOP1, both system and bus clocks are gated. 00b - STOP - Normal Stop mode 01b - PSTOP1 - Partial Stop with system and bus clock disabled 10b - PSTOP2 - Partial Stop with system clock disabled and bus clock enabled 11b - PSTOP3 - Partial Stop with system clock enabled and bus clock enabled
15-11 —	Reserved
10 —	Reserved
9-8 RUNM	Run Mode Control When written, causes entry into the selected run mode. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. **NOTE:** Stop mode entry should not be attempted while RUNM=HSRUN or PMSTAT=HSRUN. 00b - Normal Run mode (RUN) 01b - Reserved 10b - Very-Low-Power Run mode (VLPR) 11b - High Speed Run mode (HSRUN)
7-3 —	Reserved
2-0 STOPM	Stop Mode Control Selects the desired stop mode when a core enters deepsleep. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. After any system reset, this field is cleared by hardware on any successful write to the PMPROT register. **NOTE:** When set to STOP, the PSTOPO bits can be used to select a Partial Stop mode if desired. 000b - Normal Stop (STOP) 001b - Reserved 010b - Very-Low-Power Stop (VLPS) 011b - Low-Leakage Stop (LLS) 100b - Very-Low-Leakage Stop (VLLS) 101b - Reserved 110b - Reserved 111b - Reserved

30.4.2.6 Power Mode Status register (PMSTAT)

30.4.2.6.1 Offset

Register	Offset
PMSTAT	18h

30.4.2.6.2 Function

PMSTAT is a read-only, one-hot register which indicates the current power mode of the system.

30.4.2.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								PMSTAT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

30.4.2.6.4 Fields

Field	Function
31-24 —	Reserved
23-8 —	Reserved
7-0 PMSTAT	Power Mode Status **NOTE:** When debug is enabled, the PMSTAT will not update to STOP or VLPS **NOTE:** When a Partial Stop Option is enabled, the PMSTAT will not update to STOP or VLPS 00000001b - Current power mode is RUN. 00000010b - Current power mode is any STOP mode. 00000100b - Current power mode is VLPR. 10000000b - Current power mode is HSRUN

30.4.2.7 System Reset Status (SRS)

30.4.2.7.1 Offset

Register	Offset
SRS	20h

30.4.2.7.2 Function

This register includes read-only status flags to indicate the source of the most recent reset.

NOTE

The reset state of this register depends on the reset source:

- POR = 0x0000_0006
- LVD = 0x0000_0004
- VLLS Wakeup = 0x0000_0001
- VLLS Reset Wakeup = 0x0000_0111

30.4.2.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TAMPER	SECVIO	TZWDG	JTAG	0			VBAT	0						0	CORE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SW	WDOG	SCG	STOPACK	RSTACK	MDM	PIN	CORE	0	FATAL	WARM	HVD	LVD	POR	WAKEUP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

30.4.2.7.4 Fields

Field	Function
31 TAMPER	Tamper Reset Indicates a reset has been caused by Tamper Detection logic. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by tamper detection. 1b - Reset generated by tamper detection.
30 SECVIO	Security Violation Reset Indicates a reset has been caused by a Security Violation logic. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by security violation. 1b - Reset generated by security violation.
29 TZWDG	TrustZone WatchDog Reset Indicates a reset has been caused by the TrustZone WatchDog timeout. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by TrustZone WatchDog timeout. 1b - Reset generated by TrustZone WatchDog timeout.
28 JTAG	JTAG System Reset Indicates a reset has been caused by a JTAG system reset request. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by JTAG system reset. 1b - Reset generated by JTAG system reset.
27-25 —	Reserved
24 VBAT	VBAT System Reset VBAT POR will generate a system reset. 0b - Reset not generated by VBAT system reset. 1b - Reset generated by VBAT system reset.
23-18 —	Reserved
17 —	Reserved
16 CORE0	Core0 System Reset Indicates the reset source was generated by a Core0 specific reset. 0b - Reset not generated from Core0 system reset source. 1b - Reset generated from Core0 system reset source.
15 —	Reserved
14 SW	Software Reset Indicates a reset has been caused by a software reset request from the Arm core (SYSRESETREQ). 0b - Reset not generated by software request from core. 1b - Reset generated by software request from core.
13 WDOG	Watchdog Reset Indicates a reset has been caused by a WatchDog timeout. 0b - Reset is not generated from the WatchDog timeout. 1b - Reset is generated from the WatchDog timeout.

Table continues on the next page...

Field	Function
12 SCG	SCG Reset Indicates a reset has been caused by a loss-of-clock or loss-of-lock event in the SCG. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset is not generated from an SCG loss of lock or loss of clock. 1b - Reset is generated from an SCG loss of lock or loss of clock.
11 STOPACK	Stop Timeout Reset Indicates a reset has been caused by a timeout in the Stop mode entry logic. This timeout is generated if a peripheral does not acknowledge entry into any Stop mode within approximately one second. 0b - Reset not generated by Stop Controller Timeout. 1b - Reset generated by Stop Controller Timeout.
10 RSTACK	Reset Timeout Indicates a reset has been caused by a timeout in the system reset generaton logic. This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated from Reset Controller Timeout. 1b - Reset generated from Reset Controller Timeout.
9 MDM	MDM Reset Indicates a reset has been caused by a reset request from the Miscellaneous Debug Module (MDM). 0b - Reset was not generated from the MDM reset request. 1b - Reset was generated from the MDM reset request.
8 PIN	Pin Reset Indicates a reset has been caused by external assertion of the RESET_b pin. 0b - Reset was not generated from the assertion of RESET_B pin. 1b - Reset was generated from the assertion of RESET_B pin.
7 CORE	Core Reset Indicates the last reset was a reset that only reset the core. 0b - Reset source was not core only reset. 1b - Reset source was core reset and reset the core only.
6 —	Reserved
5 FATAL	Fatal Reset Fatal Reset will assert if the system reset source was fatal. SRAM contents cannot be guaranteed following a fatal reset source. 0b - Reset was not generated by a fatal reset source. 1b - Reset was generated by a fatal reset source.
4 WARM	Warm Reset Warm Reset flag will assert whenever any of the system reset sources in this register assert (SRS[31:8]). 0b - Reset not generated by Warm Reset source. 1b - Reset generated by Warm Reset source.
3 HVD	HVD Reset Indicates a reset has been caused by a High Voltage Detect. SRAM contents cannot be guaranteed following a HVD reset source. 0b - Reset not generated by HVD. 1b - Reset generated by HVD.
2 LVD	LVD Reset Indicates a reset has been caused by a Low Voltage Detect. SRAM contents cannot be guaranteed following a LVD reset source. 0b - Reset not generated by LVD. 1b - Reset generated by LVD.

Table continues on the next page...

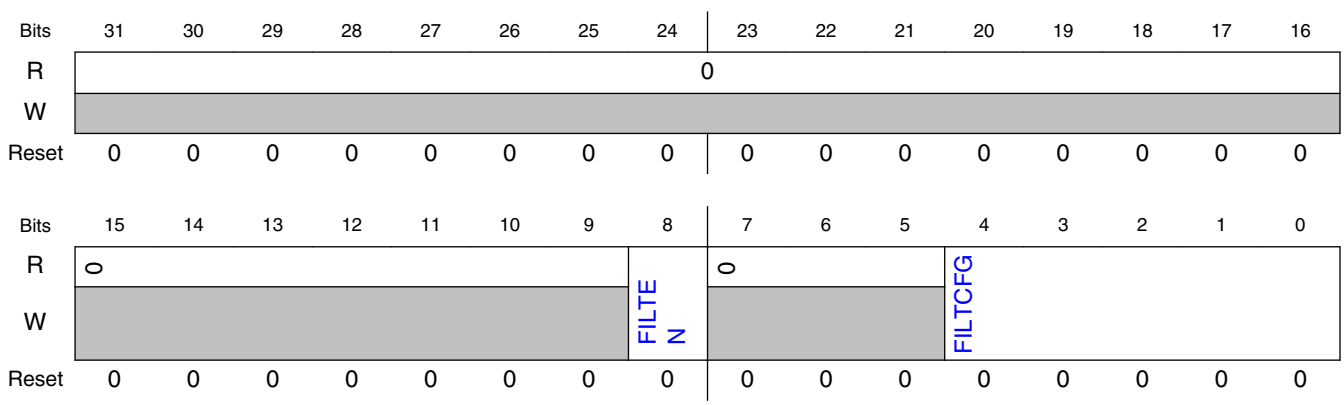
Field	Function
1 POR	POR Reset Indicates a reset has been caused by Power On Reset detection logic. SRAM contents cannot be guaranteed following a POR reset source. 0b - Reset not generated by POR. 1b - Reset generated by POR.
0 WAKEUP	Wakeup Reset Indicates a reset has been caused by a wakeup from VLLS mode. 0b - Reset not generated by wakeup from VLLS mode. 1b - Reset generated by wakeup from VLLS mode.

30.4.2.8 Reset Pin Control (RPC)

30.4.2.8.1 Offset

Register	Offset
RPC	24h

30.4.2.8.2 Diagram



30.4.2.8.3 Fields

Field	Function
31-9 —	Reserved
8	Filter Enable 0b - Slow clock reset pin filter disabled.

Table continues on the next page...

Field	Function
FILTEN	1b - Slow clock reset pin filter enabled in Run modes.
7-5 —	Reserved
4-0 FILTCFG	Reset Filter Configuration Configures the width of the reset pin filter from 1 to 32 slow clock cycles.

30.4.2.9 Sticky System Reset Status (SSRS)

30.4.2.9.1 Offset

Register	Offset
SSRS	28h

30.4.2.9.2 Function

The SSRS are only reset on POR/LVD/HVD/WAKEUP and remain set until cleared by software. The SSRS stores all system reset sources that have generated a system reset, since the last POR/LVD/HVD/WAKEUP, that have not been cleared by software. The SSRS does not update following a core software reset.

30.4.2.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TAMPER	SECVIO	TZWDG	JTAG	0			VBAT	0						0	CORE0
W	W1C	W1C	W1C	W1C				W1C								W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SW	WDOG	SCG	STOPACK	RSTACK	MDM	PIN	0		FATAL	WARM	HVD	LVD	POR	WAKEUP
W		W1C	W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

30.4.2.9.4 Fields

Field	Function
31 TAMPER	Tamper Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by tamper detection. 1b - Reset generated by tamper detection.
30 SECVIO	Security Violation Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by Security Violation detection. 1b - Reset generated by Security Violation detection.
29 TZWDG	TrustZone WatchDog Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by TrustZone WatchDog timeout. 1b - Reset generated by TrustZone WatchDog timeout.
28 JTAG	JTAG System Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated by JTAG system reset. 1b - Reset generated by JTAG system reset.
27-25 —	Reserved
24 VBAT	VBAT System Reset VBAT POR will generate a system reset. 0b - Reset not generated by VBAT system reset. 1b - Reset generated by VBAT system reset.
23-18 —	Reserved
17 —	Reserved
16 CORE0	Core0 Reset 0b - Reset not generated from Core0 reset source. 1b - Reset generated from Core0 reset source.
15 —	Reserved
14 SW	Software Reset 0b - Reset not generated by software request from core. 1b - Reset generated by software request from core.
13 WDOG	Watchdog Reset 0b - Reset is not generated from the WatchDog timeout. 1b - Reset is generated from the WatchDog timeout.
12 SCG	SCG Reset This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset is not generated from an SCG loss of lock or loss of clock. 1b - Reset is generated from an SCG loss of lock or loss of clock.
11	Stop Timeout Reset 0b - Reset not generated by Stop Controller Timeout.

Table continues on the next page...

Field	Function
STOPACK	1b - Reset generated by Stop Controller Timeout.
10 RSTACK	Reset Timeout This is a fatal reset source and SRAM contents are not guaranteed. 0b - Reset not generated from Reset Controller Timeout. 1b - Reset generated from Reset Controller Timeout.
9 MDM	MDM Reset 0b - Reset was not generated from the MDM reset request. 1b - Reset was generated from the MDM reset request.
8 PIN	Pin Reset 0b - Reset was not generated from the RESET_B pin. 1b - Reset was generated from the RESET_B pin.
7-6 —	Reserved
5 FATAL	Fatal Reset 0b - Reset was not generated by a fatal reset source. 1b - Reset was generated by a fatal reset source.
4 WARM	Warm Reset 0b - Reset not generated by system reset source. 1b - Reset generated by system reset source.
3 HVD	HVD Reset 0b - Reset not generated by HVD. 1b - Reset generated by HVD.
2 LVD	LVD Reset 0b - Reset not generated by LVD. 1b - Reset generated by LVD.
1 POR	POR Reset 0b - Reset not generated by POR. 1b - Reset generated by POR.
0 WAKEUP	Wakeup Reset 0b - Reset not generated by wakeup from VLLS mode. 1b - Reset generated by wakeup from VLLS mode.

30.4.2.10 System Reset Interrupt Enable (SRIE)

30.4.2.10.1 Offset

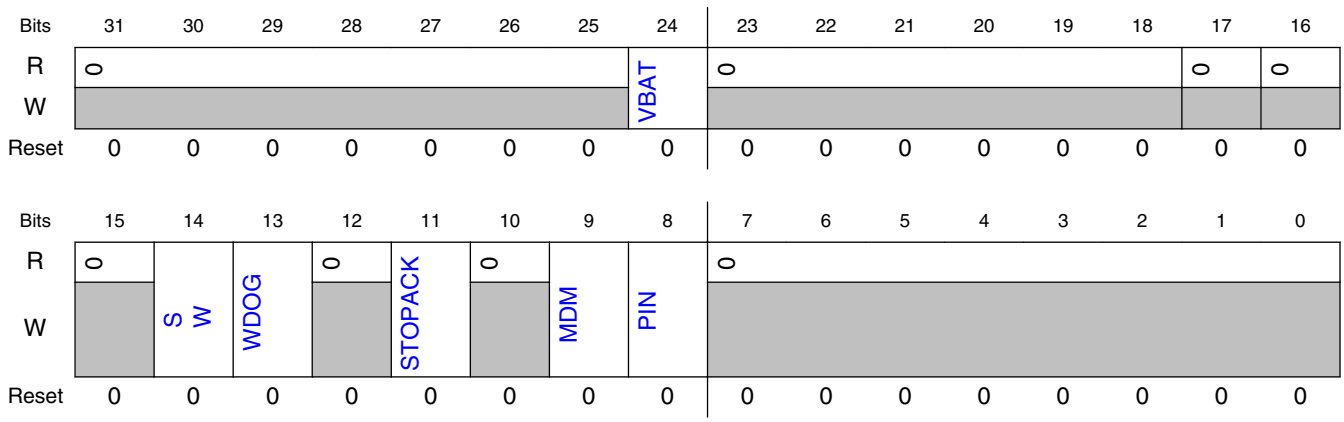
Register	Offset
SRIE	2Ch

30.4.2.10.2 Function

This registers delays the assertion of a system reset for between 9 and 10 LPO clock cycles while an interrupt is generated. This allows software to perform a graceful

shutdown or to abort the system reset provided the pending reset source is cleared by resetting the source of the reset and then clearing the pending flag. A Chip POR or fatal reset source cannot be delayed by this feature. The SRS will only update after the system reset occurs.

30.4.2.10.3 Diagram



30.4.2.10.4 Fields

Field	Function
31-25 —	Reserved
24 VBAT	VBAT System Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
23-18 —	Reserved
17 —	Reserved
16 —	Reserved.
15 —	Reserved
14 SW	Software Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
13 WDOG	Watchdog Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
12	Reserved

Table continues on the next page...

Field	Function
—	
11 STOPACK	Stop Timeout Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
10 —	Reserved
9 MDM	MDM Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
8 PIN	Pin Reset 0b - Interrupt disabled. 1b - Interrupt enabled.
7-0 —	Reserved

30.4.2.11 System Reset Interrupt Flag (SRIF)

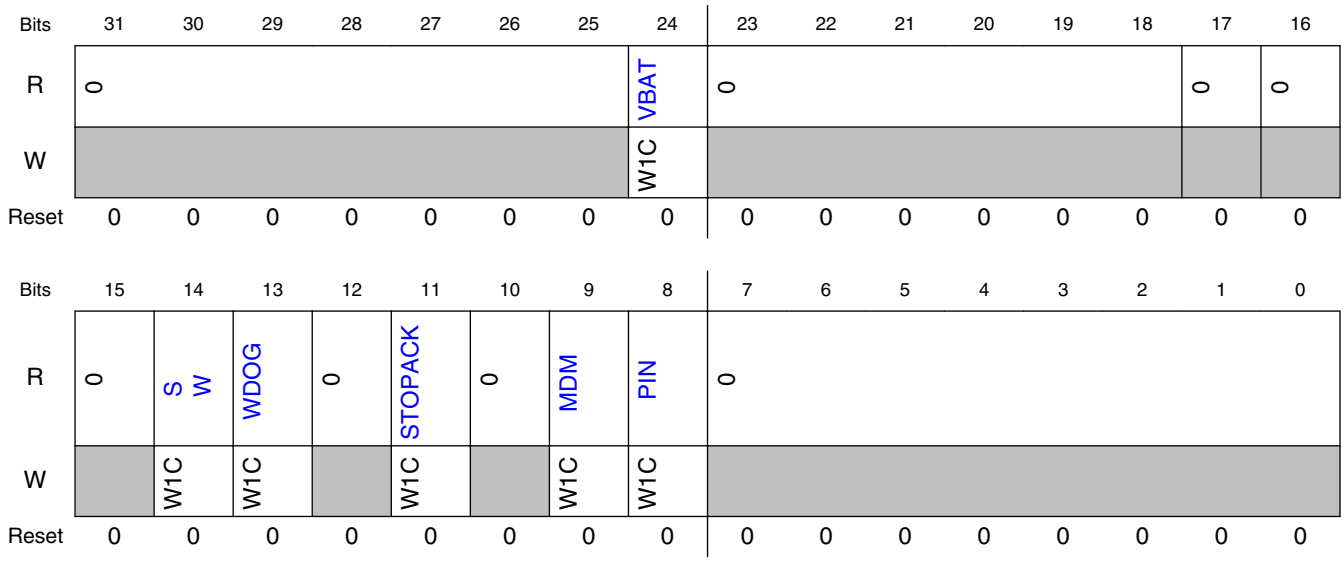
30.4.2.11.1 Offset

Register	Offset
SRIF	30h

30.4.2.11.2 Function

This registers returns the source of the reset interrupt. The pending reset source can be cleared by resetting the source of the reset and then clearing the pending flag.

30.4.2.11.3 Diagram



30.4.2.11.4 Fields

Field	Function
31-25 —	Reserved
24 VBAT	VBAT System Reset 0b - Reset source not pending. 1b - Reset source pending.
23-18 —	Reserved
17 —	Reserved
16 —	Reserved.
15 —	Reserved
14 SW	Software Reset 0b - Reset source not pending. 1b - Reset source pending.
13 WDOG	Watchdog Reset 0b - Reset source not pending. 1b - Reset source pending.
12 —	Reserved
11	Stop Timeout Reset 0b - Reset source not pending.

Table continues on the next page...

Field	Function
STOPACK	1b - Reset source pending.
10 —	Reserved
9 MDM	MDM Reset 0b - Reset source not pending. 1b - Reset source pending.
8 PIN	Pin Reset 0b - Reset source not pending. 1b - Reset source pending.
7-0 —	Reserved

30.4.2.12 Core Software Reset Enable (CSRE)

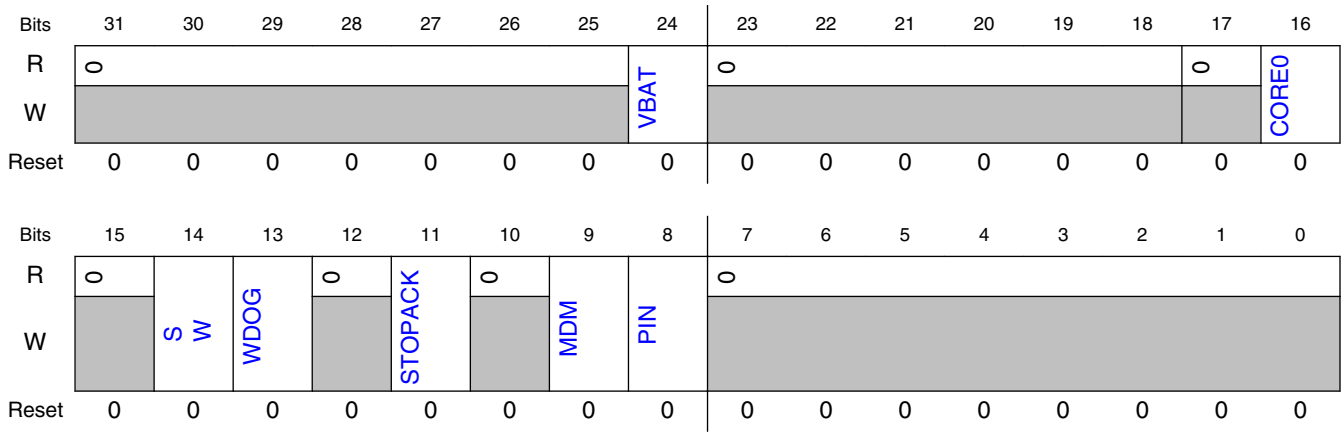
30.4.2.12.1 Offset

Register	Offset
CSRE	34h

30.4.2.12.2 Function

This registers delays the assertion of a system reset for between 9 and 10 LPO clock cycles while a core software reset is generated. This allows software to recover without resetting the entire system, provided the pending reset source is cleared by resetting the source of the reset and then clearing the pending flag. A Chip POR or fatal reset source cannot be delayed by this feature.

30.4.2.12.3 Diagram



30.4.2.12.4 Fields

Field	Function
31-25 —	Reserved
24 VBAT	VBAT System Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
23-18 —	Reserved
17 —	Reserved
16 CORE0	Core0 Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
15 —	Reserved
14 SW	Software Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
13 WDOG	Watchdog Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
12 —	Reserved
11 STOPACK	Stop Timeout Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
10 —	Reserved

Table continues on the next page...

Field	Function
9 MDM	MDM Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
8 PIN	Pin Reset 0b - Core software reset disabled. 1b - Core software reset enabled.
7-0 —	Reserved

30.4.2.13 Mode Register (MR)

30.4.2.13.1 Offset

Register	Offset
MR	40h

30.4.2.13.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BOOTCFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BOOTCFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

30.4.2.13.3 Fields

Field	Function
31-0 BOOTCFG	Boot Configuration The definition of this field is device specific.

30.4.2.14 Force Mode Register (FM)

30.4.2.14.1 Offset

Register	Offset
FM	60h

30.4.2.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	FORCECFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FORCECFG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

30.4.2.14.3 Fields

Field	Function
31-0 FORCECFG	Boot Configuration This register can force the corresponding bit in the Mode Register to assert on next system reset. 00000000000000000000000000000000b - No effect. 00000000000000000000000000000001b - Assert corresponding bit in Mode Register on next system reset.

30.5 Functional description

This section provides a complete functional description of the block.

30.5.1 Power mode transitions

A figure showing the power mode state transitions and a table defining triggers for the various state transitions can be found here.

The following figure shows the power mode state transitions available to each core. Any chip reset will initially bring both cores back to a RUN state. An exception is a POR event which could force core 1 to be disabled until re-enabled by core 0.

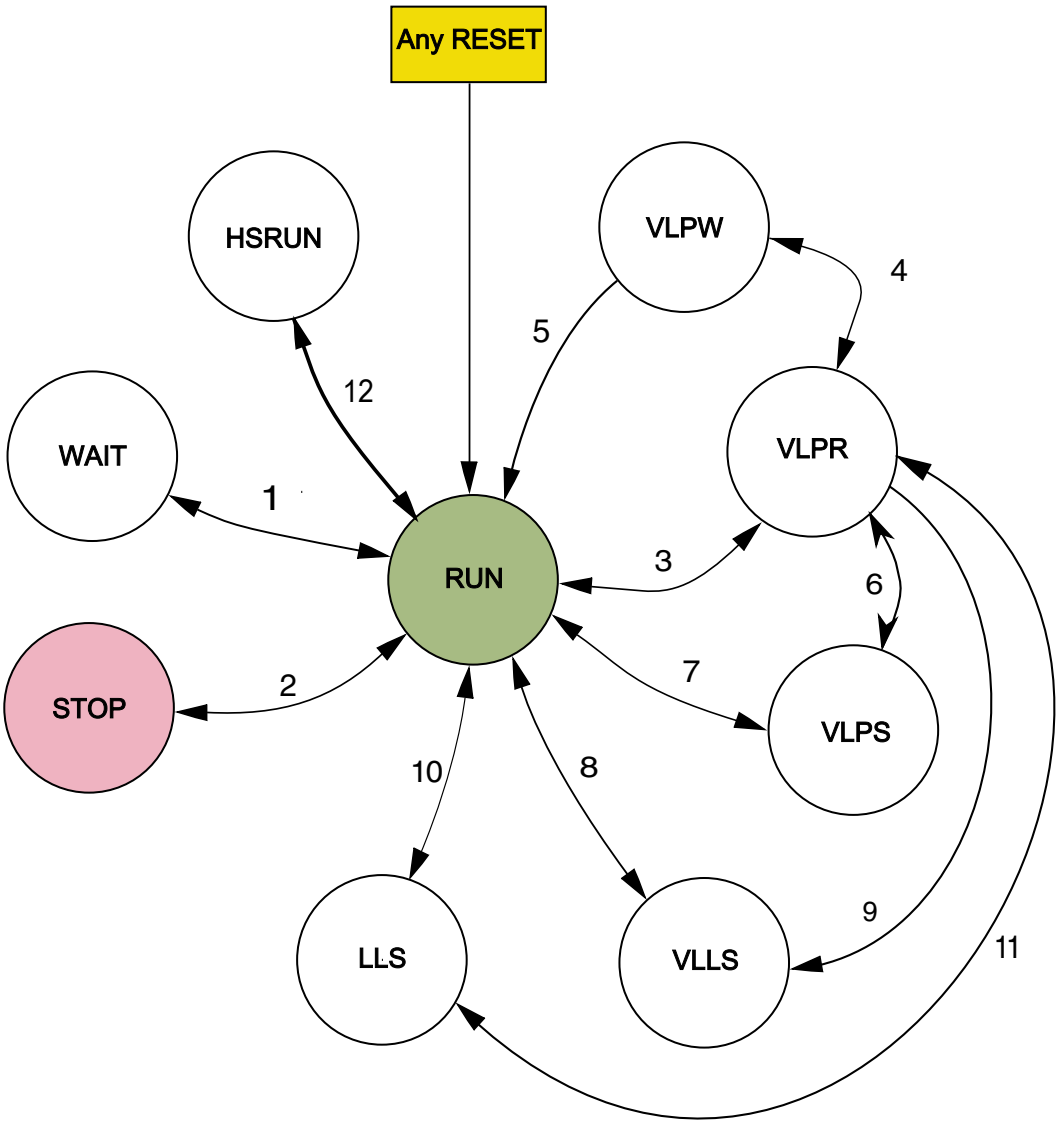


Figure 30-2. Power mode state diagram

The MSMC allows each core to enter/exit both stop and run modes independently of the other. The user should take care to ensure that any common system resources (e.g. clock sources) are always available whenever required for a core's current power mode. For example, if core 1 is being driven by a clock source in core 0's power domain, the user should ensure that core 0 does not enter a run/stop mode which could cause that clock to be disabled.

The specific trigger conditions required for each core to transition into or out of power modes depend on the device. See the chip-specific MSMC information for details.

30.5.2 Run modes

The MSMC supports the following run modes:

- Normal Run (RUN)
- Very Low-Power Run (VLPR)
- High Speed Run (HSRUN)

30.5.2.1 Run (RUN) mode

This is the normal operating mode for the device.

This mode is selected after any reset. When the Arm processor exits reset, it sets up the stack, program counter (PC), and link register (LR):

- The processor reads the start SP (SP_main) from vector-table offset 0x000
- The processor reads the start PC from vector-table offset 0x004
- LR is set to 0xFFFF_FFFF.

To reduce power in this mode, clocks to unused modules can be disabled using their corresponding clock gating control bits in the Peripheral Clock Control modules (see PCC chapter) or compute operation can be enabled via the Miscellaneous Control Module (see MCM chapter).

30.5.2.2 Very Low Power Run (VLPR) mode

VLPR mode is designed to provide the lowest power operating mode to the core at a reduced frequency. See the PMC chapter for additional power options available in this mode. Before entering this mode, the following conditions must be met:

- The maximum frequencies of the system, bus, flash, and core are restricted. See the Power Management details about which frequencies are supported.
- Mode protection must be set to allow VLP modes, that is, PMPROT[AVLP] is 1.
- PMCTRL[RUNM] must be set to VLPR.
- Flash programming/erasing is not allowed.

NOTE

While in VLPR mode, do not modify the clock source in the SCG module or any clock divider registers. Module clock enables in the PCC can be set, however should only be cleared one at a time to limit load transitions.

To reenter Normal Run mode, clear PMCTRL[RUNM]. PMSTAT is a read-only status register that can be used to determine when the system has completed an exit to RUN mode. When PMSTAT=RUN, the system can run at full speed in any clock mode.

Any reset always causes an exit from VLPR and returns a core to RUN mode after the exiting its reset flow.

To further reduce power in this mode, clocks to unused modules can be disabled using their corresponding clock gating control bits in the Peripheral Clock Control modules (see PCC chapter) or compute operation can be enabled via the Miscellaneous Control Module (see MCM chapter).

30.5.2.3 High Speed Run (HSRUN) mode

HSRUN mode is designed to provide the highest performance mode to the core at the fastest supported frequency. See the PMC chapter for additional power options available in this mode. Before entering this mode, the following conditions must be met:

- The maximum allowable change in frequency of the system, bus, flash or core clocks is restricted to x2.
- Mode protection must be set to allow HSRUN modes, that is, PMPROT[AHSRUN] is 1.
- PMCTRL[RUNM] must be set to HSRUN
- Flash programming/erasing is not allowed. (Not all devices contain on-chip flash.)
- Stop mode entry is not allowed.

To reenter normal RUN mode, clear the RUNM register. PMSTAT is a read-only status register that can be used to determine when the system has completed an exit to RUN mode. Any reset will also clear RUNM and cause the core to exit to normal RUN mode after exiting its reset flow.

30.5.3 Wait modes

The MSMC supports the following wait modes.

- Wait (WAIT)
- Very-Low Power Wait (VLPW)

30.5.3.1 Wait (WAIT) mode

WAIT mode is designed to allow a core to enter a static, low power state with instant wakeup times, while still allowing peripherals to operate with full functionality. WAIT mode is entered when the Arm core enters the Sleep-Now or Sleep-On-Exit mode while SLEEDEEP is cleared. The core enters a low-power state in which it is not clocked, but peripherals continue to be clocked provided they are enabled. Clock gating to the peripheral is enabled via the PCC module.

When an interrupt request occurs, the core exits WAIT mode and resumes processing in RUN mode, beginning with the stacking operations leading to the interrupt service routine.

A system reset will also cause an exit from WAIT mode, returning the core to normal RUN mode.

30.5.3.2 Very Low Power Wait (VLPW) mode

VLPW mode is designed to allow a core to enter a static, very low power state, while still allowing peripherals to operate with full functionality at a reduced frequency. VLPW mode is entered when the Arm core enters the Sleep-Now or Sleep-On-Exit mode while SLEEPDEEP is cleared and the core is in VLPR mode. The ARM CPU enters a low-power state in which it is not clocked, but peripherals continue to be clocked provided they are enabled. Clock gating to the peripheral is enabled via the PCC module.

VLPR mode restrictions also apply to VLPW.

When an interrupt from VLPW occurs, the device returns to VLPR mode to execute the interrupt service routine.

When an interrupt request occurs, the CPU exits VLPW mode and resumes processing in VLPR mode, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from VLPW mode, returning the device to normal RUN mode.

30.5.4 Stop modes

The MSMC supports a wide variety of stop modes for each core ranging from:

- a stopped core, with all I/O, logic, and memory states retained, and certain asynchronous mode peripherals operating

to:

- a powered down core, with only I/O and a small register file retained, very few asynchronous mode peripherals operating

The choice of stop mode depends upon the user's application, and how power usage and state retention versus functional needs and recovery time may be traded off.

The stop mode for a core can be selected by setting the appropriate fields in PMPROT and PMCTRL register fields in SMC0 and SMC1 (see "Power mode transitions" section for details).

The following stop modes are supported:

- Normal Stop (STOP)
- Very Low Power Stop (VLPS)
- Low Leakage Stop (LLS)
- Very Low Leakage Stop (VLLS)

30.5.4.1 Stop (STOP) mode

STOP mode is designed to allow a core and its system to enter a static, low power state with relatively fast wakeup times, while still allowing peripherals to operate with full asynchronous functionality. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in STOP mode and see the PMC chapter for additional power options available in this mode.

A core can enter STOP mode by executing Sleep-Now or Sleep-On-Exit with SLEEPDEEP=1 while the core's STOPM equates to STOP (see "Power mode transitions" section for details). Note that a core's PSTOPO field can be set to enable a partial stop mode, allowing instantaneous wakeup at the expense of power consumption.

A module capable of providing an asynchronous interrupt can trigger an exit from STOP mode and return the device to normal RUN mode. When an interrupt request occurs, the core exits STOP mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will also cause an exit from STOP mode, returning the device to normal RUN mode.

30.5.4.2 Very Low Power Stop (VLPS) mode

VLPS mode is designed to allow a core and its system to enter a static, very low power state with relatively fast wakeup times, while still allowing peripherals to operate with limited asynchronous functionality. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in VLPS mode and see the PMC chapter for additional power options available in this mode.

A core can enter VLPS mode by executing Sleep-Now or Sleep-On-Exit with SLEEPDEEP=1 while the core's STOPM either equates to VLPS or equates to STOP with RUNM=VLPR (see "Power mode transitions" section for details).

A module capable of providing an asynchronous interrupt can trigger an exit from VLPS mode and return the device to the run mode from which VLPS was entered (i.e. RUN or VLPR mode). When an interrupt request occurs, the core exits VLPS mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will also cause an exit from VLPS mode, returning the device to normal RUN mode.

30.5.4.3 Low Leakage Stop (LLS) mode

LLS mode is designed to allow a core and its system to enter a low leakage, power gated state with relatively fast wakeup times, while still retaining the state of all logic. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in LLS mode and see the PMC chapter for additional power options available in this mode.

A core can enter LLS mode by executing Sleep-Now or Sleep-On-Exit with SLEEPDEEP=1 while the core's STOPM equates to LLS (see "Power mode transitions" section for details).

Before entering LLS mode, the user should configure the Low Leakage Wake-up unit (LLWU) to enable the desired pins and/or peripherals as wake-up sources for core 0. The available wake-up sources to the LLWU are detailed in the chip configuration details for this device. For core 1, the Messaging Unit (MU) needs to configured to allow core 0 to assert an interrupt or reset to wakeup core 1.

Once configured, the LLWU can trigger an exit from LLS mode and return the device to the run mode from which LLS was entered (i.e. RUN or VLPR mode) with a pending LLWU interrupt. When an LLWU wakeup request occurs, the core exits LLS mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine. In the LLWU interrupt service routine (ISR), the user can read the LLWU module wake-up flags to determine the source of the wakeup.

A pin reset will also cause an exit from LLS mode, returning the device to normal RUN mode. When exiting LLS via the reset pin, the PIN and WAKEUP bits within the SRS register will be set.

NOTE

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully recover from LLS mode after an LLWU wakeup event.

30.5.4.4 Very Low Leakage Stop (VLLS) modes

VLLS mode is designed to allow a core and its system to enter a very low leakage, power gated state. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in VLLS mode and see the PMC chapter for additional power options available in this mode.

A core can enter VLLS mode by executing Sleep-Now or Sleep-On-Exit with SLEEPDEEP=1 while the core's STOPM equates to VLLS (see "Power mode transitions" section for details).

Before entering VLLS mode, the user should configure the Low Leakage Wake-up unit (LLWU) to enable the desired pins and/or peripherals as wake-up sources for core 0. The available wake-up sources to the LLWU are detailed in the chip configuration details for this device. For core 1, the Messaging Unit (MU) needs to configured to allow core 0 to assert an interrupt or reset to wakeup core 1.

Once configured, the LLWU can trigger an exit from VLLS mode and return the device to RUN mode with a pending LLWU interrupt. When an LLWU wakeup request occurs, the core exits VLLS mode and begins executing the reset flow. In the LLWU interrupt service routine (ISR), the user can read the LLWU module wake-up flags to determine the source of the wakeup.

A pin reset will also cause an exit from VLLS mode, returning the device to normal RUN mode. When exiting VLLS via the reset pin, the PIN and WAKEUP bits within the SRS register will be set.

When entering VLLS, each I/O pin in the core's power domain is statically latched as configured before executing VLLS. Because all logic is powered off, all port and peripheral data is lost during VLLS. This data should be restored before writing the ACKISO bit in the power management module, after which I/O states are released to be driven by the restored logic.

Chapter 31
System Integration Module (SIM)

31.1 Chip-specific SIM information

Table 31-1. Reference links to related information

Topic	Related module	Reference
Full description	SIM	SIM
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

31.1.1 System Integration Module

The System Integration Module (SIM) provides system control and chip configuration registers. The SIM also includes an instance of TSTMR.

Table 31-2. SIM configuration

Parameter	Description
Name	SIM
Instances	1
Configurable features	Device specific
Interface speed	NA
External I/O pins	NA

31.2 Introduction

The system integration module (SIM) provides system control and chip configuration registers.

31.3 SIM DGO Register Protocol

A set of registers have been made available in DGO domain for SW use, with the objective to maintain its value between system resets. DGO registers are reseted only in POR event. Since LV and DGO registers are in asynchronous domains a synchronization plus rising edge detector is present in the update register path. In addition DGO domain is in slower clock, so an interrupt is available for software to detect DGO register is updated. In addition a reset de-assert detect circuit is used to update LV registers with DGO value in the event of system reset. In case of POR event, both LV and DGO versions are reset to all zeroes.

DGO register update need to follow the sequence below:

1. Write to SIM_DGO_GPx register

2. Set SIM_DGO_CTRLx[UPDATE] bit

3. Poll SIM_DGO_CTRLx[ACK] bit or wait for interrupt (in this case need to enable interrupt prior to the sequence)

4. Clear SIM_DGO_CTRLx[UPDATE] bit

5. Write 1 to clear SIM_DGO_CTRLx[ACK] bit

31.4 GPIO pads operating range configuration

GPIO pads can operate in 3 ranges of I/O supply:

- 1.8 V range (1.71 V ~ 1.89 V)
- 3.3 V range (3.00 V ~ 3.60 V)
- Continuous range (1.71 V ~ 3.60 V)

NOTE

See i.MX 7ULP Data Sheet for recommended operating values for each I/O supply

- Ports A, C, E, F GPIO pads must be configured to operate in one of the 3 possible operating ranges.
- Port B GPIO pads must be configured to operate in either "Continuous range" or in "1.8 V range".
- Port D GPIO pads are not configurable and can operate in "1.8 V range" or "3.3 V range".
- Each GPIO port can be individually configured in the SIM_DGO_GP11 register according to the values shown in the following table.
- Operating in the "Continuous range" turns on a voltage detection circuitry resulting in additional power consumption. If this additional power consumption is not desired, for example in low power modes, it's recommended to configure either Low or High range.

Table 31-3. Pads Range Configuration

Mode	SIM_DGO_GP11 [PT*_RANGE_CTRL]	Operation
Continuous Range	00	Pads can be powered in the "Continuous range". The I/O supply voltage detector is enabled.
Low Range	01	Pads can be powered in the "1.8 V range" only. Powering the I/O port in the "3.3 V range" may damage the pads. The I/O supply voltage detector is disabled.
High Range	10	Pads can be powered in the "3.3 V range" only. Powering the pads in the "1.8 V range" may cause malfunction. The I/O supply voltage detector is disabled.
Prohibited	11	Invalid configuration. Will cause extra power consumption and may cause pads damage.

Reset value for SIM_DGO_GP11[PT*_RANGE_CTRL] bits are loaded from fuses during reset, but values can be overriden.

NOTE

SIM_DGO_GP11 register is write-once after reset

31.5 Memory Map and register definition

This section includes the SIM module memory map and detailed descriptions of all registers.

31.5.1 SIM Memory Map register descriptions

The memory map comprises of 32-bit aligned registers which can be accessed via 8-bit, 16-bit, or 32-bit accesses. Write access to reserved locations will generate transfer error. Read access to reserved locations will also generate transfer error and the read data bus will show all 0s.

The module will not check for correctness of programmed values in the registers and software must ensure that correct values are being written.

31.5.1.1 SIM Memory map

SIM base address: 410A_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	SIM Systems Options Register 1 (SOPT1)	32	RW	0000_0000h
4h	SIM SOPT1 Configuration Register (SOPT1CFG)	32	RW	0000_0400h
Ch	HW SIM HSIC CAL Register (HSIC_CAL)	32	RW	0000_0000h
28h	SNVS Misc Control Register (SNVS_MISC_CTRL)	32	RW	0000_0000h
2Ch	HW SIM General Purpose Register 0 (GPR0)	32	RW	0000_0000h
30h	HW SIM General Purpose Register 1 (GPR1)	32	RW	0000_0000h
34h	HW SIM General Purpose Register 2 (GPR2)	32	RW	0000_0000h
38h	HW SIM General Purpose Register 3 (GPR3)	32	RW	0000_0000h
3Ch	HW SIM MISC Register 0 (MISC_CTRL0)	32	RW	0000_0001h
50h	SIM DGO Control Register 0 (SIM_DGO_CTRL0)	32	RW	0000_0000h
54h	SIM DGO Control Register 1 (SIM_DGO_CTRL1)	32	RW	0000_0000h
58h	SIM DGO General Purpose Register 1 (SIM_DGO_GP1)	32	RW	0000_0000h
5Ch	SIM DGO general Purpose Register 2 (SIM_DGO_GP2)	32	RW	0000_0000h
60h	SIM DGO General Purpose Register 3 (SIM_DGO_GP3)	32	RW	0000_0000h
64h	SIM DGO General Purpose Register 4 (SIM_DGO_GP4)	32	RW	0000_0000h
68h	SIM DGO General Purpose Register 5 (SIM_DGO_GP5)	32	RW	0000_0000h
6Ch	SIM DGO General Purpose Register 6 (SIM_DGO_GP6)	32	RW	0000_0000h
70h	SIM DGO General Purpose Register 7 (SIM_DGO_GP7)	32	RW	0000_0000h
74h	SIM DGO General Purpose Register 8 (SIM_DGO_GP8)	32	RW	0000_0000h
78h	SIM DGO General Purpose Register 9 (SIM_DGO_GP9)	32	RW	0000_0000h
7Ch	SIM DGO General Purpose Register 10 (SIM_DGO_GP10)	32	RW	0000_0000h
80h	SIM DGO General Purpose Register 11 (SIM_DGO_GP11)	32	RW	0000_0000h
88h	WKPU Wake-up Enable (WKPU_WAKEUP_EN)	32	RW	0E0F_C7FFh
8Ch	Mirror of JTAG ID Register (JTAG_ID_REG)	32	RO	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
90h	Lower A7 TS Timer compare value (A7_TSTMR_CMP_VAL_L)	32	RW	0000_0000h
94h	Upper A7 TS Timer compare value (A7_TSTMR_CMP_VAL_H)	32	RW	0000_0000h
98h	Override Control for Compensation Codes (COMP_CELL_OVERRIDE)	32	RW	See description.

31.5.1.2 SIM Systems Options Register 1 (SOPT1)

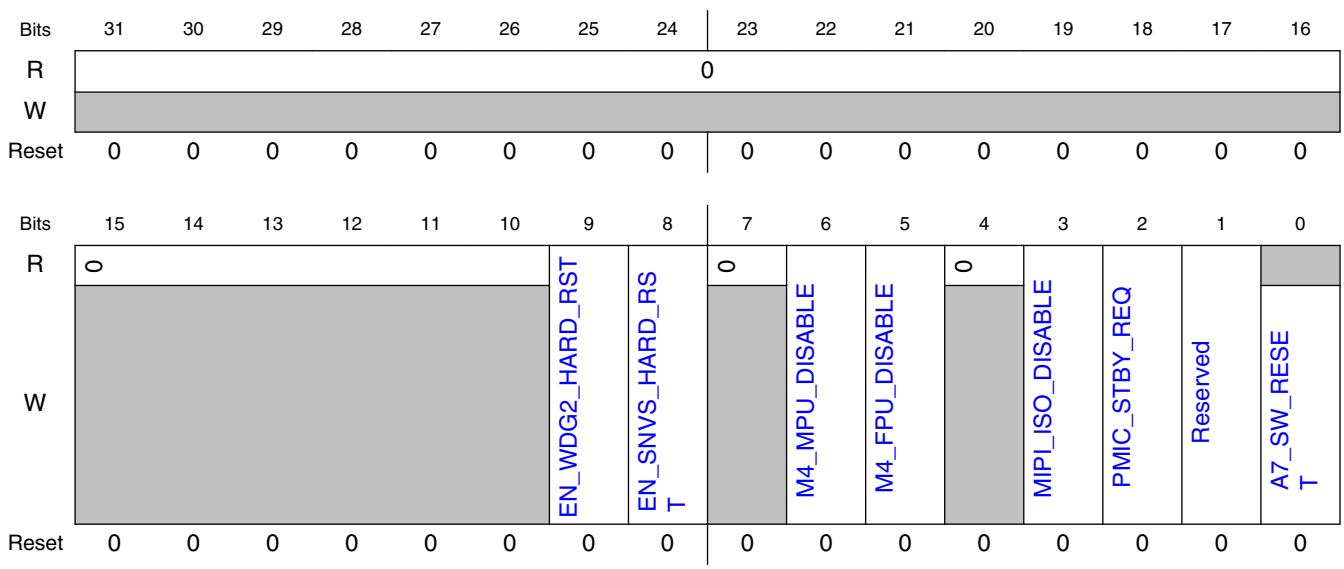
31.5.1.2.1 Offset

Register	Offset
SOPT1	0h

31.5.1.2.2 Function

This control register provides System Options for various functions on i.MX7ULP.

31.5.1.2.3 Diagram



31.5.1.2.4 Fields

Field	Function
31-10 —	Reserved.
9 EN_WDG2_HARD_RST	Watchdog 2 Reset Enable. 1'b0 : WDG2 system reset disabled 1'b1 : WDG2 system reset enabled
8 EN_SNVS_HARD_RST	SNVS_HP system reset enable (Write Once). 1'b0 : SNVS_HP system reset disabled 1'b1 : SNVS_HP system reset enabled
7 —	Reserved.
6 M4_MPU_DISABLE	Disables M4 MPU unit. 1'b0 : MPU enabled 1'b1 : MPU disabled
5 M4_FPU_DISABLE	Disables M4 FPU unit. 1'b0 : FPU enabled 1'b1 : FPU disabled
4 —	Reserved.
3 MIPI_ISO_DISABLE	MIPI Isolation Disable. To disable isolation to MIPI core for the cases where MIPI is powered up. Initially isolation will come as enabled. 1'b0 - MIPI isolation enabled. In this scenario the recommendation is to have VDD_DSI and VDD11_DSI connected to ground. 1'b1 - MIPI isolation disabled.
2 PMIC_STBY_REQ	PMIC Standby Request. This bit defines PMIC_STBY_REQ pin, which notifies external power management IC to move from functional voltage to standby voltage. 1'b0 : PMIC_STBY_REQ will negate - functional voltage 1'b1 : PMIC_STBY_REQ will assert - Voltage will be requested to change to standby voltage
1 —	Reserved.
0 A7_SW_RESET	SW reset for A7 domain. 1'b0 : No action 1'b1 : Writing value 1'b1 triggers A7 domain reset (even if previous bit value was 1'b1). **NOTE:** It is recommended to clear the bit with additional register write after bit is set, to prevent further SOPT1 read-modify-write to trigger undesirable reset.

31.5.1.3 SIM SOPT1 Configuration Register (SOPT1CFG)

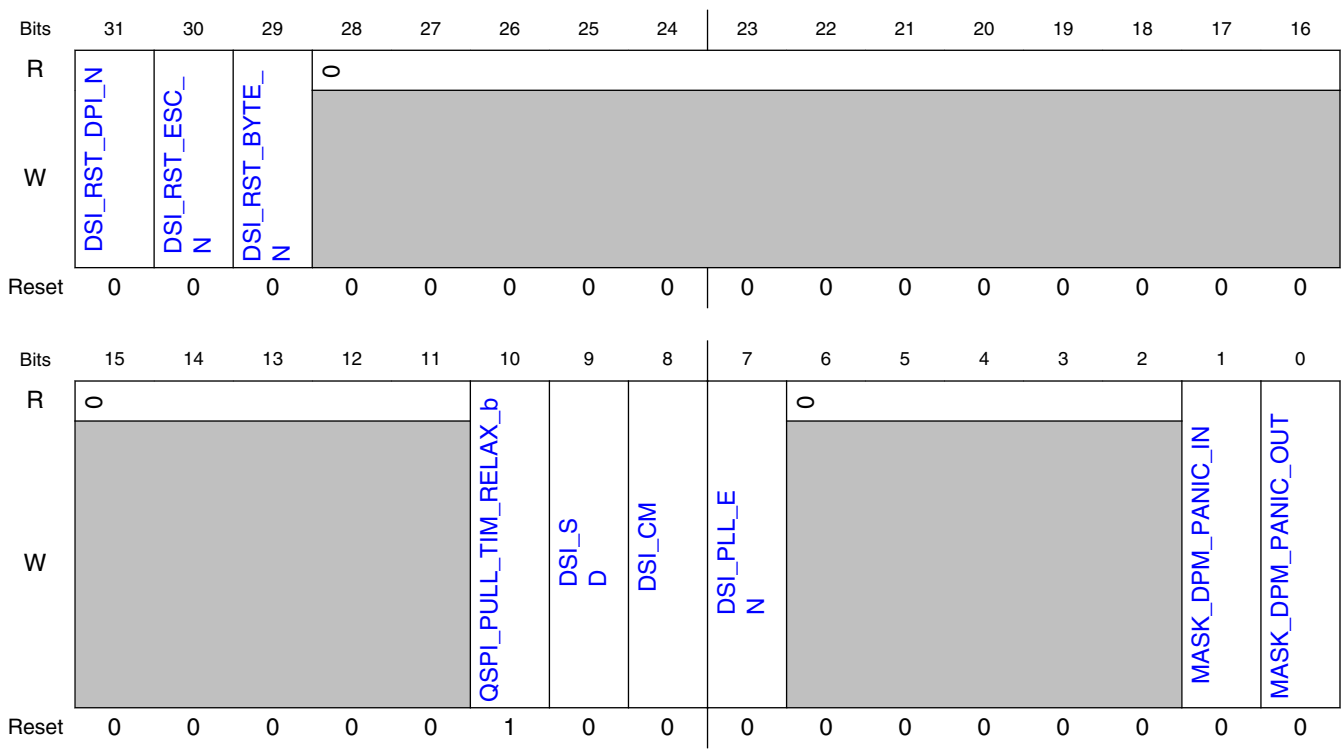
31.5.1.3.1 Offset

Register	Offset
SOPT1CFG	4h

31.5.1.3.2 Function

This register is used to configure system options on i.MX7ULP1.

31.5.1.3.3 Diagram



31.5.1.3.4 Fields

Field	Function
31 DSI_RST_DPI_N	DSI Reset DPI Control. The DSI Host Controller initialization requires that all its reset sources to be controlled independently in order to have its functional blocks coming out of reset in different times. This reset source affects all logic in the DPI domain. Coming out of system reset, the DPI domain will be in reset state, and this bit shall be configured to take this domain out of reset state. See MIPI-DSI chapter for further detail on how resets should be released to correct operation of the module.

Table continues on the next page...

Field	Function
	1'b0: DPI domain is under reset state (default). 1'b1: DPI domain is out of reset state.
30 DSI_RST_ESC_N	DSI Reset Escape Control. The DSI Host Controller initialization requires that all its reset sources be controlled independently in order to have its functional blocks coming out of reset in different times. This reset source affects all logic in the Escape domain. Coming out of system reset, the Escape domain will be in reset state, and this bit shall be configured to take this domain out of reset state. See MIPI-DSI chapter for further detail of how resets should be released to correct operation of the module. 1'b0: Escape domain is under reset state (default). 1'b1: Escape domain is out of reset state.
29 DSI_RST_BYTE_N	DSI Reset Byte Control. The DSI Host Controller initialization requires that all its reset sources be controlled independently in order to have its functional blocks coming out of reset in different times. This reset source affects all logic in the Byte domain. Coming out of system reset, the Byte domain will be in reset state, and this bit shall be configured to take this domain out of reset state. See DSI chapter for further detail of how resets should be released to correct operation of the module. 1'b0: Byte domain is under reset state (default). 1'b1: Byte domain is out of reset state.
28-11 —	Reserved.
10 QSPI_PULL_TIM_RELAX_b	QSPI OBE Assertion Control. This feature is used to pull by half a cycle the assertion of the Output Buffer Enable (OBE) for the QSPIA_DATA1 port. No effect on the de-assertion phase. This feature does not affect the OBE configurations for the other QSPI ports. 1'b0: OBE assertion is pulled by a half a cycle. OBE assertion is generated off the inverted internal reference clock of QSPI (ipg_clk_sfif_b). 1'b1: OBE assertion is not relaxed. OBE assertion is generated off the internal reference clock of QSPI (ipg_clk_sfif) (default).
9 DSI_SD	DSI Shutdown Control. Shutdown Control for Type-4 Display only. This feature triggers the display module to shut down the streaming video interface to reduce power consumption. 1'b0: Shutdown command not to be sent to the Type-4 display (default). 1'b1: Shutdown command to be sent to the Type-4 display.
8 DSI_CM	DSI Color Mode Control. Color Mode Control for Type-4 Display only. This feature is used to configure the display on Normal Mode or Low-color mode. When bit is configured to low-color mode, display works in 8-bit mode and display power is reduced. 1'b0: Normal mode (full color) (default). 1'b1: Low color mode (8-bit).
7 DSI_PLL_EN	DSI PLL Enable. This configuration enables the DSI PLL. By default, in order to avoid any power issues during power up, the DSI PLL is disabled. 1'b0: DSI PLL disabled 1'b1: DSI PLL enabled

Table continues on the next page...

Field	Function
6-2 —	Reserved.
1 MASK_DPM_PANIC_IN	DPM Panic In Mask (Write Once). Indicates if Panic Input from PMC should be ignored. 1'b0: Not Masked 1'b1: Masked
0 MASK_DPM_PANIC_OUT	DPM Panic Out Mask (Write Once). Indicates if Panic alarm from DPM is to be masked. 1'b0: Not Masked 1'b1: Masked

31.5.1.4 HW SIM HSIC CAL Register (HSIC_CAL)

31.5.1.4.1 Offset

Register	Offset
HSIC_CAL	Ch

31.5.1.4.2 Function

Calibration Settings for HSIC DDR pads.

DDR pads have calibration settings, which can be performed either automatically (via HW) or manually (via SW). An example of automatic calibration is the one performed by MMDC module.

Manual calibration is provided via SIM. The process then consists in carrying out an automatic calibration via MMDC, read the values from that calibration, and use them to set up HSIC DDR pads via SIM registers.

NOTE

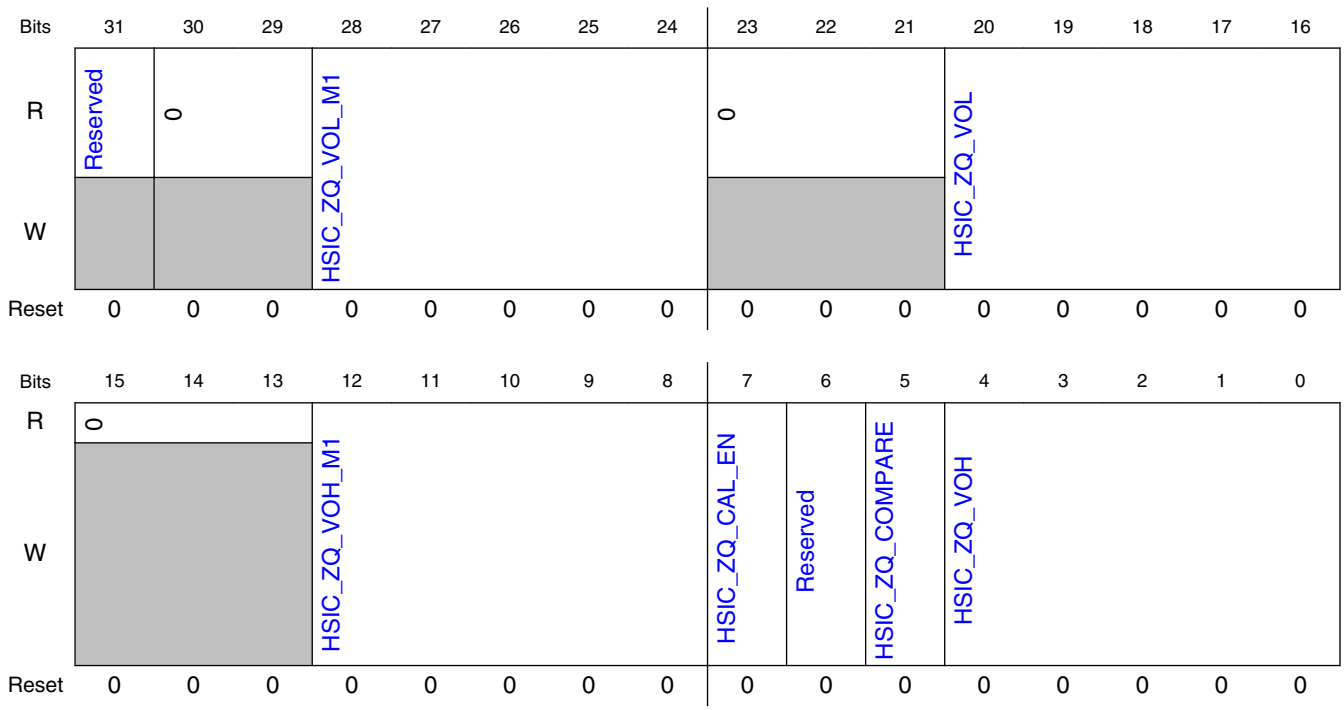
Check the "Calibration Process" section in MMDC chapter for details on automatic calibration.

Calibration procedure:

- Copy MMDC_MPZQSWCTRL or MMDC_MPZQHWCTRL to the corresponding HSIC_CAL bit fields + HSIC_ZQ_COMPARE = 1'b1

- Copy MMDC_MPZQSWCTRL or MMDC_MPZQHWCTRL to the corresponding HSIC_CAL bit fields + HSIC_ZQ_COMPARE = 1'b0
- This is required to latch the values to the pads through HSIC_ZQ_CAL_EN pulse.

31.5.1.4.3 Diagram



31.5.1.4.4 Fields

Field	Function
31 —	Reserved.
30-29 —	Reserved.
28-24 HSIC_ZQ_VOL_M1	HSIC ZQ pull-down resistance incremented. This field determines the value of the Pull-down resistor during SW ZQ calibration. Should be copied from MMDC_MPZQSWCTRL[ZQ_SW_PD_VAL] bit + 5'h01. 5'b00000 - Minimum resistance. 5'b11111 - Maximum resistance.
23-21 —	Reserved.
20-16 HSIC_ZQ_VOL	HSIC ZQ pull-down resistance. This field determines the value of the Pull-down resistor during SW ZQ calibration.

Table continues on the next page...

Field	Function
	Should be copied from MMDC_MPZQSWCTRL[ZQ_SW_PD_VAL] bitfield. 5'b00000 - Minimum resistance. 5'b11111 - Maximum resistance.
15-13 —	Reserved.
12-8 HSIC_ZQ_VOH_M1	HSIC ZQ pull-up resistance incremented. This field determines the value of the Pull-up resistor during SW ZQ calibration. Should be copied from MMDC_MPZQSWCTRL[ZQ_SW_PU_VAL] bit + 5'h01. 5'b00000 - Minimum resistance. 5'b11111 - Maximum resistance.
7 HSIC_ZQ_CAL_EN	HSIC ZQ compare enable. Controls the latching of the compare result.
6 —	Reserved.
5 HSIC_ZQ_COMPARE	HSIC ZQ Compare. Not Used.
4-0 HSIC_ZQ_VOH	HSIC ZQ pull-up resistance. This field determines the value of the Pull-up resistor during SW ZQ calibration. Should be copied from MMDC_MPZQSWCTRL[ZQ_SW_PU_VAL] bitfield. 5'b00000 - Minimum resistance. 5'b11111 - Maximum resistance.

31.5.1.5 SNVS Misc Control Register (SNVS_MISC_CTRL)

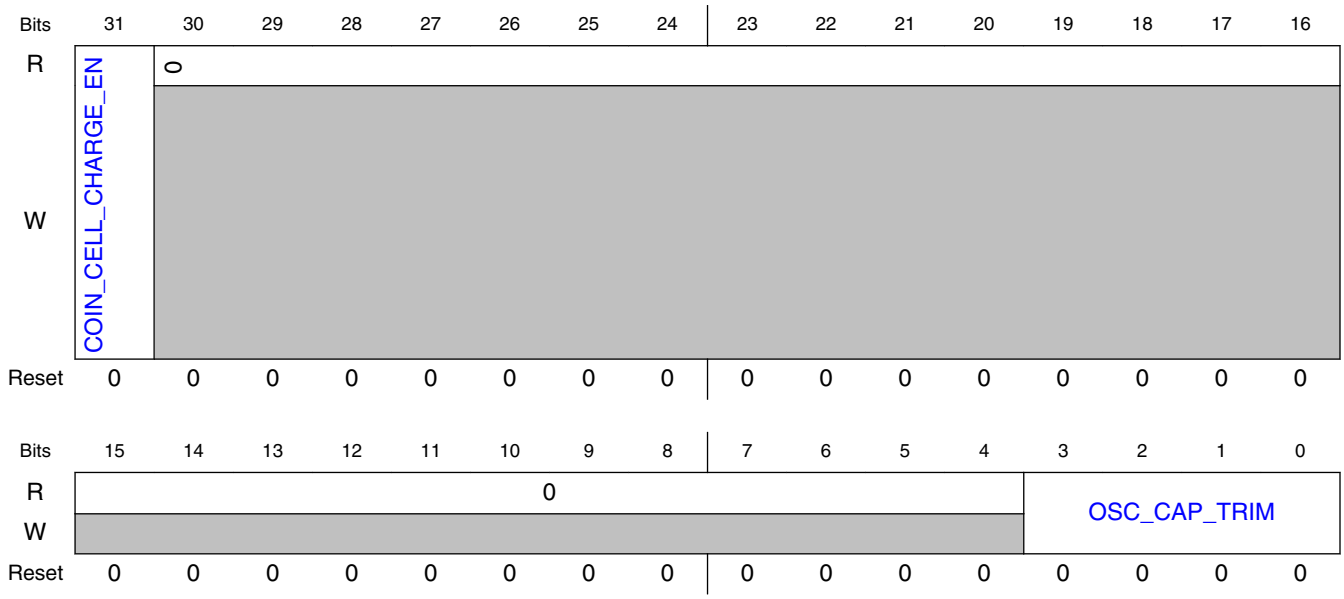
31.5.1.5.1 Offset

Register	Offset
SNVS_MISC_CTRL	28h

31.5.1.5.2 Function

This register defines additional control for the SNVS domain.

31.5.1.5.3 Diagram



31.5.1.5.4 Fields

Field	Function
31 COIN_CELL_CHARGE_EN	Enables Signal Isolation on SNVS Software Trims and Pull controls (PUS/PUE) until the software writes "1" to this register field. This ensures trim values are held until registers get updated by software after boot up. 1'b0 : Enable Isolation on Software Trims/Pull controls. 1'b1 : Remove Isolation on Software Trims/Pull controls. **NOTE:** 1. Software must remove isolation by writing "1" on every power-up. 2. Software must write "0" to this bit every time application enters LPSR mode where SoC power is removed.
30-4 —	Reserved.
3-0 OSC_CAP_TRIM	Trims to control the CAP on 32K Oscillator.

31.5.1.6 HW SIM General Purpose Register 0 (GPR0)

31.5.1.6.1 Offset

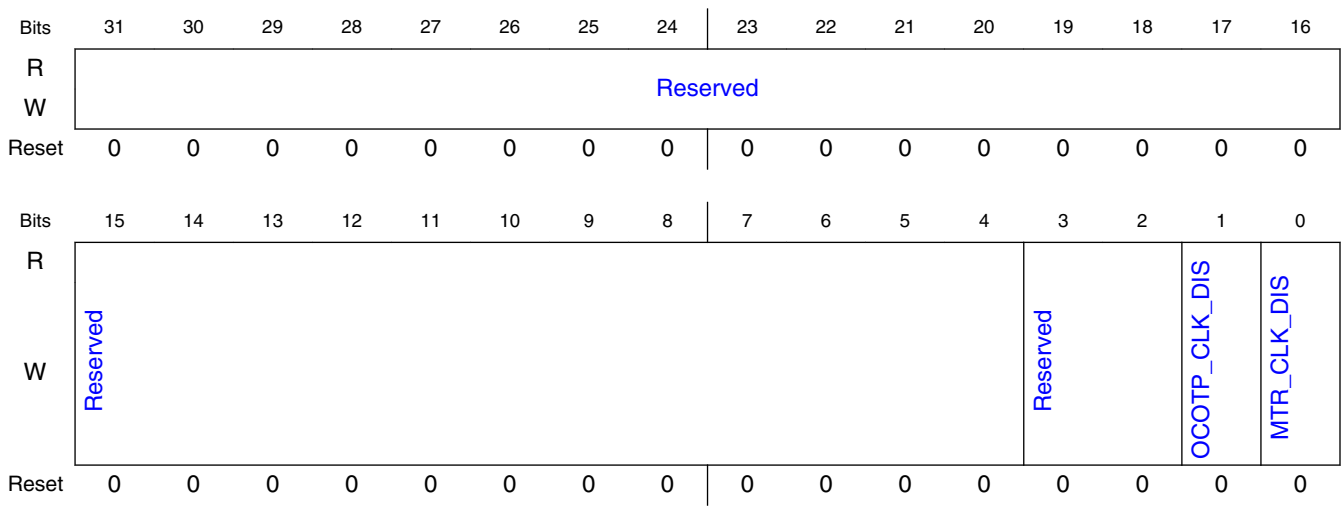
Register	Offset
GPR0	2Ch

31.5.1.6.2 Function

General Purpose Read/Write Register 0.

Hardware GPR to store chip specific control bits.

31.5.1.6.3 Diagram



31.5.1.6.4 Fields

Field	Function
31-4 —	Reserved.
3-2 —	Reserved.
1 OCOTP_CLK_DIS	OCOTP clocks disable. Disables OCOTP clocks to save power. 1'b0: OCOTP clock is enabled. 1'b1: OCOTP clock is disabled.
0 MTR_CLK_DIS	MBIST clocks disable. Disables MBIST clocks to save power.

Field	Function
	1'b0: MBIST clocks are enabled. 1'b1: MBIST clocks are disabled.

31.5.1.7 HW SIM General Purpose Register 1 (GPR1)

31.5.1.7.1 Offset

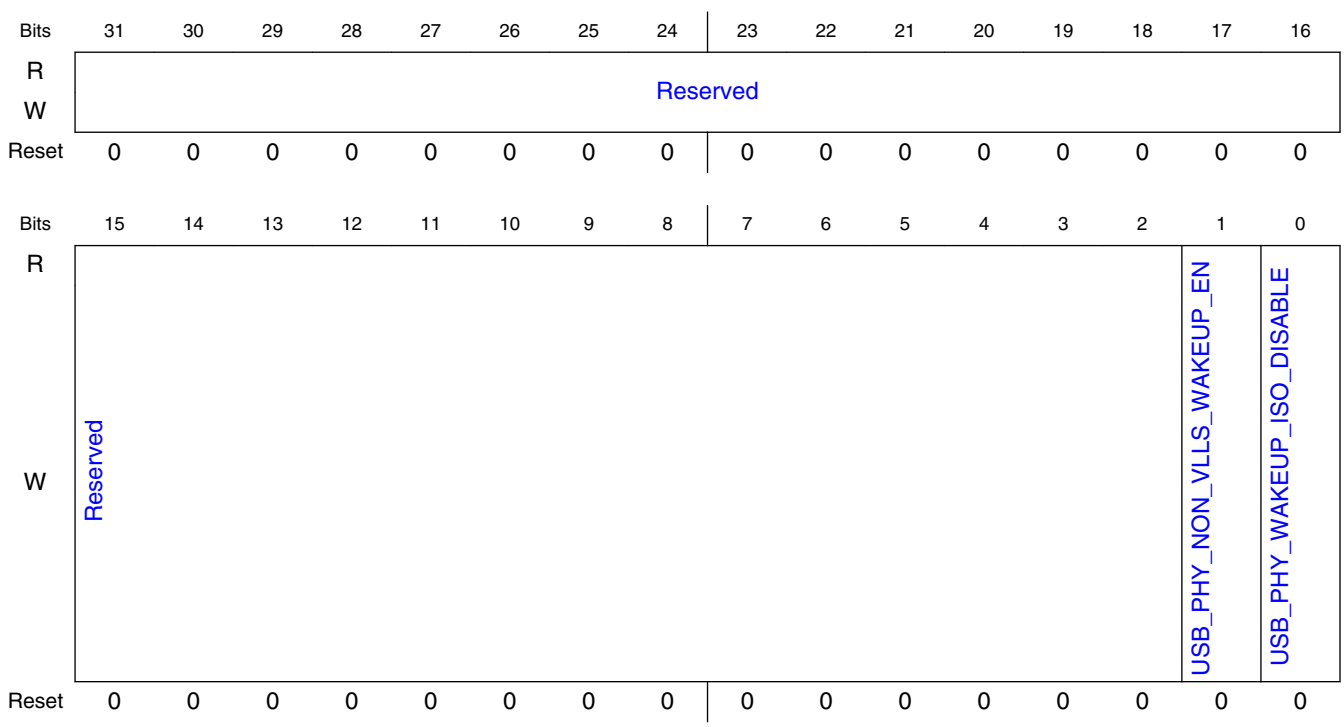
Register	Offset
GPR1	30h

31.5.1.7.2 Function

General Purpose Read/Write Register 1.

Hardware GPR to store chip specific control bits.

31.5.1.7.3 Diagram



31.5.1.7.4 Fields

Field	Function
31-2 —	Reserved.
1 USB_PHY_NON_VLLS_WAKEUP_EN	USB PHY non VLLS wakeup enable. 1'b0 : Disabled. 1'b1 : Enabled.
0 USB_PHY_WAKEUP_ISO_DISABLE	USB PHY wakeup ISO disable. USB PHY VLLS wakeup source pins isolation control. Wakeup pins are powered by VDD_USB18 and wakeup logic is powered by VDD_PMC18. 1'b0 : Wakeup pins are isolated from wakeup logic. 1'b1 : Wakeup pins are connected to wakeup logic.

31.5.1.8 HW SIM General Purpose Register 2 (GPR2)

31.5.1.8.1 Offset

Register	Offset
GPR2	34h

31.5.1.8.2 Function

General Purpose Read/Write Register 2.

Hardware GPR to store chip specific control bits.

31.5.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.8.4 Fields

Field	Function
31-0 —	Reserved.

31.5.1.9 HW SIM General Purpose Register 3 (GPR3)

31.5.1.9.1 Offset

Register	Offset
GPR3	38h

31.5.1.9.2 Function

General Purpose Read/Write Register 3.

Hardware GPR to store chip specific control bits.

31.5.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.9.4 Fields

Field	Function
31-0 —	Reserved.

31.5.1.10 HW SIM MISC Register 0 (MISC_CTRL0)

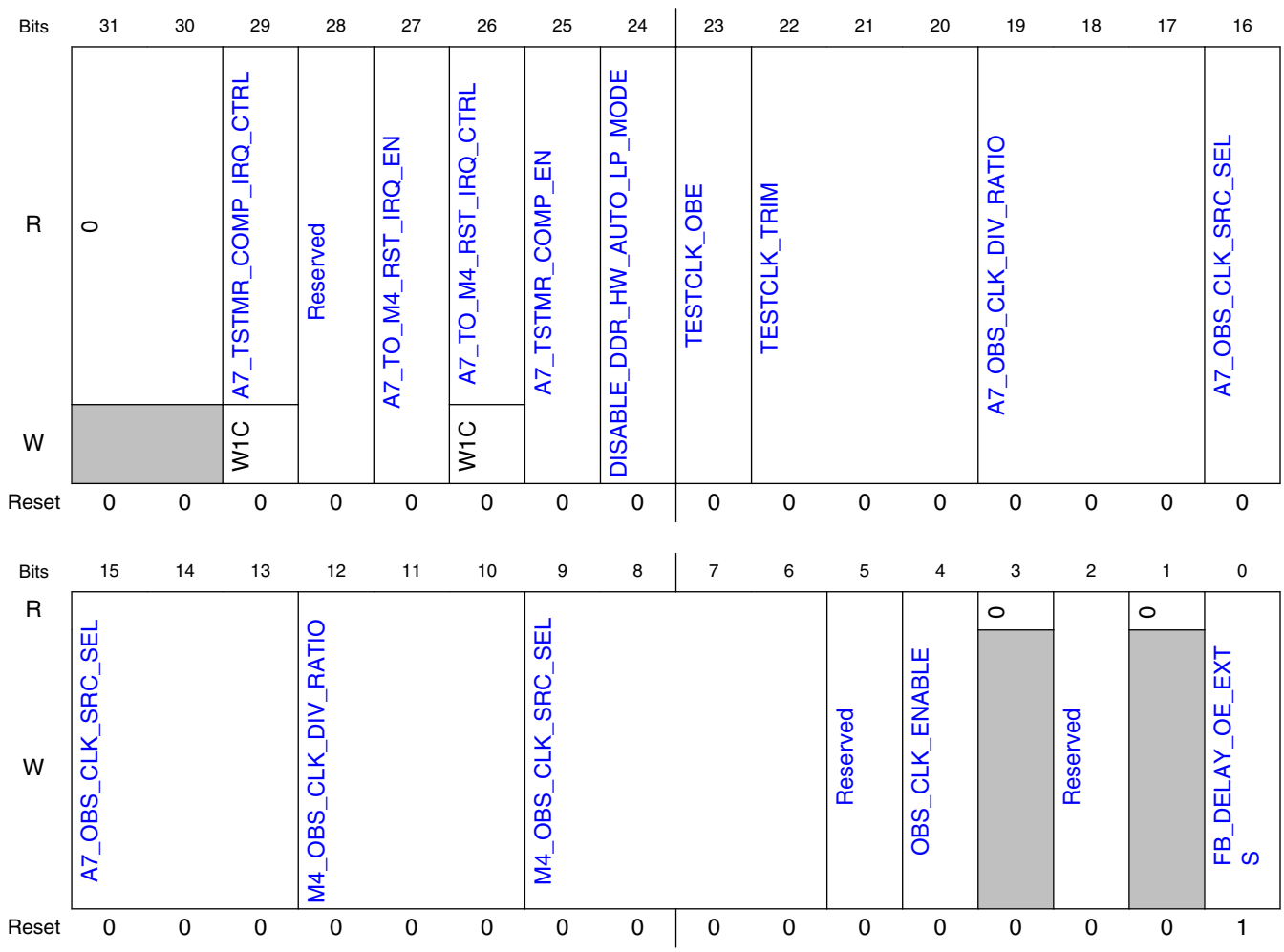
31.5.1.10.1 Offset

Register	Offset
MISC_CTRL0	3Ch

31.5.1.10.2 Function

This register is used to store Miscellaneous Control Bits.

31.5.1.10.3 Diagram



31.5.1.10.4 Fields

Field	Function
31-30 —	Reserved.
29 A7_TSTMR_COMP_IRQ_CTRL	Controls the compare of A7 reset as IRQ. This bit is set when A7 TSTimer counter value matches the value stored in A7_TSTMR_CMP_VAL registers. Staying high until a 1'b1 bit write. **NOTE:** This bit should be cleared in the interrupt routine.
28 —	Reserved.
27 A7_TO_M4_RST_IRQ_EN	Enables the assertion of A7 reset as IRQ/Wake-up to M4. If set this bit enables the A7 reset as IRQ/Wake-up source to M4. 1'b0: Compare operation is disabled. 1'b1: Compare operation is enabled.
26 A7_TO_M4_RST_IRQ_CTRL	Controls the assertion of A7 reset as IRQ/Wake-up to M4. This bit is set when A7 reset occurs. Staying high until a 1'b1 bit write.
25 A7_TSTMR_COMP_EN	Enables the compare of A7 TS Timer versus a programmed value. If set, this bit enables the compare of A7 TS Timer versus the value stored in SIM registers A7_TSTMR_CMP_VAL_L and A7_TSTMR_CMP_VAL_H. 1'b0: Compare operation is disabled. 1'b1: Compare operation is enabled.
24 DISABLE_DDR_HW_AUTO_LP_MODE	Disable control to put DDR in self-refresh mode automatically by HW during low power modes. 1'b0: Self-refresh mode enabled. 1'b1: Self-refresh mode disabled.
23 TESTCLK_OBE	Output Buffer Enable for LVDS TESTCLK_P/TESTCLK_N pad.
22-20 TESTCLK_TRIM	Trim Delay for LVDS TESTCLK_P/TESTCLK_N pad. TESTCLK_TRIM[1:0]: Control pins for output current trimming (process case compensation): 1'b11: Best case. 1'b10: Typical case. 1'b00: Worst case. **NOTE:** TESTCLK_TRIM[2]: Not used.
19-17 A7_OBS_CLK_DIV_RATIO	Selection of division rate (2**n) of A7 observation clock div rate = 2**n (n=0..7). 3'b000: divide by 1. 3'b001: divide by 2. 3'b010: divide by 4. 3'b011: divide by 8. 3'b100: divide by 16.

Table continues on the next page...

Field	Function
	3'b101: divide by 32. 3'b110: divide by 64. 3'b111: divide by 128.
16-13 A7_OBS_CLK_SRC_SEL	Selects the clock to be observed on A7 domain. 4'b0000 - PMC0_TST_DIG_OUT 4'b0001 - PMC1_TST_DIG_OUT 4'b0010 - Reserved 4'b0011 - Reserved 4'b0100 - Reserved 4'b0101 - PLL2 VCO **NOTE:** PLL2 VCO comes from scg_spll_clk signal (after plls mux sel) differently of the PFD signals. 4'b0110 - PLL2 PFD0 4'b0111 - PLL2 PFD1 4'b1000 - PLL2 PFD2 4'b1001 - PLL2 PFD3 4'b1010 - PLL3 VCO **NOTE:** PLL3 VCO comes from scg_apll_clk signal (after plls mux sel) differently of the PFD signals. 4'b1011 - PLL3 PFD0 4'b1100 - PLL3 PFD1 4'b1101 - PLL3 PFD2 4'b1110 - PLL3 PFD3 4'b1111 - Reserved
12-10 M4_OBS_CLK_DIV_RATIO	Selection of division rate (2**n) of M4 observation clock div rate = 2**n (n=0..7). 3'b000: divide by 1. 3'b001: divide by 2. 3'b010: divide by 4. 3'b011: divide by 8. 3'b100: divide by 16. 3'b101: divide by 32. 3'b110: divide by 64. 3'b111: divide by 128.
9-6 M4_OBS_CLK_SRC_SEL	Selects the clock to be observed on M4 domain. 4'b0000 - LPO1KHZ clock 4'b0001 - SIRC clock 4'b0010 - FIRC clock 4'b0011 - System OSC 4'b0100 - RTC OSC - 32KHz 4'b0101 - PLL0 VCO **NOTE:** PLL0 VCO comes from scg_spll_clk signal (after plls mux sel) differently of the PFD signals.

Table continues on the next page...

Field	Function
	4'b0110 - PLL0 PFD0 4'b0111 - PLL0 PFD1 4'b1000 - PLL0 PFD2 4'b1001 - PLL0 PFD3 4'b1010 - PLL1 VCO **NOTE:** PLL1 VCO comes from scg_apll_clk signal (after plls mux sel) differently of the PFD signals. 4'b1011 - PLL1 PFD0 4'b1100 - PLL1 PFD1 4'b1101 - PLL1 PFD2 4'b1110 - PLL1 PFD3 4'b1111 - Reserved
5 —	Reserved.
4 OBS_CLK_ENABLE	Enable that clocks can be observed at M4 CLKOUT, A7 CLKOUT and LVDS clock pads. 1'b0 : Clock observation feature disabled. 1'b1 : Clock observation feature enabled.
3 —	Reserved.
2 —	Reserved.
1 —	Reserved.
0 FB_DELAY_OE_EXTS	Flexbus control of FB_OE_b signal delay when CSCR[EXTS] is enabled. 1'b0: Delay disabled. 1'b1: Delay enabled.

31.5.1.11 SIM DGO Control Register 0 (SIM_DGO_CTRL0)

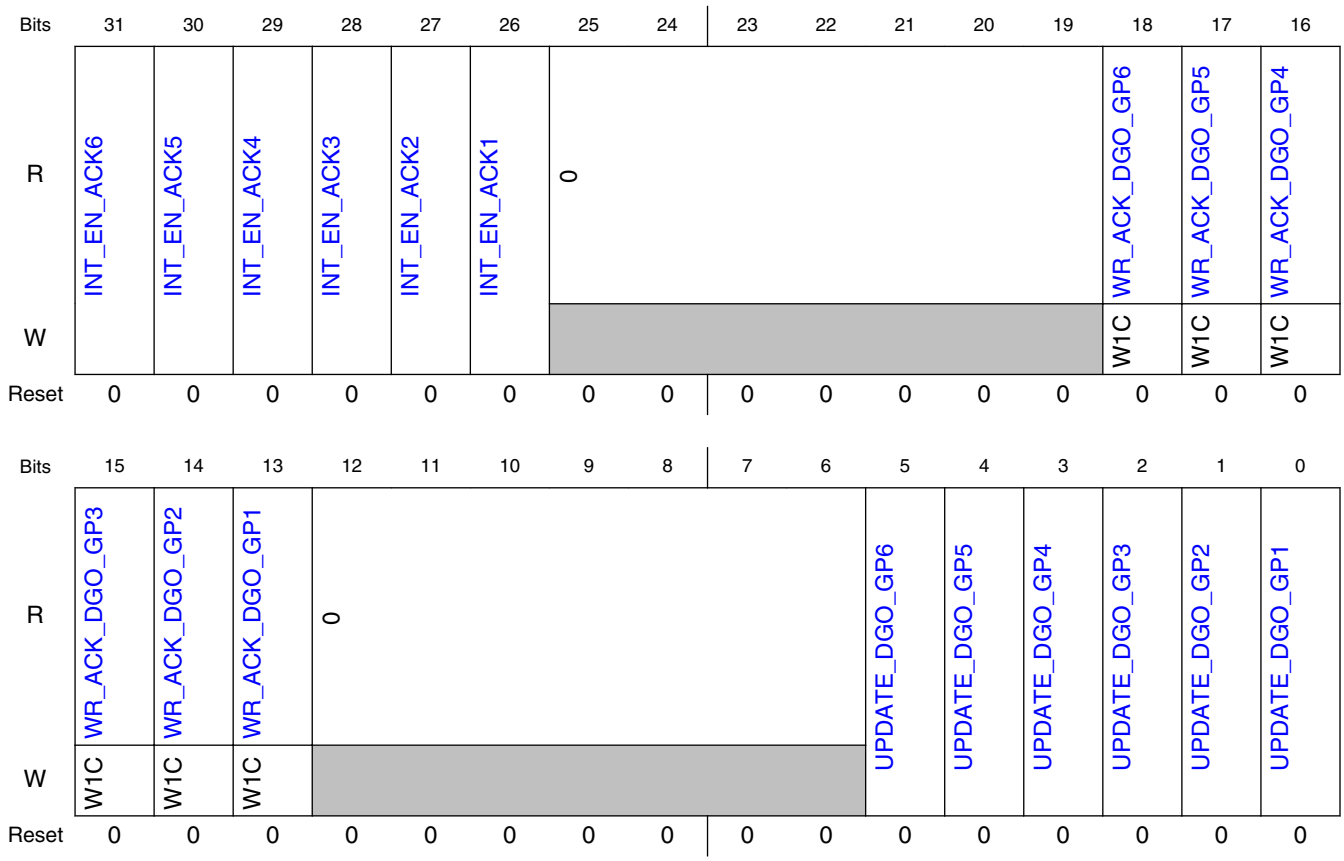
31.5.1.11.1 Offset

Register	Offset
SIM_DGO_CTRL0	50h

31.5.1.11.2 Function

Control Register for DGO GPs 1 to 6.

31.5.1.11.3 Diagram



31.5.1.11.4 Fields

Field	Function
31 INT_EN_ACK6	Interrupt enable for WR_ACK_DGO_GP6 bit field.
30 INT_EN_ACK5	Interrupt enable for WR_ACK_DGO_GP5 bit field.
29 INT_EN_ACK4	Interrupt enable for WR_ACK_DGO_GP4 bit field.
28 INT_EN_ACK3	Interrupt enable for WR_ACK_DGO_GP3 bit field.
27 INT_EN_ACK2	Interrupt enable for WR_ACK_DGO_GP2 bit field.
26 INT_EN_ACK1	Interrupt enable for WR_ACK_DGO_GP1 bit field.
25-19	Reserved.

Table continues on the next page...

Field	Function
—	
18 WR_ACK_DGO_GP6	This bit field is set automatically when corresponding DGO_GP6 register is shadowed in DGO domain. SW needs to write "1" to clear.
17 WR_ACK_DGO_GP5	This bit field is set automatically when corresponding DGO_GP5 register is shadowed in DGO domain. SW needs to write "1" to clear.
16 WR_ACK_DGO_GP4	This bit field is set automatically when corresponding DGO_GP4 register is shadowed in DGO domain. SW needs to write "1" to clear.
15 WR_ACK_DGO_GP3	This bit field is set automatically when corresponding DGO_GP3 register is shadowed in DGO domain. SW needs to write "1" to clear.
14 WR_ACK_DGO_GP2	This bit field is set automatically when corresponding DGO_GP2 register is shadowed in DGO domain. SW needs to write "1" to clear.
13 WR_ACK_DGO_GP1	This bit field is set automatically when corresponding DGO_GP1 register is shadowed in DGO domain. SW needs to write "1" to clear.
12-6 —	Reserved.
5 UPDATE_DGO_GP6	Writing 1 to this bit field indicates corresponding DGO_GP6 register has been updated with new values.
4 UPDATE_DGO_GP5	Writing 1 to this bit field indicates corresponding DGO_GP5 register has been updated with new values.
3 UPDATE_DGO_GP4	Writing 1 to this bit field indicates corresponding DGO_GP4 register has been updated with new values.
2 UPDATE_DGO_GP3	Writing 1 to this bit field indicates corresponding DGO_GP3 register has been updated with new values.
1 UPDATE_DGO_GP2	Writing 1 to this bit field indicates corresponding DGO_GP2 register has been updated with new values.
0 UPDATE_DGO_GP1	Writing 1 to this bit field indicates corresponding DGO_GP1 register has been updated with new values.

31.5.1.12 SIM DGO Control Register 1 (SIM_DGO_CTRL1)

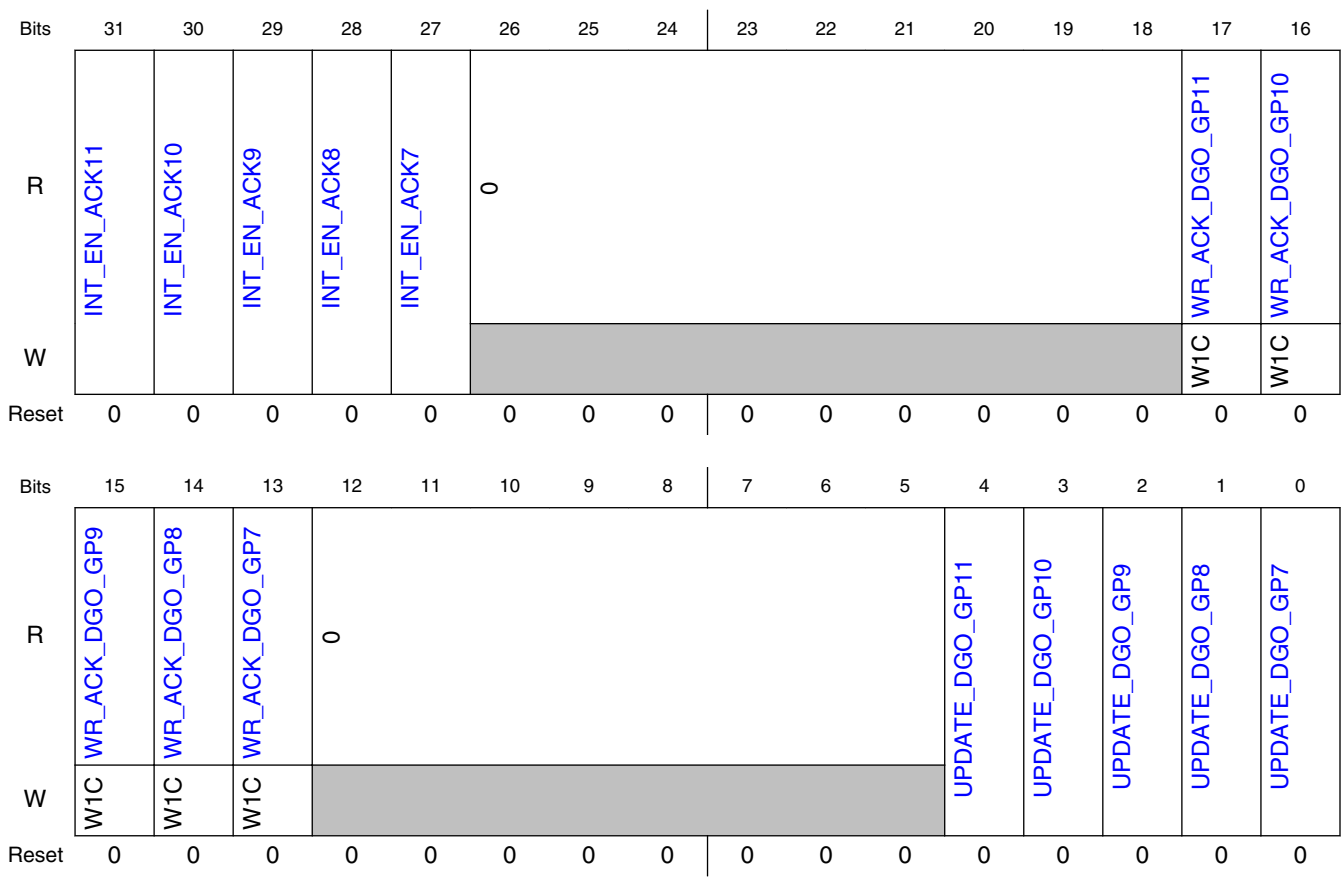
31.5.1.12.1 Offset

Register	Offset
SIM_DGO_CTRL1	54h

31.5.1.12.2 Function

Control Register for DGO GPs 7 to 11.

31.5.1.12.3 Diagram



31.5.1.12.4 Fields

Field	Function
31	Interrupt enable for WR_ACK_DGO_GP11 bit field.

Table continues on the next page...

Field	Function
INT_EN_ACK11	
30 INT_EN_ACK10	Interrupt enable for WR_ACK_DGO_GP10 bit field.
29 INT_EN_ACK9	Interrupt enable for WR_ACK_DGO_GP9 bit field.
28 INT_EN_ACK8	Interrupt enable for WR_ACK_DGO_GP8 bit field.
27 INT_EN_ACK7	Interrupt enable for WR_ACK_DGO_GP7 bit field.
26-18 —	Reserved.
17 WR_ACK_DGO_GP11	This bit field is set automatically when corresponding DGO_GP11 register is shadowed in DGO domain. SW needs to write "1" to clear.
16 WR_ACK_DGO_GP10	This bit field is set automatically when corresponding DGO_GP10 register is shadowed in DGO domain. SW needs to write "1" to clear.
15 WR_ACK_DGO_GP9	This bit field is set automatically when corresponding DGO_GP9 register is shadowed in DGO domain. SW needs to write "1" to clear.
14 WR_ACK_DGO_GP8	This bit field is set automatically when corresponding DGO_GP8 register is shadowed in DGO domain. SW needs to write "1" to clear.
13 WR_ACK_DGO_GP7	This bit field is set automatically when corresponding DGO_GP7 register is shadowed in DGO domain. SW needs to write "1" to clear.
12-5 —	Reserved.
4 UPDATE_DGO_GP11	Writing 1 to this bit field indicates corresponding DGO_GP11 register has been updated with new values.
3 UPDATE_DGO_GP10	Writing 1 to this bit field indicates corresponding DGO_GP10 register has been updated with new values.
2 UPDATE_DGO_GP9	Writing 1 to this bit field indicates corresponding DGO_GP9 register has been updated with new values.
1 UPDATE_DGO_GP8	Writing 1 to this bit field indicates corresponding DGO_GP8 register has been updated with new values.
0 UPDATE_DGO_GP7	Writing 1 to this bit field indicates corresponding DGO_GP7 register has been updated with new values.

31.5.1.13 SIM DGO General Purpose Register 1 (SIM_DGO_GP1)

31.5.1.13.1 Offset

Register	Offset
SIM_DGO_GP1	58h

31.5.1.13.2 Function

General Purpose Read/Write DGO Register 1.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.13.4 Fields

Field	Function
31-0 SIM_DGO_GP1	SIM DGO General purpose register 1. Contents are retained between power cycles

31.5.1.14 SIM DGO general Purpose Register 2 (SIM_DGO_GP2)

31.5.1.14.1 Offset

Register	Offset
SIM_DGO_GP2	5Ch

31.5.1.14.2 Function

General Purpose Read/Write DGO Register 2.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP2															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.14.4 Fields

Field	Function
31-0 SIM_DGO_GP2	SIM DGO General purpose register 2. Contents are retained between power cycles

31.5.1.15 SIM DGO General Purpose Register 3 (SIM_DGO_GP3)

31.5.1.15.1 Offset

Register	Offset
SIM_DGO_GP3	60h

31.5.1.15.2 Function

General Purpose Read/Write DGO Register 3.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP3															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.15.4 Fields

Field	Function
31-0 SIM_DGO_GP3	SIM DGO General purpose register 3. Contents are retained between power cycles

31.5.1.16 SIM DGO General Purpose Register 4 (SIM_DGO_GP4)

31.5.1.16.1 Offset

Register	Offset
SIM_DGO_GP4	64h

31.5.1.16.2 Function

General Purpose Read/Write DGO Register 4.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP4															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP4															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.16.4 Fields

Field	Function
31-0 SIM_DGO_GP4	SIM DGO General purpose register 4. Contents are retained between power cycles

31.5.1.17 SIM DGO General Purpose Register 5 (SIM_DGO_GP5)

31.5.1.17.1 Offset

Register	Offset
SIM_DGO_GP5	68h

31.5.1.17.2 Function

General Purpose Read/Write DGO Register 5.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP5															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP5															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.17.4 Fields

Field	Function
31-0 SIM_DGO_GP5	SIM DGO General purpose register 5. Contents are retained between power cycles

31.5.1.18 SIM DGO General Purpose Register 6 (SIM_DGO_GP6)

31.5.1.18.1 Offset

Register	Offset
SIM_DGO_GP6	6Ch

31.5.1.18.2 Function

General Purpose Read/Write DGO Register 6.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP6															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP6															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.18.4 Fields

Field	Function
31-0 SIM_DGO_GP6	SIM DGO General purpose register 6. Contents are retained between power cycles

31.5.1.19 SIM DGO General Purpose Register 7 (SIM_DGO_GP7)

31.5.1.19.1 Offset

Register	Offset
SIM_DGO_GP7	70h

31.5.1.19.2 Function

General Purpose Read/Write DGO Register 7.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP7															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP7															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.19.4 Fields

Field	Function
31-0 SIM_DGO_GP7	SIM DGO General purpose register 7. Contents are retained between power cycles

31.5.1.20 SIM DGO General Purpose Register 8 (SIM_DGO_GP8)

31.5.1.20.1 Offset

Register	Offset
SIM_DGO_GP8	74h

31.5.1.20.2 Function

General Purpose Read/Write DGO Register 8.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP8															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP8															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.20.4 Fields

Field	Function
31-0 SIM_DGO_GP8	SIM DGO General purpose register 8. Contents are retained between power cycles

31.5.1.21 SIM DGO General Purpose Register 9 (SIM_DGO_GP9)

31.5.1.21.1 Offset

Register	Offset
SIM_DGO_GP9	78h

31.5.1.21.2 Function

General Purpose Read/Write DGO Register 9.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP9															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP9															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.21.4 Fields

Field	Function
31-0 SIM_DGO_GP9	SIM DGO General purpose register 9. Contents are retained between power cycles

31.5.1.22 SIM DGO General Purpose Register 10 (SIM_DGO_GP10)

31.5.1.22.1 Offset

Register	Offset
SIM_DGO_GP10	7Ch

31.5.1.22.2 Function

General Purpose Read/Write DGO Register 10.

This register bank is reserved for ROM usage.

Contents are retained during power cycles.

31.5.1.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SIM_DGO_GP10															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SIM_DGO_GP10															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.22.4 Fields

Field	Function
31-0 SIM_DGO_GP10	SIM DGO General purpose register 10. Contents are retained between power cycles

31.5.1.23 SIM DGO General Purpose Register 11 (SIM_DGO_GP11)

31.5.1.23.1 Offset

Register	Offset
SIM_DGO_GP11	80h

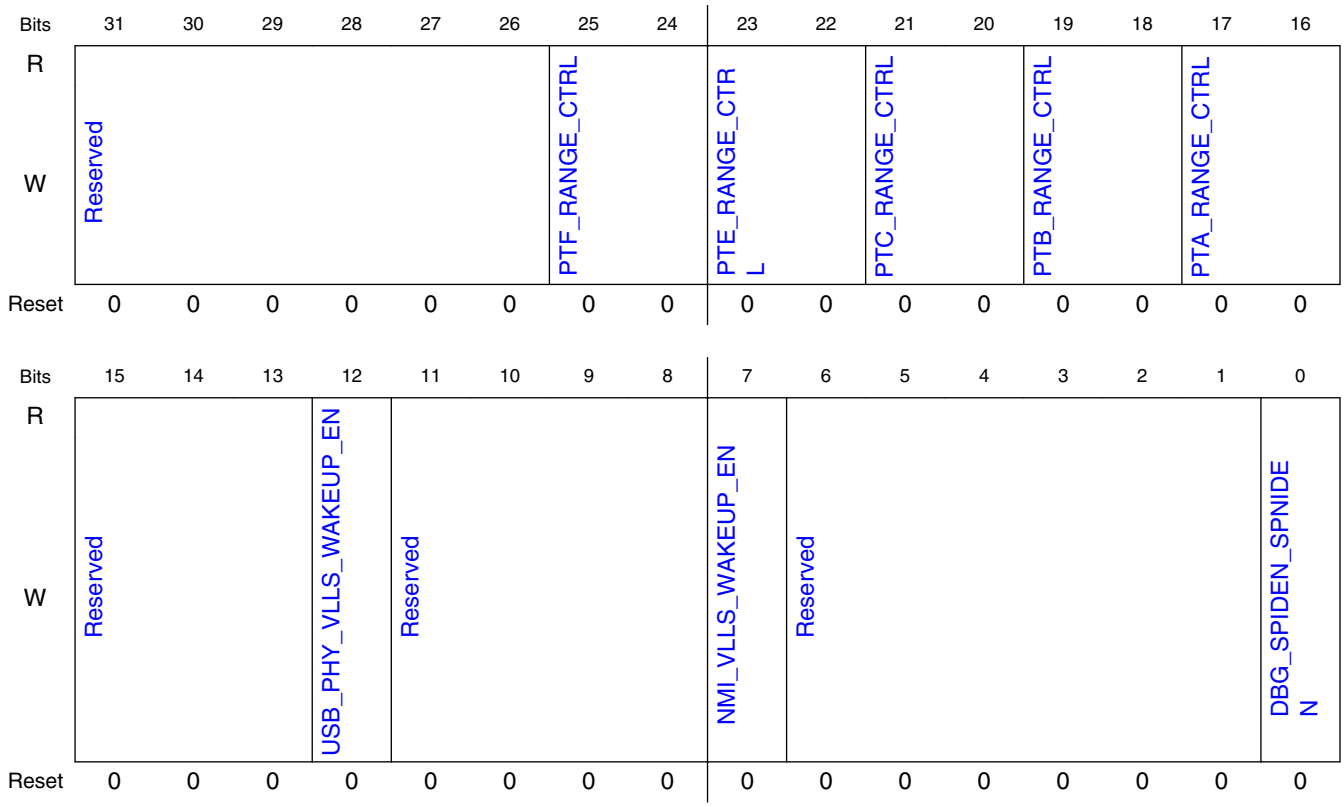
31.5.1.23.2 Function

General Purpose Read/Write DGO Register 11.

Contents are retained during power cycles.

WARNING: This DGO register is used as hardware GPR. Not intended to store ROM Code.

31.5.1.23.3 Diagram



31.5.1.23.4 Fields

Field	Function
31-26 —	Reserved.
25-24 PTF_RANGE_CTRL	PTF Range Control. 2'b00: Continuous Mode. 2'b01: Low Range Mode. 2'b10: High Range Mode. 2'b11: Prohibited. **NOTE:** Despite the access type this bit is implemented as Write-Once. **NOTE:** Reset value from fuse. See the Fusemap attached with this reference manual for more information. **NOTE:** See the section GPIO pads operating range configuration for additional details.
23-22 PTE_RANGE_CTRL	PTE Range Control. 2'b00: Continuous Mode. 2'b01: Low Range Mode. 2'b10: High Range Mode. 2'b11: Prohibited.

Table continues on the next page...

Field	Function
	NOTE: Despite the access type this bit is implemented as Write-Once. **NOTE:** Reset value from fuse. See the Fusemap attached with this reference manual for more information. **NOTE:** See the section GPIO pads operating range configuration for additional details.
21-20 PTC_RANGE_CTRL	PTC Range Control. 2'b00: Continuous Mode. 2'b01: Low Range Mode. 2'b10: High Range Mode. 2'b11: Prohibited. **NOTE:** Despite the access type this bit is implemented as Write-Once. **NOTE:** Reset value from fuse. See the Fusemap attached with this reference manual for more information. **NOTE:** See the section GPIO pads operating range configuration for additional details.
19-18 PTB_RANGE_CTRL	PTB Range Control. 2'b00: Continuous Mode. 2'b01: Low Range Mode. 2'b10: High Range Mode. 2'b11: Prohibited. **NOTE:** Despite the access type this bit is implemented as Write-Once. **NOTE:** Reset value from fuse. See the Fusemap attached with this reference manual for more information. **NOTE:** See the section GPIO pads operating range configuration for additional details.
17-16 PTA_RANGE_CTRL	PTA Range Control. 2'b00: Continuous Mode. 2'b01: Low Range Mode. 2'b10: High Range Mode. 2'b11: Prohibited. **NOTE:** Despite the access type this bit is implemented as Write-Once. **NOTE:** Reset value from fuse. See the Fusemap attached with this reference manual for more information. **NOTE:** See the section GPIO pads operating range configuration for additional details. section for additional details.
15-13 —	Reserved.
12 USB_PHY_VLLS_WAKEUP_EN	USB PHY VLLS wakeup enable. 1'b0 : Disabled. 1'b1 : Enabled.
11-8 —	Reserved.
7 NMI_VLLS_WAKEUP_EN	Enables NMI pin as VLLS wakeup source (not necessary for non-VLLS modes). 1'b0 : Disabled. 1'b1 : Enabled. **NOTE:** PTA9 pad must still be configured as NMI pin via the respective IOMUXC0 register.

Table continues on the next page...

Field	Function
6-1 —	Reserved.
0 DBG_SPIDEN_SPNIDEN	Debug secured access enable. Writes in this register defines the value of SPIDEN/SPNIDEN bits. See the section "Secured JTAG" in the "Debug" chapter of this manual.

31.5.1.24 WKPU Wake-up Enable (WKPU_WAKEUP_EN)

31.5.1.24.1 Offset

Register	Offset
WKPU_WAKEUP_EN	88h

31.5.1.24.2 Function

Enables for wake-up events available only in supervisor mode.

31.5.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WKPU_CH_WAKEUP_EN															
Reset	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WKPU_CH_WAKEUP_EN															
Reset	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1

31.5.1.24.4 Fields

Field	Function
31-0 WKPU_CH_WAKEUP_EN	Bit used to disable wake-up events through this channel. Each channel may have one or multiple wake-up sources associated to it. This register holds a one-to-one relationship with WKPU channels. WKPU_CH_WAKEUP_EN[n] - Enables WKPU Channel n. With n = {0 .. 31}

Field	Function
	1'b0 - wake-up event disabled through this channel. 1'b1 - wake-up event enabled through this channel.

31.5.1.25 Mirror of JTAG ID Register (JTAG_ID_REG)

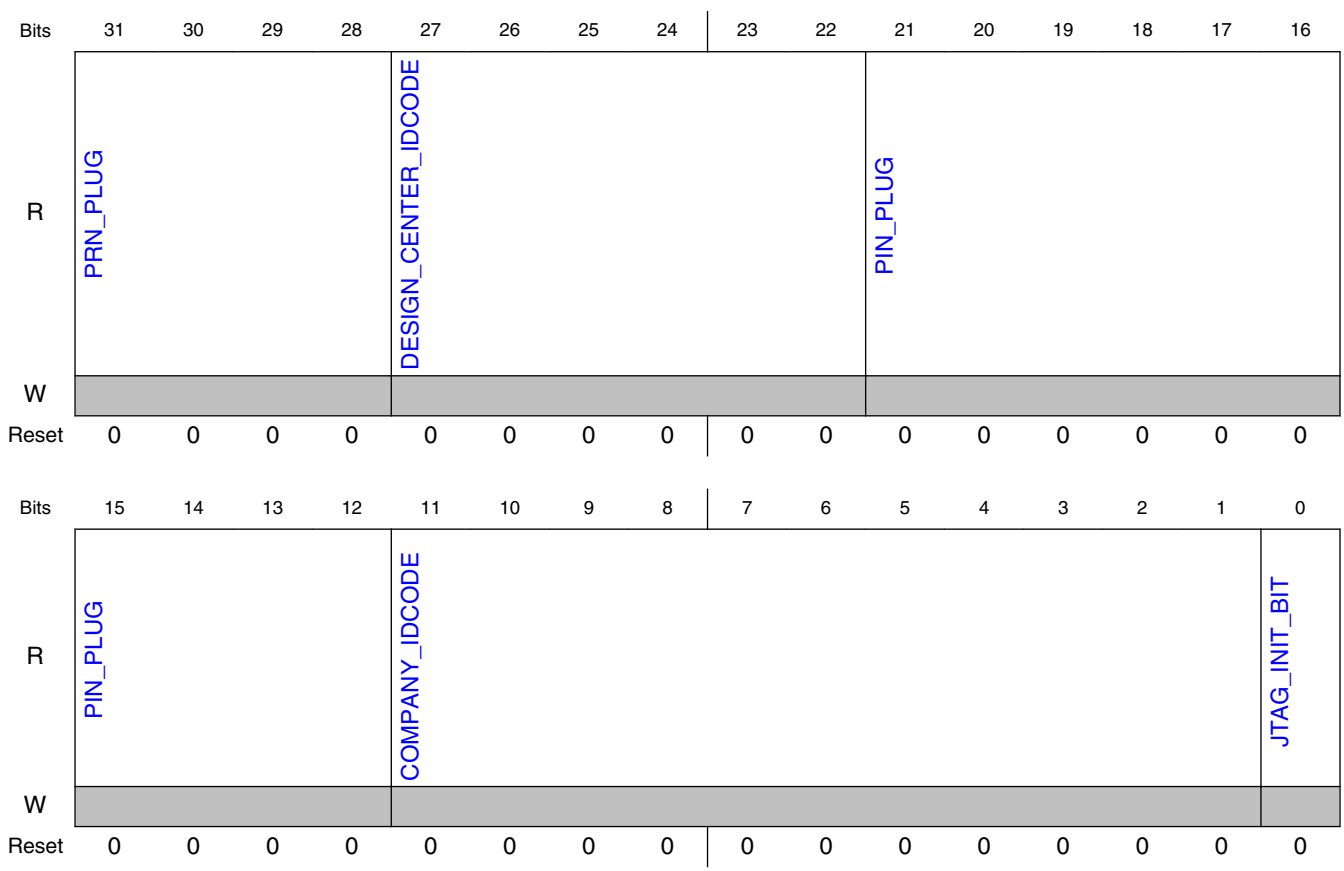
31.5.1.25.1 Offset

Register	Offset
JTAG_ID_REG	8Ch

31.5.1.25.2 Function

Mirrors JTAG ID Register.

31.5.1.25.3 Diagram



31.5.1.25.4 Fields

Field	Function
31-28 PRN_PLUG	Part Revision Number.
27-22 DESIGN_CENTER_IDCODE	Design Center ID Code.
21-12 PIN_PLUG	Part Identification Number.
11-1 COMPANY_IDCODE	Company ID Code.
0 JTAG_INIT_BIT	JTAG ID initial bit.

31.5.1.26 Lower A7 TS Timer compare value (A7_TSTMR_CMP_VAL_L)

31.5.1.26.1 Offset

Register	Offset
A7_TSTMR_CMP_VAL_L	90h

31.5.1.26.2 Function

Lower bits of compare value for A7 TS Timer.

31.5.1.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	A7_TSTMR_CMP_VAL_L															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	A7_TSTMR_CMP_VAL_L															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.26.4 Fields

Field	Function
31-0 A7_TSTMR_CMP_VAL_L	Lower A7 TS Timer compare value. Stores the lower 32 bits to compare with A7 TS Timer. The compare is enabled when MISC_CTRL0[A7_TSTMR_CMP_EN] is set. If enabled the compare match will issue a IRQ/wake-up signal to A7 domain.

31.5.1.27 Upper A7 TS Timer compare value (A7_TSTMR_CMP_VAL_H)

31.5.1.27.1 Offset

Register	Offset
A7_TSTMR_CMP_VAL_H	94h

31.5.1.27.2 Function

Upper bits of compare value for A7 TS Timer.

31.5.1.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	A7_TSTMR_CMP_VAL_H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	A7_TSTMR_CMP_VAL_H															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

31.5.1.27.4 Fields

Field	Function
31-0 A7_TSTMR_CMP_VAL_H	Upper A7 TS Timer compare value. Stores the upper 32 bits to compare with A7 TS Timer. The compare is enabled when MISC_CTRL0[A7_TSTMR_CMP_EN] is set. If enabled the compare match will issue a IRQ/wake-up signal to A7 domain.

31.5.1.28 Override Control for Compensation Codes (COMP_CELL_OVERRIDE)

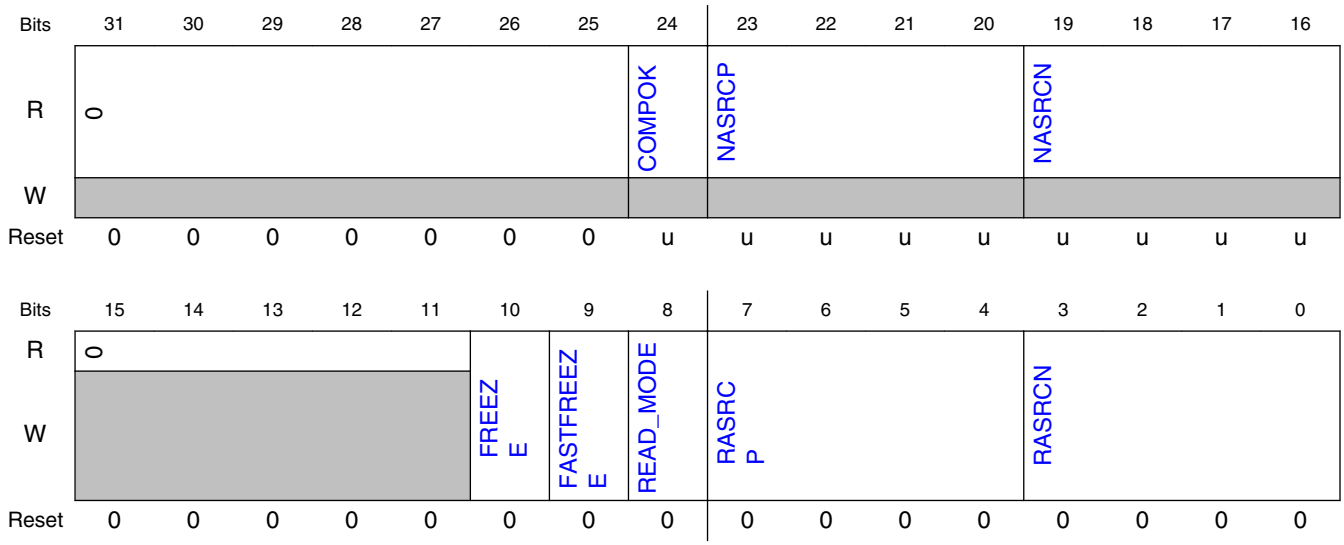
31.5.1.28.1 Offset

Register	Offset
COMP_CELL_OVERRIDE	98h

31.5.1.28.2 Function

Implements the override of Compensation cells.

31.5.1.28.3 Diagram



31.5.1.28.4 Fields

Field	Function
31-25 —	Reserved.
24 COMPOK	Compensation OK. 1'b0 - compensation cell is not in normal mode or is entering normal mode (Delay to measure a new code). 1'b1 - compensation cell is in the Normal mode and a new measured code is available on the ASRC lines. **NOTE:** This field value varies with parameters such as temperature and voltage supply. It is thus undefined.
23-20 NASRCP	4-bit PMOS compensation measured/generated codes. **NOTE:** This field value varies with parameters such as temperature and voltage supply. It is thus undefined.
19-16 NASRCN	4-bit NMOS compensation measured/generated codes. **NOTE:** This field value varies with parameters such as temperature and voltage supply. It is thus undefined.
15-11 —	Reserved.
10 FREEZE	Freeze. When set to "1", it freezes the compensation code at its running value. Freeze mode is activated on bit setting to "1". The compensation cell delivers refreshed compensation code at a delay after bit is cleared.
9 FASTFREEZE	Fast-Freeze. When set to "1", it fast-freezes the compensation code at its running value.

Table continues on the next page...

Field	Function
	Fast Freeze mode is activated on bit setting to "1". The compensation cell delivers a refreshed compensation code, at a comparatively lesser delay after signal falling edge than the delay of FREEZE bit.
8 READ_MODE	Read Mode. When set to "1", it forces Compensation Cell to "READ MODE". It has priority over FREEZE and FASTFREEZE configuration. In read mode, compensation codes are received from RASRCN and RASRCP.
7-4 RASRCP	4-bit PMOS compensation codes.
3-0 RASRCN	4-bit NMOS compensation codes.

Chapter 32
Low-Leakage Wake-Up Unit (LLWU)

32.1 Chip-specific LLWU information

Table 32-1. Reference links to related information

Topic	Related module	Reference
Full description	LLWU	LLWU
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

32.1.1 Low-Leakage Wake-Up Unit

The Low-Leakage Wake-Up Unit (LLWU) module allows user to select up to 32 external pin sources and up to 8 internal modules as a wake up source from low-leakage power modes. Each of the available wake-up sources can be individually enabled. I.MX 7ULP integrates 1 instance of LLWU in the "always-on" DGO power domain. LLWU external pins can be used as wake-up sources for external devices associated with M4 or A7 domain. See Table 32-3 for details.

Table 32-2. LLWU configuration

Parameter	Description
Name	LLWU
Instances	1
Configurable features	• Port A and B pins can be configured as LLWU GPIO interrupts. • Other LLWU sources include RTC alarm, RTC seconds, CMP0-1 and LPTMR0-1 interrupts
Interface speed	NA
External I/O pins	See the attached IOMUXC and pinout spreadsheet

Table 32-3. LLWU wake-up sources

LLWU Pin/Module	Wakeup Sources
LLWU_P0	PTA0
LLWU_P1	PTA3
LLWU_P2	PTA13
LLWU_P3	PTA14
LLWU_P4	PTA18
LLWU_P5	PTA19
LLWU_P6	PTA23
LLWU_P7	PTA31
LLWU_P8	PTB1
LLWU_P9	PTB3
LLWU_P10	PTB6
LLWU_P11	PTB7
LLWU_P12	PTB9
LLWU_P13	PTB14
LLWU_P14	PTB16
LLWU_P15	PTB19
LLWU_P16	Reserved
LLWU_P17	Reserved
LLWU_P18	Reserved
LLWU_P19	Reserved
LLWU_P20	Reserved
LLWU_P21	Reserved
LLWU_P22	Reserved
LLWU_P23	Reserved
LLWU_P24	Reserved
LLWU_P25	Reserved
LLWU_P26	Reserved
LLWU_P27	Reserved
LLWU_P28	Reserved
LLWU_P29	Reserved
LLWU_P30	Reserved
LLWU_P31	Reserved
LLWU_M0IF	LPTMR0
LLWU_M1IF	LPTMR1
LLWU_M2IF	CMP0
LLWU_M3IF	CMP1
LLWU_M4IF	SNVS[1]
LLWU_M5IF	Reserved
LLWU_M6IF	USB PHY VLLS wakeup interrupt
LLWU_M7IF	Reserved

1. SNVS wake-up alarm

32.1.2 DMA support

The Low-Leakage Wake-Up Unit (LLWU) module on i.MX 7ULP doesn't support DMA/TRIGGER request as wake-up sources. Any text mentioning DMA trigger request as wake-up source throughout the chapter, should be ignored. Ideally, the reset value of DMAS field should be 0, however, the PARAM register is implemented with a reset value of 2020_2004 (as shown in memory map), showing that it could support 32 channels.

32.1.3 Operational requirements

LLWU module on i.MX 7ULP requires an M4 system IP bus clock/M4 slow clock maximum ratio of 3:1 (bus clock 3x faster than slow clock) prior to entering LLS/VLLS mode. This constraint applies only if any LLWU option is enabled as LLS/VLLS wakeup source.

32.2 Introduction

The LLWU module allows the user to select external pins and internal modules as interrupt wake-up sources from LLS/VLLS power modes. It also allows pins and/or modules to generate a temporary wake-up from LLS to allow servicing of DMA or trigger events. Digital filtering is also available for the external wakeup pins.

The RESET_B pin is an additional source for triggering an exit from low-leakage power modes, and causes the MCU to exit both LLS and VLLS through a reset flow. Input sources to the LLWU are described in the device's chip configuration details.

See AN4503: Power Management for Kinetis MCUs for further details on using the LLWU.

32.2.1 Features

The LLWU module features include:

- Support for up to 8'd32 external input pins and up to 8 internal modules with individual enable bits for MCU interrupt from low leakage modes.

- Input sources may be external pins or from internal modules capable of running in LLS or VLLS. See the chip configuration information for wakeup input sources for this device.
- Support to configure internal modules as DMA or trigger sources for temporary wakeup from LLS.
- External input pins programmable for falling-edge, rising-edge, or any-edge detection.
- Optional digital filters provided to qualify an external pin detect. Note that when the LPO clock is disabled, filters are disabled and bypassed.

32.2.2 Modes of operation

The LLWU module becomes functional on entry into a low-leakage power mode. After recovery from LLS, the LLWU is immediately disabled. After recovery from VLLS, the LLWU continues to detect wake-up events until the user has reinitialized the system and deasserted isolation.

32.2.2.1 LLS mode

Wake-up events due to external pin inputs (LLWU_Px) and internal modules (LLWU_MxIF) result in an interrupt flow when exiting LLS. Wake-up events due to internal module requests (LLWU_MxDF) and external pin inputs cause the system to temporarily exit LLS with the CPU clock disabled, allowing the DMA/trigger request to be serviced. When the DMA/trigger event is complete, the system will re-enter LLS.

NOTE

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully recover from LLS after an external pin event.

32.2.2.2 VLLS modes

All wakeup and reset events result in VLLS exit via a reset flow.

32.2.2.3 Non-low leakage modes

The LLWU is inactive in all non-low leakage modes. The LLWU registers are accessible in non-low leakage modes and are available for configuring and reading status when bus transactions are possible.

When an external wake-up pin filter is first enabled in the LLWU_FILT register, filter operation begins immediately. To allow the filter to initialize, the user should wait at least five LPO clock cycles before entering a low leakage mode.

32.2.3 Block diagram

The following figure is the block diagram for the LLWU module.

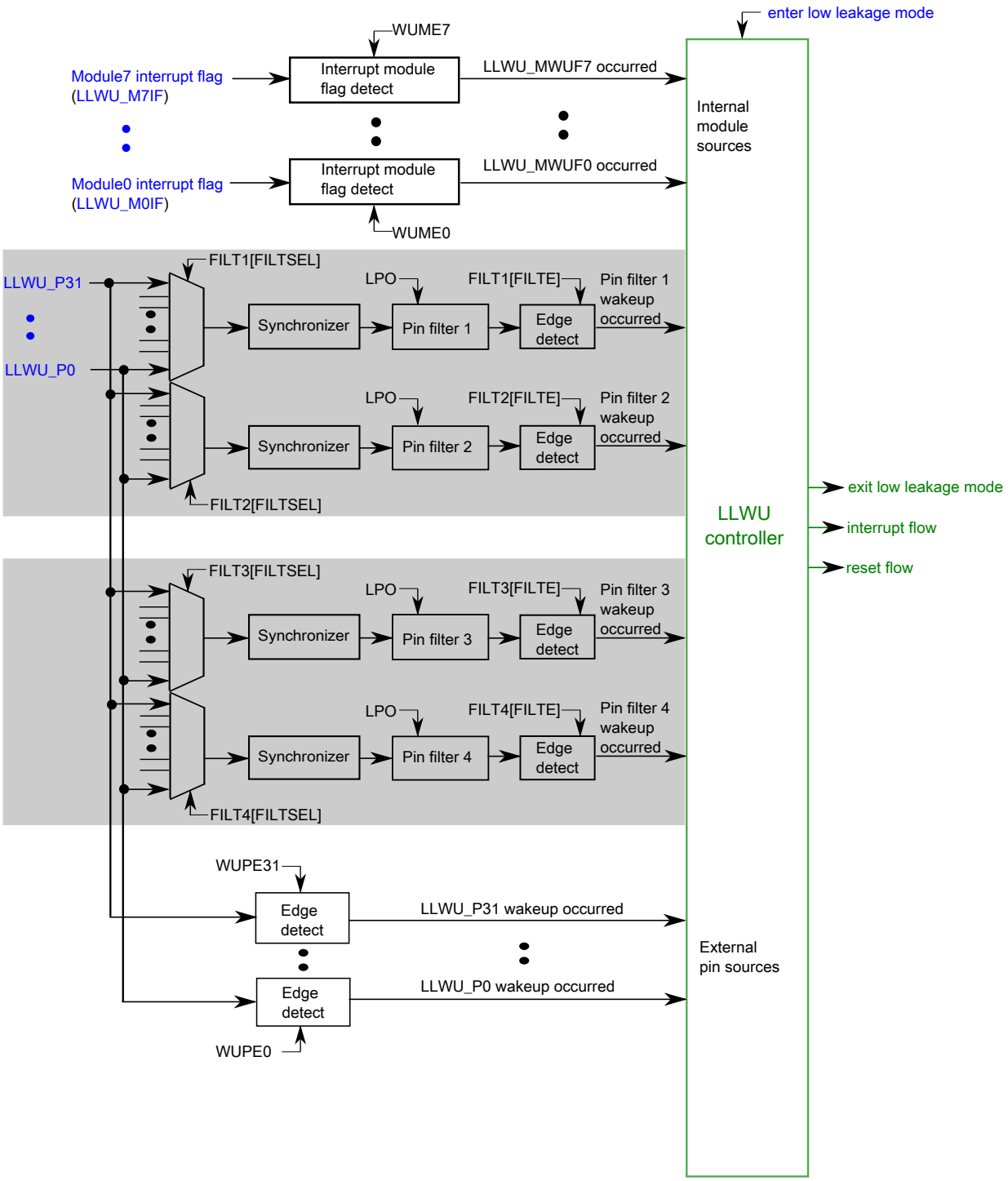


Figure 32-1. LLWU block diagram

32.3 LLWU signal descriptions

The signal properties of LLWU are shown in the table found here.

The external wakeup input pins can be enabled to detect either rising-edge, falling-edge, or on any change.

Table 32-4. LLWU signal descriptions

Signal	Description	I/O
LLWU_Pn	External Pin Wakeup inputs (n = 0-31)	I
LLWU_MnIF	Internal Module Interrupt flag (n = 0-7)	I
LLWU_MnDR	Internal Module DMA/trigger request (n = 0-7)	I

32.4 Memory map/register definition

Information about LLWU registers can be found here.

32.4.1 LLWU register descriptions

NOTE

The LLWU registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

Registers for Pin/Module sources marked "Reserved" in the LLWU chip configuration information will not be writable and will return 0 when read.

All LLWU registers are reset by Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. LLWU registers are unaffected by reset types that do not trigger Chip Reset not VLLS.

32.4.1.1 LLWU Memory map

LLWU base address: 4102_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
4h	Parameter Register (PARAM)	32	RO	2020_2004h
8h	Pin Enable 1 register (PE1)	32	RW	0000_0000h
Ch	Pin Enable 2 register (PE2)	32	ROZ	0000_0000h
18h	Module Interrupt Enable register (ME)	32	RW	0000_0000h
1Ch	Module DMA/Trigger Enable register (DE)	32	ROZ	0000_0000h
20h	Pin Flag register (PF)	32	W1C	0000_0000h
28h	Module Interrupt Flag register (MF)	32	RO	0000_0000h
30h	Pin Filter register (FILT)	32	RW	0000_0000h

32.4.1.2 Version ID Register (VERID)

32.4.1.2.1 Offset

Register	Offset
VERID	0h

32.4.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.4.1.2.3 Fields

Field	Function
31-24	Major Version Number
MAJOR	This read only field returns the major version number for the module specification.
23-16	Minor Version Number

Table continues on the next page...

Field	Function
MINOR	This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000000000000b - Standard features implemented

32.4.1.3 Parameter Register (PARAM)

32.4.1.3.1 Offset

Register	Offset
PARAM	4h

32.4.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PINS								MODULES							
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMAS								FILTERS							
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0

32.4.1.3.3 Fields

Field	Function
31-24 PINS	Pin Number Number of Pin wakeup sources supported
23-16 MODULES	Module Number Number of Module wakeup sources supported
15-8 DMAS	DMA Number Number of DMA wakeup sources supported
7-0 FILTERS	Filter Number Number of Pin Filters implemented

32.4.1.4 Pin Enable 1 register (PE1)

32.4.1.4.1 Offset

Register	Offset
PE1	8h

32.4.1.4.2 Function

The PE1 register contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P15-LLWU_P0.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS.

NOTE

If an external wakeup pin is not supported on a given device, the corresponding WUPE*n* field always reads as 00.

32.4.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WUPE15		WUPE14		WUPE13		WUPE12		WUPE11		WUPE10		WUPE9		WUPE8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WUPE7		WUPE6		WUPE5		WUPE4		WUPE3		WUPE2		WUPE1		WUPE0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.4.1.4.4 Fields

Field	Function
31-30: WUPE15	Wakeup pin enable for LLWU_Pn
29-28: WUPE14	Enables and configures the edge detection for the wakeup pin. 00b - External input pin disabled as wakeup input

Field	Function
27-26: WUPE13 25-24: WUPE12 23-22: WUPE11 21-20: WUPE10 19-18: WUPE9 17-16: WUPE8 15-14: WUPE7 13-12: WUPE6 11-10: WUPE5 9-8: WUPE4 7-6: WUPE3 5-4: WUPE2 3-2: WUPE1 1-0: WUPE0	01b - External input pin enabled with rising edge detection 10b - External input pin enabled with falling edge detection 11b - External input pin enabled with any change detection

32.4.1.5 Pin Enable 2 register (PE2)

32.4.1.5.1 Offset

Register	Offset
PE2	Ch

32.4.1.5.2 Function

The PE2 register contains the field to enable and select the edge detect type for the external wakeup input pins LLWU_P31-LLWU_P16.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS.

NOTE

If an external wakeup pin is not supported on a given device, the corresponding WUPE*n* field always reads as 00.

32.4.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		0		0		0		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		0		0		0		0		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.4.1.5.4 Fields

Field	Function
31-30: Reserved31	Wakeup pin enable for LLWU_Pn Enables and configures the edge detection for the wakeup pin. 00b - External input pin disabled as wakeup input 01b - External input pin enabled with rising edge detection 10b - External input pin enabled with falling edge detection 11b - External input pin enabled with any change detection
29-28: Reserved30	
27-26: Reserved29	
25-24: Reserved28	
23-22: Reserved27	
21-20: Reserved26	
19-18: Reserved25	
17-16: Reserved24	
15-14: Reserved23	
13-12: Reserved22	
11-10: Reserved21	
9-8: Reserved20	
7-6: Reserved19	
5-4: Reserved18	
3-2: Reserved17	
1-0: Reserved16	

32.4.1.6 Module Interrupt Enable register (ME)

32.4.1.6.1 Offset

Register	Offset
ME	18h

32.4.1.6.2 Function

The ME register contains the bits to enable an internal module interrupt as a wakeup source for inputs LLWU_M7IF - LLWU_M0IF.

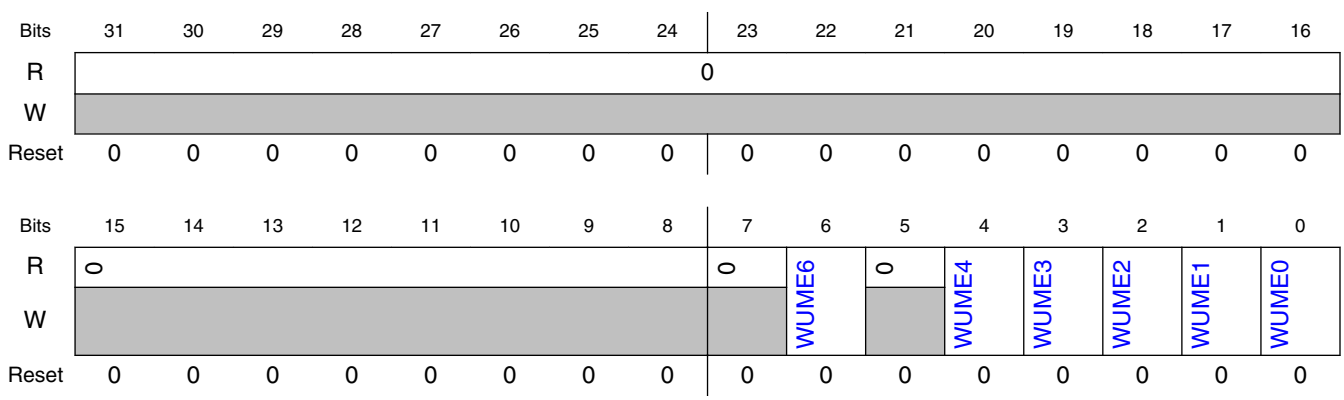
NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS.

NOTE

If a module interrupt is not supported on a given device, the corresponding WUME*n* bit always reads as a 0.

32.4.1.6.3 Diagram



32.4.1.6.4 Fields

Field	Function
31-8	Reserved.

Table continues on the next page...

Field	Function
—	
7-0 WUMEn	Wakeup module enable for module n Enables an internal module as a wakeup source input. 0b - Internal module flag not used as wakeup source 1b - Internal module flag used as wakeup source

32.4.1.7 Module DMA/Trigger Enable register (DE)

32.4.1.7.1 Offset

Register	Offset
DE	1Ch

32.4.1.7.2 Function

The DE register contains the bits to enable an internal module DMA/Trigger request as a wakeup source for inputs LLWU_M7DR - LLWU_M0DR.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS.

NOTE

If a module DMA/Trigger request is not supported on a given device, the corresponding WUDE*n* bit always reads as a 0.

32.4.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.4.1.7.4 Fields

Field	Function
31-8 —	Reserved.
7-0 WUDEn	DMA/Trigger wakeup enable for module n Enables an internal module as a DMA/Trigger wakeup source. 0b - Internal module request not enabled as a DMA/Trigger wakeup source 1b - Internal module request enabled as a DMA/Trigger wakeup source

32.4.1.8 Pin Flag register (PF)

32.4.1.8.1 Offset

Register	Offset
PF	20h

32.4.1.8.2 Function

The PF register contains the wakeup flags indicating which wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this is the source causing the CPU interrupt flow. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUF*n* bit. The wakeup flag (WUF*n*), if set, will remain set if the associated WUPE*n* bit is cleared.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS.

NOTE

If an external wakeup pin is not supported on a given device, the corresponding WUF*n* bit always reads as a 0.

32.4.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WUF15	WUF14	WUF13	WUF12	WUF11	WUF10	WUF9	WUF8	WUF7	WUF6	WUF5	WUF4	WUF3	WUF2	WUF1	WUF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.4.1.8.4 Fields

Field	Function
31-0 WUFn	Wakeup flag for LLWU_Pn Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF*n*. 0b - LLWU_Pn input was not a wakeup source 1b - LLWU_Pn input was a wakeup source

32.4.1.9 Module Interrupt Flag register (MF)

32.4.1.9.1 Offset

Register	Offset
MF	28h

32.4.1.9.2 Function

The MF register contains the wakeup flags indicating which internal module interrupt caused the MCU to exit LLS or VLLS mode. For LLS, this is the source causing the CPU interrupt flow. For VLLS, this is the source causing the MCU reset flow.

For internal peripherals that are capable of running in a low-leakage power mode, such as a real time clock module or CMP module, the flag from the associated peripheral is accessible as the MWUF*n* bit. The flag will need to be cleared in the peripheral instead of writing a 1 to the MWUF*n* bit.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS.

NOTE

If a module interrupt is not supported on a given device, the corresponding MWUF*n* bit always reads as a 0.

32.4.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	MWUF6	0	MWUF4	MWUF3	MWUF2	MWUF1	MWUF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.4.1.9.4 Fields

Field	Function
31-8 —	Reserved.
7-0 MWUFn	Wakeup flag for module n Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism. 0b - Module input was not a wakeup source 1b - Module input was a wakeup source

32.4.1.10 Pin Filter register (FILT)

32.4.1.10.1 Offset

Register	Offset
FILT	30h

32.4.1.10.2 Function

The FILT register is a control and status register that is used to enable/disable the digital filter features for an external pin.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS.

32.4.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FILTF4	FILTE4		FILTSEL 4					FILTF3	FILTE3		FILTSEL 3				
W	W1C								W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FILTF2	FILTE2		FILTSEL 2					FILTF1	FILTE1		FILTSEL 1				
W	W1C								W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

32.4.1.10.4 Fields

Field	Function
31 FILTF4	Filter 4 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0b - Pin Filter 1 was not a wakeup source

Table continues on the next page...

Field	Function
	1b - Pin Filter 1 was a wakeup source
30-29 FILTE4	Filter 4 Enable Controls the digital filter 4 options for the external pin detect. 00b - Filter disabled 01b - Filter posedge detect enabled 10b - Filter negedge detect enabled 11b - Filter any edge detect enabled
28-24 FILTSEL4	Filter 4 Pin Select Selects 1 of the wakeup pins to be muxed into filter 4. 00000b - Select LLWU_P0 for filter 11111b - Select LLWU_P31 for filter
23 FILTF3	Filter 3 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0b - Pin Filter 1 was not a wakeup source 1b - Pin Filter 1 was a wakeup source
22-21 FILTE3	Filter 3 Enable Controls the digital filter 3 options for the external pin detect. 00b - Filter disabled 01b - Filter posedge detect enabled 10b - Filter negedge detect enabled 11b - Filter any edge detect enabled
20-16 FILTSEL3	Filter 3 Pin Select Selects 1 of the wakeup pins to be muxed into filter 3. 00000b - Select LLWU_P0 for filter 11111b - Select LLWU_P31 for filter
15 FILTF2	Filter 2 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0b - Pin Filter 2 was not a wakeup source 1b - Pin Filter 2 was a wakeup source
14-13 FILTE2	Filter 2 Enable Controls the digital filter 2 options for the external pin detect. 00b - Filter disabled 01b - Filter posedge detect enabled 10b - Filter negedge detect enabled 11b - Filter any edge detect enabled
12-8 FILTSEL2	Filter 2 Pin Select Selects 1 of the wakeup pins to be muxed into filter 2. 00000b - Select LLWU_P0 for filter 11111b - Select LLWU_P31 for filter
7 FILTF1	Filter 1 Flag Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. 0b - Pin Filter 1 was not a wakeup source 1b - Pin Filter 1 was a wakeup source
6-5 FILTE1	Filter 1 Enable Controls the digital filter 1 options for the external pin detect.

Table continues on the next page...

Field	Function
	00b - Filter disabled 01b - Filter posedge detect enabled 10b - Filter negedge detect enabled 11b - Filter any edge detect enabled
4-0 FILTSEL1	Filter 1 Pin Select Selects 1 of the wakeup pins to be muxed into filter 1. 00000b - Select LLWU_P0 for filter 11111b - Select LLWU_P31 for filter

32.5 Functional description

The low-leakage wakeup unit (LLWU) module allows internal peripherals and external input pins as a source of wakeup from low-leakage modes.

The LLWU module contains pin enables for each external pin and internal module. For each external pin, the user can disable or select the edge type for the wakeup with the following options:

- Falling-edge
- Rising-edge
- Either-edge

When an external pin is enabled as a wakeup source, the pin must be configured as an input pin.

The LLWU implements optional 3-cycle glitch filters, based on the LPO clock. A detected external pin is required to remain asserted until the enabled glitch filter times out. Additional latency of up to 2 cycles is due to synchronization, which results in a total of up to 5 cycles of delay before the detect circuit alerts the system to the wakeup or reset event when the filter function is enabled. Four wakeup detect filters are available for selected external pins. Glitch filtering is not provided on the internal modules.

For internal module interrupts, the WUMEx bit enables the associated module interrupt as a wakeup source.

32.5.1 LLS mode

Wakeup events triggered from either an external pin input or an internal module interrupt, result in a CPU interrupt flow to begin user code execution.

Wakeup events triggered from an internal module DMA/Trigger request, result in a temporary wake-up from LLS with CPU remaining clock gated to allow the DMA/ Trigger request to be serviced. After the DMA/Trigger request is cleared, the system will re-enter LLS mode.

An LLS reset event due to RESET_B pin assertion causes an exit via a system reset. State retention data is lost, and the I/O states return to their reset state.

32.5.2 VLLS modes

For any LLWU wakeup from VLLS, recovery is always via a reset flow. A VLLS exit event due to RESET_B pin assertion causes an exit via a system reset with I/O returning to their reset state immediately.

32.5.3 Initialization

For an enabled peripheral wakeup input, the peripheral flag must be cleared by software before entering LLS or VLLS mode to avoid an immediate exit from the mode.

Flags associated with external input pins, filtered and unfiltered, must also be cleared by software prior to entry to LLS or VLLS mode.

After enabling an external pin filter or changing the source pin, wait at least five LPO clock cycles to allow the filter to initialize before entering LLS or VLLS mode .

NOTE

After recovering from a VLLS mode, user must restore chip configuration before clearing the PMC ACKISO bit. In particular, pin configuration for enabled LLWU wake-up pins must be restored to avoid any LLWU flag from being falsely set when ACKISO is cleared.

Chapter 33
Wake-Up Unit (WKPU)

33.1 Chip-specific WKPU information

Table 33-1. Reference links to related information

Topic	Related module	Reference
Full description	WKPU	WKPU
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

33.1.1 Wakeup Unit

The Wakeup Unit manages wakeup events and wakes up the device upon a valid wake-up event. The WKPU supports 21 resources. See Table 33-3 for list of wake-up channel assignments. The WKPU combines its wakeup events to supply a single wake-up to the system.

NOTE

Masking of the WKPU interrupt sources is enabled via SIM_WKPU_WAKEUP_EN register.

Table 33-2. WKPU configuration

Parameter	Description
Name	Wakeup Unit
Instances	1
Configurable features	30 wake-up sources
Interface speed	NA
External I/O pins	NA

Table 33-3. Wake-up channel assignments

WKPU channel	Interrupt sources	GIC IRQ#
0	DMA1 channel 0-31 transfer complete DMA1 error interrupt - all channels	0-16
1	A7 CTI	17
2	MU_B	19
3	TPM4,5,6,7 and LPIT1	21-24, and 25
4	FlexIO1, LPSPI2, 3	26, 28-29
5	LPUART4, 5, 6, 7	30 - 33
6	LP2IC4, 5, 6, 7	34 - 37
7	USBPHY, USBOTG1, USBOTG2	39, 40, 41
8	uSDHC0, uSDHC1	42, 43
9	GPU-3D/GPU-2D	44, 52
10	VIU, LCDIF, and MIPI-DSI	45, 46, 47
11	M4 LPIT0	77
12	M4 LPUARTs (0-3)	80, 81, 82, 83
13	M4 LPI2Cs (0-3)	73, 74, 75, 76
14	PCTLC, D, E, F	48 - 51
15	WDOG1	55
16	WDOG2	56
17	PMC1	57
18	SCG1	58
19	CMC1	59
20	M4 QSPI	64
21	M4 SAI0, 1	65 - 66
22	M4 PCTLA, B	67 - 68
23	TSTMR	91
24	M4 XRDC	70
25	Reserved	-
26	CAAM[1]	54
27	MU_B NMI	90
28	M4 LPSPI0,1	71, 72
29	M4 SNVS	88
30	M4 X-Domain-DMA-Interrupt	78
31	M4 X-Domain-Master-Interrupt	79

1. See i.MX7ULPRM Security Reference Manual for details on this chapter

33.1.2 Supported wake-up sources

The Wake-Up Unit in i.MX 7ULP supports upto 32 wake-up sources, out of which 8 are reserved. External NMI is not supported on this device.

Chapter 34
On-Chip One-Time-Programmable Controller (OCOTP_CTRL)

34.1 Chip-specific OCTOP_CTRL information

Table 34-1. Reference links to related information

Topic	Related module	Reference
Full description	OCOTP_CTRL	OCOTP_CTRL
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

34.1.1 On-Chip One Time Programmable Controller (OCOTP_CTRL)

The OCOTP_CTRL or fuse box control (FBC) module[1] provides an interface for reading, programming and/or overriding identification and control information stored in on-chip fuse elements. The module supports electrically-programmable poly fuses (e-Fuses). Among the uses for the fuses are flash security and phantom configuration of the crypto modules. For security purposes, the fuses protect the confidentiality or integrity of critical security data against both software attacks and board-level hardware attacks. The following table shows the configuration of the FBC.

Table 34-2. OCOTP controller configuration

Parameter	Description
Name	OCOTP_CTRL
Instances	1

Table continues on the next page...

1. For complete chapter with all security registers, see i.MX 7ULP Security Reference Manual

Table 34-2. OCOTP controller configuration (continued)

Parameter	Description
Configurable features	8-kbit eFUSE OTP
Interface speed	NA
External I/O pins	NA
Interrupts	NA

34.2 Overview

This section contains information describing the requirements for the on-chip eFuse OTP controller along with details about the block functionality and implementation.

In this document, the words "eFuse" and "OTP" are interchangeable. OCOTP refers to the hardware block itself.

34.2.1 Features

The OCOTP provides the following features :

- 32-bit word restricted program and read to 8 kbit of eFuse OTP.
- Loading and housing of fuse content into shadow registers
- Memory-mapped (restricted) access to shadow registers
- Generation of STICKY_REG which consists of sticky register bits
- Provides program-protect and read-protect eFuse
- Provides override and read protection of shadow register
- Supports ECC mode programming and reading for MTR fuse words by SkyBlue IPS bus.
- Supports ECC mode programming and reading for all the user fuse words .
- Supports redundancy mode programming and reading for all the supplementary fuse words

34.3 Clocks

The table found here describes the clock sources for OCOTP. Please see the chip-specific clocking section for clock setting, configuration and gating information.

Table 34-3. OCOTP Clocks

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

34.4 Top-Level Symbol and Functional Overview

The figure below shows the OCOTP system level diagram.

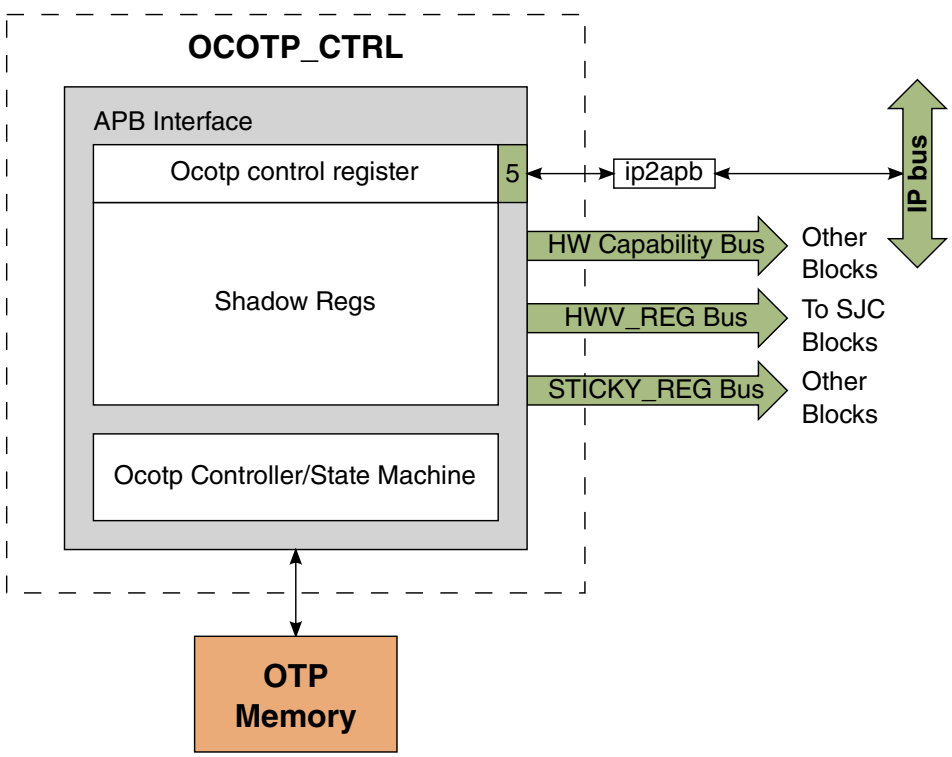


Figure 34-1. OCOTP System Level Diagram

34.4.1 Operation

The IP bus interface of the OCOTP has the following functions.

- Configure control registers for programming and reading all the fuse words
- Override and read shadow registers
- Programming and reading MTR fuse words by SkyBlue IPS bus.

OCOTP configuration for programs and reads are performed on 32-bit words for software (SW) convenience. For writes, the 32-bit word reflects the "write-mask." Bit fields with 0 will not be programmed and bit fields with 1 will be programmed.

NOTE

If ECC is supported, the fuse word can only be programmed once, otherwise it will report SEC/DED issues in read or reload.

NOTE

If redundancy is supported, the fuse word can be programmed more than once (not the same fuse).

In this document, 8 kbit fuse are divided into 32 banks by function. Each bank has 8 fuse words. Physically, 8 kbits fuse are in one 8k OTP memory.

34.4.1.1 Shadow Register Reload

All fuse words in efusebox are shadowed. Therefore, fuse information is available through memory mapped shadow registers. If fuses are subsequently programmed, the shadow registers should be reloaded to keep them coherent with the fuse bank arrays.

The "reload shadows" feature allows the user to force a reload of the shadow registers without having to reset the device. To force a reload, complete the following steps:

1. Check that HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write, read or reload must be completed before a new access can be requested.
2. Set the HW_OCOTP_CTRL[RELOAD_SHADOWS] bit. OCOTP will read all the fuses one by one and put it into corresponding shadow register.
3. Wait for HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[RELOAD_SHADOWS] to be cleared by the controller.
4. After reload, it needs to check if HW_OCOTP_OUT_STATUS[SEC_RELOAD] and HW_OCOTP_OUT_STATUS[DED_RELOAD] are asserted.
   - If HW_OCOTP_OUT_STATUS[SEC_RELOAD] is asserted, it indicates single bit correction ever happened when reloading.
   - If HW_OCOTP_OUT_STATUS[DED_RELOAD] is asserted, it indicates double bits detection ever happened when reloading. That is to say, one or more fuse word are not correct when reading from eFuse OTP.

The controller will automatically clear the HW_OCOTP_CTRL[RELOAD_SHADOWS] bit after the successful completion of the operation.

34.4.1.2 Fuse and Shadow Register Read

All shadow registers are always readable through the IPS bus except some secret keys regions. When their corresponding fuse lock bits are set, the shadow registers also become read locked. After read locking, reading from these registers will return 0xBADABADA.

In addition HW_OCOTP_CTRL[ERROR] will be set. It must be cleared by software before any new write, read or reload access can be issued. Subsequent reads to unlocked shadow locations will still work successfully however.

To read fuse words directly from the fusebox, complete the following steps:

1. Check that HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write, read or reload must be completed before a read access can be requested.
2. Write the requested fuse address to HW_OCOTP_CTRL[ADDR].
3. When reading supplementary fuse, HW_OCOTP_CTRL[SUPPAD] must be set. When reading fuse word, HW_OCOTP_CTRL[SUPPAD] must be 0.
4. Set HW_OCOTP_READ_CTRL[READ_FUSE] to 1. OCOTP will auto read the fuse word according to HW_OCOTP_CTRL[ADDR] in fusebox. Then put read value into HW_OCOTP_READ_FUSE_DATA
5. Once complete, the controller will clear BUSY. A read request to a protected or locked region will result in no OTP access and no setting of HW_OCOTP_CTRL[BUSY]. In addition HW_OCOTP_CTRL[ERROR] will be set. It must be cleared by software before any new access can be issued.
6. Read HW_OCOTP_READ_FUSE_DATA to get fuse word value. HW_OCOTP_READ_FUSE_DATA will be 0xBADABADA when HW_OCOTP_CTRL[ERROR] is set.
7. After reading is completed, it needs to check HW_OCOTP_OUT_STATUS[DED] value to confirm if read fuse value is correct. If it is 1, read fuse value is wrong, otherwise it is correct.

34.4.1.3 Fuse and Shadow Register Writes

Shadow register bits can be overridden by software until the corresponding fuse lock bit for the region is set. When the lock shadow bit is set, the shadow registers for that lock region become write locked. The HW_OCOTP_LOCKn register also has no overwrite or fuse lock bits, but it is always read-only.

In order to avoid "rogue" code performing erroneous writes to OTP, a special unlocking sequence is required for writes to the fuse banks. To program fuse bank complete the following steps:

1. Check that HW_OCOTP_CTRL[BUSY] and HW_OCOTP_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write or reload must be completed before a write access can be requested.
2. Write the requested address to HW_OCOTP_CTRL[ADDR] and program the unlock code into HW_OCOTP_CTRL[WR_UNLOCK]. The unlock code must be programmed for each write access. The unlock code is documented in the register description. Both the unlock code and address can be written in the same operation. When programming supplementary fuse, HW_OCOTP_CTRL[SUPPAD] must be set. When programming fuse word, HW_OCOTP_CTRL[SUPPAD] must be 0. HW_OCOTP_CTRL[SUPPAD] must be set with unlock code and address in same operation.
3. When programming fuse word, OCOTP_CTRL will auto select ECC or redundancy mode to program the fuse word according to fuse map definition. In ECC mode, the 32 fuse bits in one word can only be written once. In redundancy mode, the word can be written more than once as long as they are different fuse bits.For the fuse word which needs be programmed with ECC mode, it is recommended to set HW_OCOTP_CTRL[WORDLOCK] with unlock code and address in same operation.
4. Write the data to the HW_OCOTP_DATA register. This will automatically set HW_OCOTP_CTRL[BUSY] and clear HW_OCOTP_CTRL[WR_UNLOCK]. Bit fields with 1's will result in that OTP bit being programmed. Bit fields with 0's will be ignored. At the same time that the write is accepted, the controller makes an internal copy of HW_OCOTP_CTRL[ADDR] which cannot be updated until the next write sequence is initiated. This copy guarantees that erroneous writes to HW_OCOTP_CTRL[ADDR] will not affect an active write operation. During the write operation, HW_OCOTP_DATA cannot be modified.
5. Once complete, the controller will clear HW_OCOTP_CTRL[BUSY]. A write request to a protected or locked region will result in no OTP access and no setting of HW_OCOTP_CTRL[BUSY]. In addition HW_OCOTP_CTRL[ERROR] will be set. It must be cleared by software before any new write access can be issued.
6. After programming is completed, it needs to check if HW_OCOTP_OUT_STATUS[PROGFAIL] or HW_OCOTP_OUT_STATUS[LOCKED] is asserted. If either of these two bits is asserted, it indicates that programming has failed.

It should be noted that write latencies to OTP are numbers of 10 micro-seconds per word. The write latency is based on the number of "1" bits being written. For example : to program half the fuse bits in one 32-bt word needs 10 us x 16.

HW_OCOTP_CTRL[ERROR] will be set under the following conditions:

- A write is performed to a shadow register during a shadow reload (essentially, while HW_OCOTP_CTRL[RELOAD_SHADOWS] is set). In addition, the contents of the shadow register shall not be updated.
- A write is performed to a shadow register which has been locked.
- A read is performed to a shadow register which has been read locked.
- A program is performed to a fuse word which has been locked.
- A read is performed to a fuse word which has been read locked.

34.4.1.4 Write Postamble

Due to internal electrical characteristics of the OTP during writes, all OTP operations (direct reads, shadow reloads, or other writes) following a write must wait at least 2 µs after the clearing of HW_OCOTP_CTRL[BUSY] following the write. The delay guarantees programming voltages on-chip reach a steady state when exiting a write sequence.

A recommended software sequence to meet the postamble requirements is as follows:

1. Issue the write and poll for HW_OCOTP_CTRL[BUSY].
2. Perform the next OTP operation.

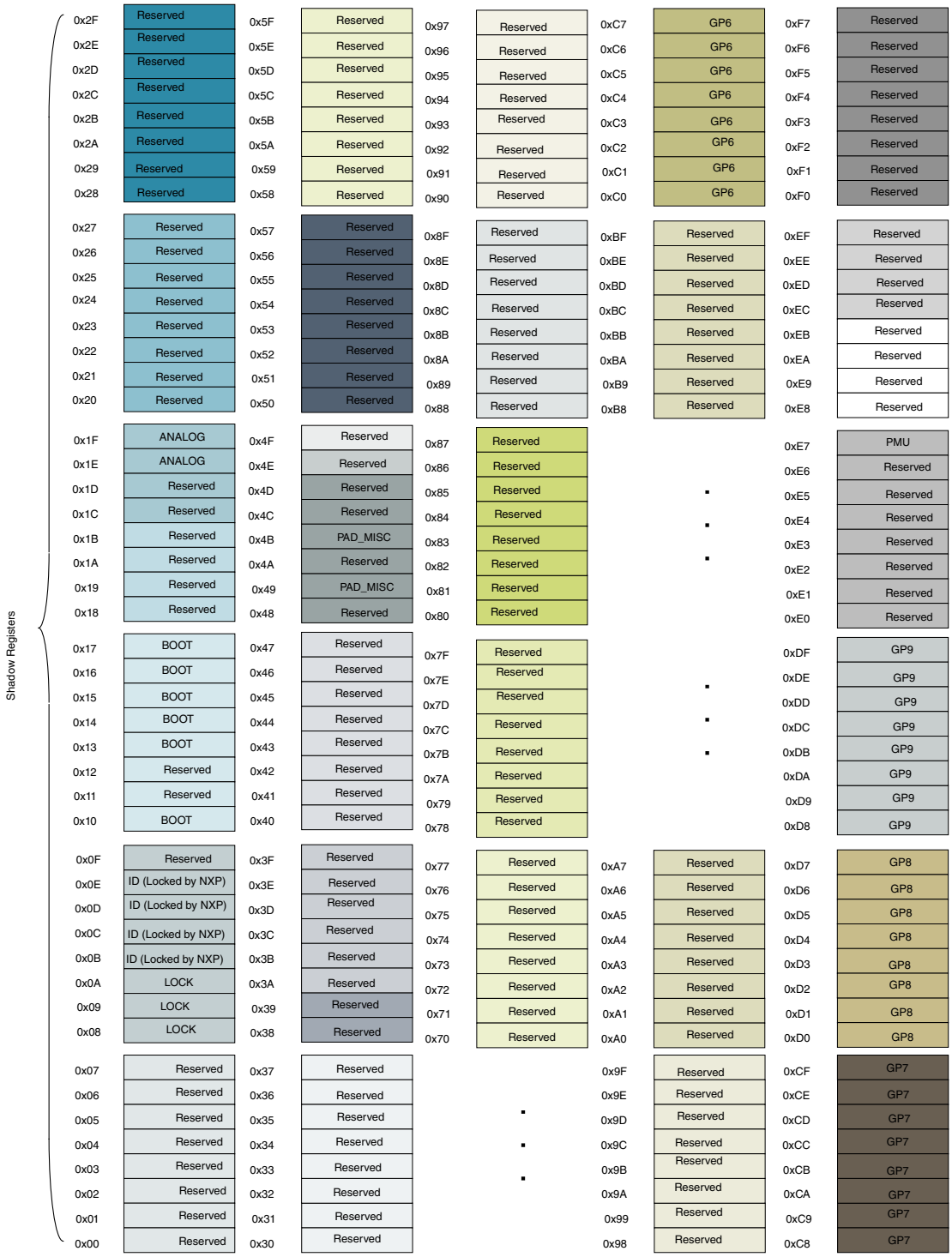
34.4.1.5 Power control of OTP memory

For every fuse operation, OCOTP_CTRL will automatically enable power of OTP memory. It is SW responsibility to shut off power OTP memory in idle status to save power.

1. After system power-up, SW needs to write 1 to HW_OCOTP_PDN[PDN] to shut off power OTP if there is not any fuse operation request.
2. After fuse program/read or MTR program/read triggered by SW are completed, SW needs to write 1 to HW_OCOTP_PDN[PDN] to shut off power OTP if there is not any new fuse operation request.

34.4.2 Fuse Shadow Memory Footprint

The OTP memory footprint is shown in the following figure. The registers are grouped by lock region. Their names correspond to the fusemap names.



Some of the reserved areas may contain secure keys. To see full memory footprint including security details, refer the i.MX7ULP Security RM.

Figure 34-2. OTP Memory Footprint

34.4.3 Behavior During Reset

The OCOTP is always active. The shadow registers automatically load the appropriate OTP contents after reset is deasserted. During this load-time HW_OCOTP_CTRL[BUSY] is set.

34.4.4 Secure JTAG control

The JTAG control fuses are used to allow or disallow JTAG access to secured resources.

Three JTAG security levels are implemented as shown in the table below.

Table 34-4. JTAG Security Level Control Bits

Security Mode	JTAG_SMODE	Description
No Debug	2'b11	The highest security level.
Secure JTAG	2'b01	Limit the JTAG access by using key based authentication mechanism.
JTAG Enable	2'b00	Low Security, all JTAG features are enabled.

34.5 Fuse Map

See the Fusemap attached with this reference manual for more information.

34.6 Unique Identification (UNIQUE_ID)

UNIQUE_ID contains 64 bits, allowing unique identification for every chip. The UNIQUE_ID number is bound to the device and cannot be altered without destroying the chip. The UNIQUE_ID is also used as a secure JTAG challenge. The user reads the chip's UNIQUE_ID to reference a server-stored response which can be used to open the chip for debug.

34.7 OCOTP Memory Map/Register Definition

The OCOTP Memory Map/Register Definition can be found here.

NOTE

Writing or reading unimplemented register address in OCOTP Controller will not send error and read data will be 0.

34.7.1 OCOTP register descriptions

OCOTP Register Reference Index

34.7.1.1 OCOTP memory map

OCOTP_CTRL base address: 410A_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	OTP Controller Control Register (HW_OCOTP_CTRL)	32	RW	0000_0000h
4h	OTP Controller Control Register (HW_OCOTP_CTRL_SET)	32	RW	0000_0000h
8h	OTP Controller Control Register (HW_OCOTP_CTRL_CLR)	32	RW	0000_0000h
Ch	OTP Controller Control Register (HW_OCOTP_CTRL_TOG)	32	RW	0000_0000h
10h	OTP Controller PDN Register (HW_OCOTP_PDN)	32	RW	0000_0000h
20h	OTP Controller Write Data Register (HW_OCOTP_DATA)	32	RW	0000_0000h
30h	OTP Controller Read Control Register (HW_OCOTP_READ_CTRL)	32	RW	0000_0000h
40h	OTP Controller Read Fuse Data Register (HW_OCOTP_READ_FUSE_DATA)	32	RO	0000_0000h
50h	OTP Controller Sticky Bit Register (HW_OCOTP_SW_STICKY)	32	RW	0000_0000h
60h	OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS)	32	RW	0000_0000h
64h	OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS_SET)	32	RW	0000_0000h
68h	OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS_CLR)	32	RW	0000_0000h
6Ch	OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS_TOG)	32	RW	0000_0000h
90h	OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS)	32	RW	0000_0000h
94h	OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS_SET)	32	RW	0000_0000h
98h	OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS_CLR)	32	RW	0000_0000h
9Ch	OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS_TOG)	32	RW	0000_0000h
A0h	OTP Controller Output Startword Register (HW_OCOTP_STARTWORD)	32	RO	0000_0000h
B0h	OTP Controller Version Register (HW_OCOTP_VERSION)	32	RO	0700_0000h
480h	OTP Controller Lock Control Register 0 (HW_OCOTP_LOCK0)	32	RO	0000_0000h
490h	OTP Controller Lock Control Register 1 (HW_OCOTP_LOCK1)	32	RO	0000_0000h
4A0h	OTP Controller Lock Control Register 2 (HW_OCOTP_LOCK2)	32	RO	0000_0000h
4B0h	Value of OTP Bank1 Word3 (ID Info.) (HW_OCOTP_CFG0)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
4C0h	Value of OTP Bank1 Word4 (ID Info.) (HW_OCOTP_CFG1)	32	RW	0000_0000h
4D0h	Value of OTP Bank1 Word5 (ID Info.) (HW_OCOTP_CFG2)	32	RW	0000_0000h
4E0h	Value of OTP Bank1 Word6 (ID Info.) (HW_OCOTP_CFG3)	32	RW	0000_0000h
4F0h	Value of OTP Bank1 Word7 (ID Info.) (HW_OCOTP_CFG4)	32	RW	0000_0000h
500h	Value of OTP Bank2 Word0 (Boot Configuration Info.) (HW_OCOTP_BOOT0)	32	RW	0000_0000h
530h	OTP Controller Boot Configuration Register 3 (HW_OCOTP_BOOT3)	32	RW	0000_0000h
540h	OTP Controller Boot Configuration Register 4 (HW_OCOTP_BOOT4)	32	RW	0000_0000h
550h	OTP Controller Boot Configuration Register 5 (HW_OCOTP_BOOT5)	32	RW	0000_0000h
560h	OTP Controller Boot Configuration Register 6 (HW_OCOTP_BOOT6)	32	RW	0000_0000h
570h	OTP Controller Boot Configuration Register 7 (HW_OCOTP_BOOT7)	32	RW	0000_0000h
5E0h	OTP Controller Analog Register 2 (HW_OCOTP_ANA2)	32	RW	0000_0000h
5F0h	OTP Controller Analog Register 3 (HW_OCOTP_ANA3)	32	RW	0000_0000h
890h	Value of OTP Bank9 Word1 (HW_OCOTP_PAD_MISC1)	32	RW	0000_0000h
8B0h	Value of OTP Bank9 Word3 (HW_OCOTP_PAD_MISC3)	32	RW	0000_0000h
1000h	Value of OTP Bank24 Word0 (GP6) (HW_OCOTP_GP60)	32	RW	0000_0000h
1010h	Value of OTP Bank24 Word1 (GP6) (HW_OCOTP_GP61)	32	RW	0000_0000h
1020h	Value of OTP Bank24 Word2 (GP6) (HW_OCOTP_GP62)	32	RW	0000_0000h
1030h	Value of OTP Bank24 Word3 (GP6) (HW_OCOTP_GP63)	32	RW	0000_0000h
1040h	Value of OTP Bank24 Word4 (GP6) (HW_OCOTP_GP64)	32	RW	0000_0000h
1050h	Value of OTP Bank24 Word5 (GP6) (HW_OCOTP_GP65)	32	RW	0000_0000h
1060h	Value of OTP Bank24 Word6 (GP6) (HW_OCOTP_GP66)	32	RW	0000_0000h
1070h	Value of OTP Bank24 Word7 (GP6) (HW_OCOTP_GP67)	32	RW	0000_0000h
1080h	Value of OTP Bank25 Word0 (GP7) (HW_OCOTP_GP70)	32	RW	0000_0000h
1090h	Value of OTP Bank25 Word1 (GP7) (HW_OCOTP_GP71)	32	RW	0000_0000h
10A0h	Value of OTP Bank25 Word2 (GP7) (HW_OCOTP_GP72)	32	RW	0000_0000h
10B0h	Value of OTP Bank25 Word3 (GP7) (HW_OCOTP_GP73)	32	RW	0000_0000h
10C0h	Value of OTP Bank25 Word4 (GP7) (HW_OCOTP_GP74)	32	RW	0000_0000h
10D0h	Value of OTP Bank25 Word5 (GP7) (HW_OCOTP_GP75)	32	RW	0000_0000h
10E0h	Value of OTP Bank25 Word6 (GP7) (HW_OCOTP_GP76)	32	RW	0000_0000h
10F0h	Value of OTP Bank25 Word7 (GP7) (HW_OCOTP_GP77)	32	RW	0000_0000h
1100h	Value of OTP Bank26 Word0 (GP8) (HW_OCOTP_GP80)	32	RW	0000_0000h
1110h	Value of OTP Bank26 Word1 (GP8) (HW_OCOTP_GP81)	32	RW	0000_0000h
1120h	Value of OTP Bank26 Word2 (GP8) (HW_OCOTP_GP82)	32	RW	0000_0000h
1130h	Value of OTP Bank26 Word3 (GP8) (HW_OCOTP_GP83)	32	RW	0000_0000h
1140h	Value of OTP Bank26 Word4 (GP8) (HW_OCOTP_GP84)	32	RW	0000_0000h
1150h	Value of OTP Bank26 Word5 (GP8) (HW_OCOTP_GP85)	32	RW	0000_0000h
1160h	Value of OTP Bank26 Word6 (GP8) (HW_OCOTP_GP86)	32	RW	0000_0000h
1170h	Value of OTP Bank26 Word7 (GP8) (HW_OCOTP_GP87)	32	RW	0000_0000h
1180h	Value of OTP Bank27 Word0 (GP9) (HW_OCOTP_GP90)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
1190h	Value of OTP Bank27 Word1 (GP9) (HW_OCOTP_GP91)	32	RW	0000_0000h
11A0h	Value of OTP Bank27 Word2 (GP9) (HW_OCOTP_GP92)	32	RW	0000_0000h
11B0h	Value of OTP Bank27 Word3 (GP9) (HW_OCOTP_GP93)	32	RW	0000_0000h
11C0h	Value of OTP Bank27 Word4 (GP9) (HW_OCOTP_GP94)	32	RW	0000_0000h
11D0h	Value of OTP Bank27 Word5 (GP9) (HW_OCOTP_GP95)	32	RW	0000_0000h
11E0h	Value of OTP Bank27 Word6 (GP9) (HW_OCOTP_GP96)	32	RW	0000_0000h
11F0h	Value of OTP Bank27 Word7 (GP9) (HW_OCOTP_GP97)	32	RW	0000_0000h
1270h	Value of OTP Bank28 Word7 (HW_OCOTP_GP107)	32	RW	0000_0000h

34.7.1.2 OTP Controller Control Register (HW_OCOTP_CTRL)

34.7.1.2.1 Offset

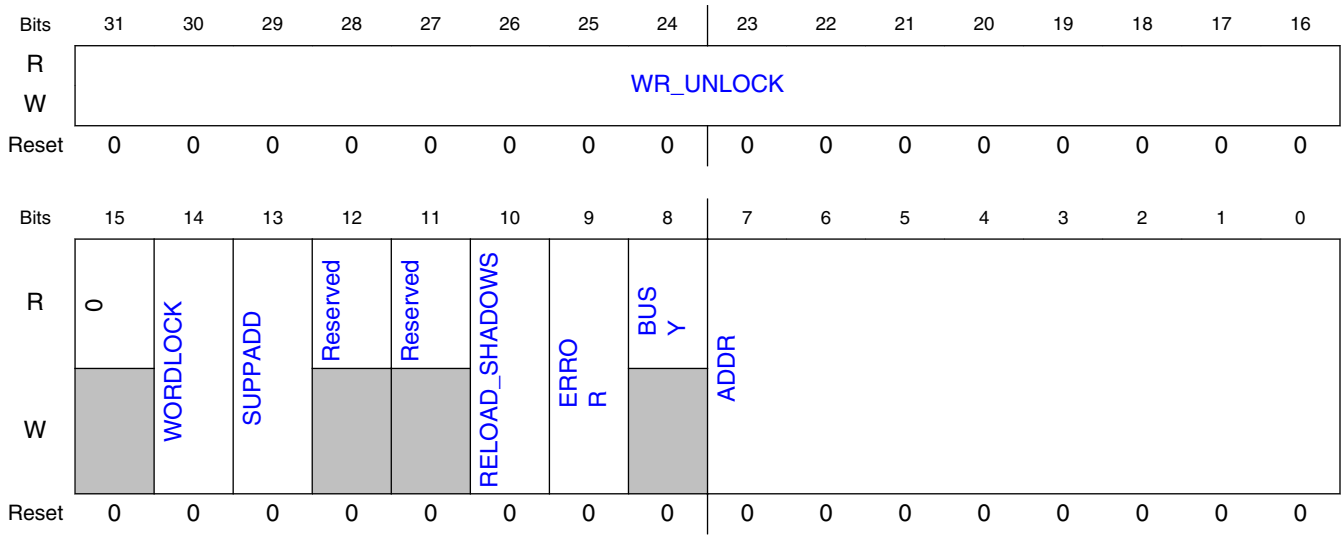
Register	Offset
HW_OCOTP_CTRL	0h

34.7.1.2.2 Function

The OCOTP Control and Status Register provides the necessary software interface for performing read and write operations to the On-Chip OTP (One-Time Programmable ROM). The control fields such as WR_UNLOCK, ADDR and BUSY/ERROR may be used in conjunction with the HW_OCOTP_DATA register to perform write operations. Read operations to the On-Chip OTP are involving ADDR, BUSY/ERROR bit field and HW_OCOTP_READ_CTRL register. Read value is saved in HW_OCOTP_READ_FUSE_DATA register.

Writing this register will set/clear/reverse all the bits of HW_OCOTP_CTRL_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_CTRL.

34.7.1.2.3 Diagram



34.7.1.2.4 Fields

Field	Function
31-16 WR_UNLOCK	Write Unlock Write 0x3E77 to enable OTP write accesses. **NOTE:** This register must be unlocked on a write-by-write basis (a write is initiated when HW_OCOTP_DATA is written), so the UNLOCK bit field must contain the correct key value during all writes to HW_OCOTP_DATA, otherwise a write shall not be initiated. This field is automatically cleared after a successful write completion (clearing of BUSY). 0000000000000000b - OTP write accesses are locked. 0011111001110111b - OPT write access is unlocked.
15 —	- Reserved
14 WORDLOCK	WORDLOCK Indicates if word has to to be locked when programming. When programming with ECC mode, it is recommended to set this bit.
13 SUPPADD	Supplementary Address This bit should be set when programming or reading supplementary address space.
12 —	Reserved
11 —	Reserved
10 RELOAD_SHADOWS	RELOAD_SHADOWS Set to force reloading the shadow registers. This operation will automatically set BUSY. Once the shadow registers have been reloaded, BUSY and RELOAD_SHADOWS are automatically cleared by the controller.
9	ERROR

Table continues on the next page...

Field	Function
ERROR	Set by the controller when an access to a locked region(OTP or shadow register) is requested. Must be cleared before any further access can be performed. This bit can only be set by the controller. This bit is also set if the Pin interface (interface to the FUSES block) is active (that is, a transaction is in execution) and software requests an access to the OTP. In this instance, the ERROR bit cannot be cleared until the Pin interface access has completed. Reset this bit by writing a 1 to the HW_OCOTP_CTRL_CLR[ERROR] bit and not by a general write.
8 BUSY	BUSY OTP controller status bit. When set, no new write access or read access to OTP(including RELOAD_SHADOWS) can be performed. Cleared by controller when access complete. After reset (or after setting RELOAD_SHADOWS), this bit is set by the controller until the HW/SW and LOCK registers are successfully copied, after which time it is automatically cleared by the controller.
7-0 ADDR	ADDRESS OTP write and read access address field. Specifies one of 256 word address locations (0x00 - 0xFF).

34.7.1.3 OTP Controller Control Register (HW_OCOTP_CTRL_SET)

34.7.1.3.1 Offset

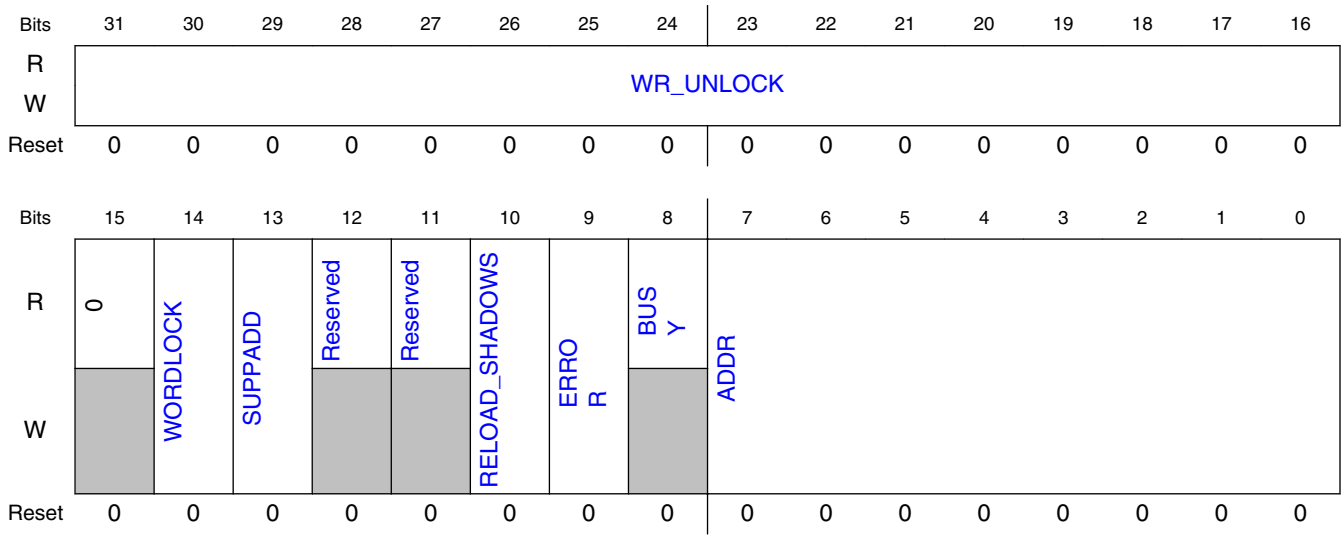
Register	Offset
HW_OCOTP_CTRL_SET	4h

34.7.1.3.2 Function

The OCOTP Control and Status Register provides the necessary software interface for performing read and write operations to the On-Chip OTP (One-Time Programmable ROM). The control fields such as WR_UNLOCK, ADDR and BUSY/ERROR may be used in conjunction with the HW_OCOTP_DATA register to perform write operations. Read operations to the On-Chip OTP are involving ADDR, BUSY/ERROR bit field and HW_OCOTP_READ_CTRL register. Read value is saved in HW_OCOTP_READ_FUSE_DATA register.

Writing this register will set/clear/reverse all the bits of HW_OCOTP_CTRL_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_CTRL.

34.7.1.3.3 Diagram



34.7.1.3.4 Fields

Field	Function
31-16 WR_UNLOCK	Write Unlock Write 0x3E77 to enable OTP write accesses. **NOTE:** This register must be unlocked on a write-by-write basis (a write is initiated when HW_OCOTP_DATA is written), so the UNLOCK bit field must contain the correct key value during all writes to HW_OCOTP_DATA, otherwise a write shall not be initiated. This field is automatically cleared after a successful write completion (clearing of BUSY).
15 —	- Reserved
14 WORDLOCK	WORDLOCK Indicates if word has to to be locked when programming. When programming with ECC mode, it is recommended to set this bit.
13 SUPPADD	Supplementary Address This bit should be set when programming or reading supplementary address space.
12 —	Reserved
11 —	Reserved
10 RELOAD_SHADOWS	RELOAD_SHADOWS Set to force reloading the shadow registers. This operation will automatically set BUSY. Once the shadow registers have been reloaded, BUSY and RELOAD_SHADOWS are automatically cleared by the controller.
9 ERROR	ERROR

Table continues on the next page...

Field	Function
	Set by the controller when an access to a locked region(OTP or shadow register) is requested. Must be cleared before any further access can be performed. This bit can only be set by the controller. This bit is also set if the Pin interface (interface to the FUSES block) is active (that is, a transaction is in execution) and software requests an access to the OTP. In this instance, the ERROR bit cannot be cleared until the Pin interface access has completed. Reset this bit by writing a 1 to the HW_OCOTP_CTRL_CLR[ERROR] bit and not by a general write.
8 BUSY	BUSY OTP controller status bit. When set, no new write access or read access to OTP(including RELOAD_SHADOWS) can be performed. Cleared by controller when access complete. After reset (or after setting RELOAD_SHADOWS), this bit is set by the controller until the HW/SW and LOCK registers are successfully copied, after which time it is automatically cleared by the controller.
7-0 ADDR	ADDRESS OTP write and read access address field. Specifies one of 256 word address locations (0x00 - 0xFF).

34.7.1.4 OTP Controller Control Register (HW_OCOTP_CTRL_CLR)

34.7.1.4.1 Offset

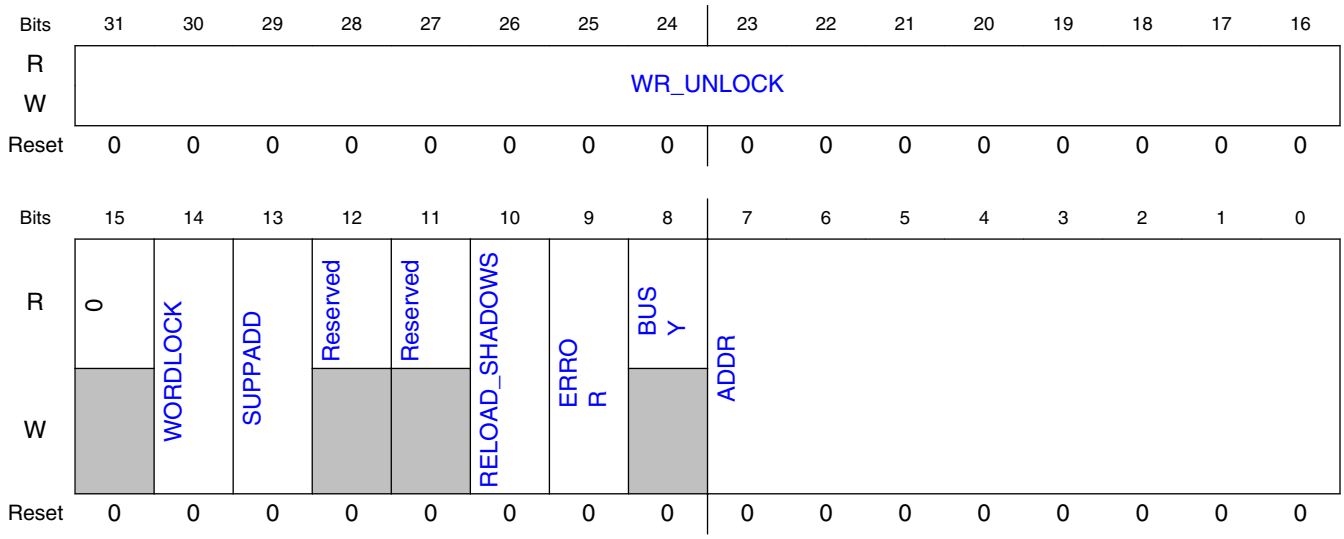
Register	Offset
HW_OCOTP_CTRL_CLR	8h

34.7.1.4.2 Function

The OCOTP Control and Status Register provides the necessary software interface for performing read and write operations to the On-Chip OTP (One-Time Programmable ROM). The control fields such as WR_UNLOCK, ADDR and BUSY/ERROR may be used in conjunction with the HW_OCOTP_DATA register to perform write operations. Read operations to the On-Chip OTP are involving ADDR, BUSY/ERROR bit field and HW_OCOTP_READ_CTRL register. Read value is saved in HW_OCOTP_READ_FUSE_DATA register.

Writing this register will set/clear/reverse all the bits of HW_OCOTP_CTRL_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_CTRL.

34.7.1.4.3 Diagram



34.7.1.4.4 Fields

Field	Function
31-16 WR_UNLOCK	Write Unlock Write 0x3E77 to enable OTP write accesses. **NOTE:** This register must be unlocked on a write-by-write basis (a write is initiated when HW_OCOTP_DATA is written), so the UNLOCK bit field must contain the correct key value during all writes to HW_OCOTP_DATA, otherwise a write shall not be initiated. This field is automatically cleared after a successful write completion (clearing of BUSY).
15 —	- Reserved
14 WORDLOCK	WORDLOCK Indicates if word has to to be locked when programming. When programming with ECC mode, it is recommended to set this bit.
13 SUPPADD	Supplementary Address This bit should be set when programming or reading supplementary address space.
12 —	Reserved
11 —	Reserved
10 RELOAD_SHADOWS	RELOAD_SHADOWS Set to force reloading the shadow registers. This operation will automatically set BUSY. Once the shadow registers have been reloaded, BUSY and RELOAD_SHADOWS are automatically cleared by the controller.
9 ERROR	ERROR

Table continues on the next page...

Field	Function
	Set by the controller when an access to a locked region(OTP or shadow register) is requested. Must be cleared before any further access can be performed. This bit can only be set by the controller. This bit is also set if the Pin interface (interface to the FUSES block) is active (that is, a transaction is in execution) and software requests an access to the OTP. In this instance, the ERROR bit cannot be cleared until the Pin interface access has completed. Reset this bit by writing a 1 to the HW_OCOTP_CTRL_CLR[ERROR] bit and not by a general write.
8 BUSY	BUSY OTP controller status bit. When set, no new write access or read access to OTP(including RELOAD_SHADOWS) can be performed. Cleared by controller when access complete. After reset (or after setting RELOAD_SHADOWS), this bit is set by the controller until the HW/SW and LOCK registers are successfully copied, after which time it is automatically cleared by the controller.
7-0 ADDR	ADDRESS OTP write and read access address field. Specifies one of 256 word address locations (0x00 - 0xFF).

34.7.1.5 OTP Controller Control Register (HW_OCOTP_CTRL_TOG)

34.7.1.5.1 Offset

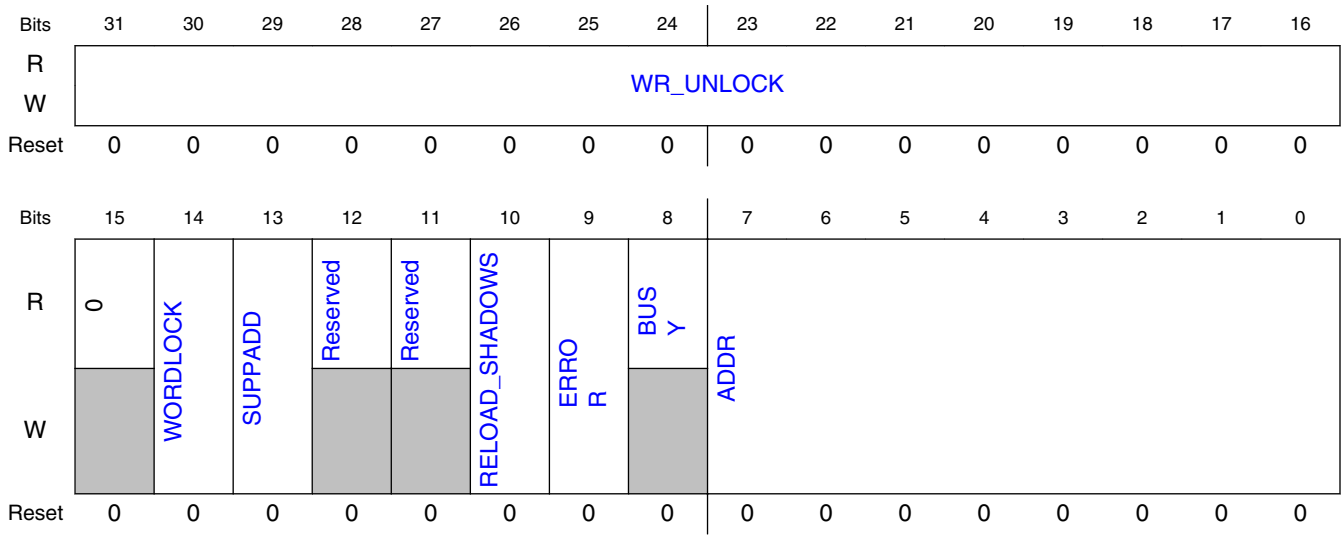
Register	Offset
HW_OCOTP_CTRL_TOG	Ch

34.7.1.5.2 Function

The OCOTP Control and Status Register provides the necessary software interface for performing read and write operations to the On-Chip OTP (One-Time Programmable ROM). The control fields such as WR_UNLOCK, ADDR and BUSY/ERROR may be used in conjunction with the HW_OCOTP_DATA register to perform write operations. Read operations to the On-Chip OTP are involving ADDR, BUSY/ERROR bit field and HW_OCOTP_READ_CTRL register. Read value is saved in HW_OCOTP_READ_FUSE_DATA register.

Writing this register will set/clear/reverse all the bits of HW_OCOTP_CTRL_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_CTRL.

34.7.1.5.3 Diagram



34.7.1.5.4 Fields

Field	Function
31-16 WR_UNLOCK	Write Unlock Write 0x3E77 to enable OTP write accesses. **NOTE:** This register must be unlocked on a write-by-write basis (a write is initiated when HW_OCOTP_DATA is written), so the UNLOCK bit field must contain the correct key value during all writes to HW_OCOTP_DATA, otherwise a write shall not be initiated. This field is automatically cleared after a successful write completion (clearing of BUSY).
15 —	- Reserved
14 WORDLOCK	WORDLOCK Indicates if word has to to be locked when programming. When programming with ECC mode, it is recommended to set this bit.
13 SUPPADD	Supplementary Address This bit should be set when programming or reading supplementary address space.
12 —	Reserved
11 —	Reserved
10 RELOAD_SHADOWS	RELOAD_SHADOWS Set to force reloading the shadow registers. This operation will automatically set BUSY. Once the shadow registers have been reloaded, BUSY and RELOAD_SHADOWS are automatically cleared by the controller.
9 ERROR	ERROR

Table continues on the next page...

Field	Function
	Set by the controller when an access to a locked region(OTP or shadow register) is requested. Must be cleared before any further access can be performed. This bit can only be set by the controller. This bit is also set if the Pin interface (interface to the FUSES block) is active (that is, a transaction is in execution) and software requests an access to the OTP. In this instance, the ERROR bit cannot be cleared until the Pin interface access has completed. Reset this bit by writing a 1 to the HW_OCOTP_CTRL_CLR[ERROR] bit and not by a general write.
8 BUSY	BUSY OTP controller status bit. When set, no new write access or read access to OTP(including RELOAD_SHADOWS) can be performed. Cleared by controller when access complete. After reset (or after setting RELOAD_SHADOWS), this bit is set by the controller until the HW/SW and LOCK registers are successfully copied, after which time it is automatically cleared by the controller.
7-0 ADDR	ADDRESS OTP write and read access address field. Specifies one of 256 word address locations (0x00 - 0xFF).

34.7.1.6 OTP Controller PDN Register (HW_OCOTP_PDN)

34.7.1.6.1 Offset

Register	Offset
HW_OCOTP_PDN	10h

34.7.1.6.2 Function

This register specifies PDN or power switch control for eFUSE macro and is used to reduce power consumption.

34.7.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															PDN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.6.4 Fields

Field	Function
31-1 —	- Reserved
0 PDN	PDN This bit indicates PDN value of OTP memory. If a 1 is written to this bit, PDN will be 0. Writing a 0 has no effect. **NOTE:** The software (SW) needs to write 1 to this bit to shut off power of OTP memory after system power up. At the beginning of every fuse operation, the hardware (HW) will auto enable power of OTP memory. After every fuse operation, the software (SW) also needs to write 1 to this bit to shut off power of OTP memory to avoid unnecessary power wastage.

34.7.1.7 OTP Controller Write Data Register (HW_OCOTP_DATA)

34.7.1.7.1 Offset

Register	Offset
HW_OCOTP_DATA	20h

34.7.1.7.2 Function

This register is used in conjunction with HW_OCOTP_CTRL to perform one-time writes to the OTP. See the Fuse and Shadow Register Writes section for operating details.

34.7.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.7.4 Fields

Field	Function
31-0 DATA	DATA Used to initiate a write to OTP. See the Fuse and Shadow Register Writes section for operating details.

34.7.1.8 OTP Controller Read Control Register (HW_OCOTP_READ_CTRL)

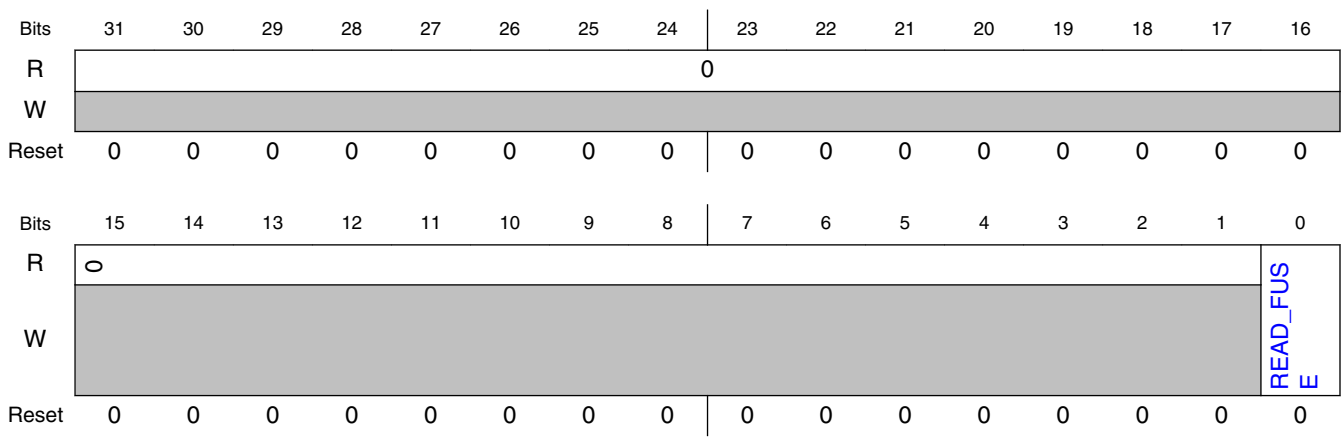
34.7.1.8.1 Offset

Register	Offset
HW_OCOTP_READ_CTRL	30h

34.7.1.8.2 Function

This register is used in conjunction with HW_OCOTP_CTRL to perform one time read to the OTP. See the Fuse and Shadow Register Read section for operating details.

34.7.1.8.3 Diagram



34.7.1.8.4 Fields

Field	Function
31-1	-

Table continues on the next page...

Field	Function
—	Reserved
0 READ_FUSE	READ_FUSE Used to initiate a read to OTP. See the Fuse and Shadow Register Read section for operating details.

34.7.1.9 OTP Controller Read Fuse Data Register (HW_OCOTP_READ_FUSE_DATA)

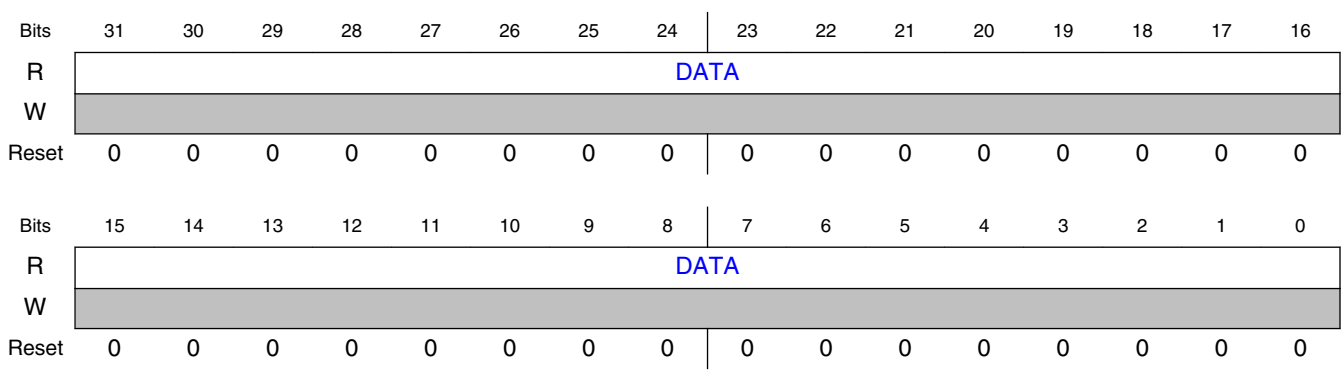
34.7.1.9.1 Offset

Register	Offset
HW_OCOTP_READ_FUSE_DATA	40h

34.7.1.9.2 Function

The data read from OTP

34.7.1.9.3 Diagram



34.7.1.9.4 Fields

Field	Function
31-0 DATA	DATA The data read from OTP. See the Fuse and Shadow Register Read section for operating details.

34.7.1.10 OTP Controller Sticky Bit Register (HW_OCOTP_SW_STICKY)

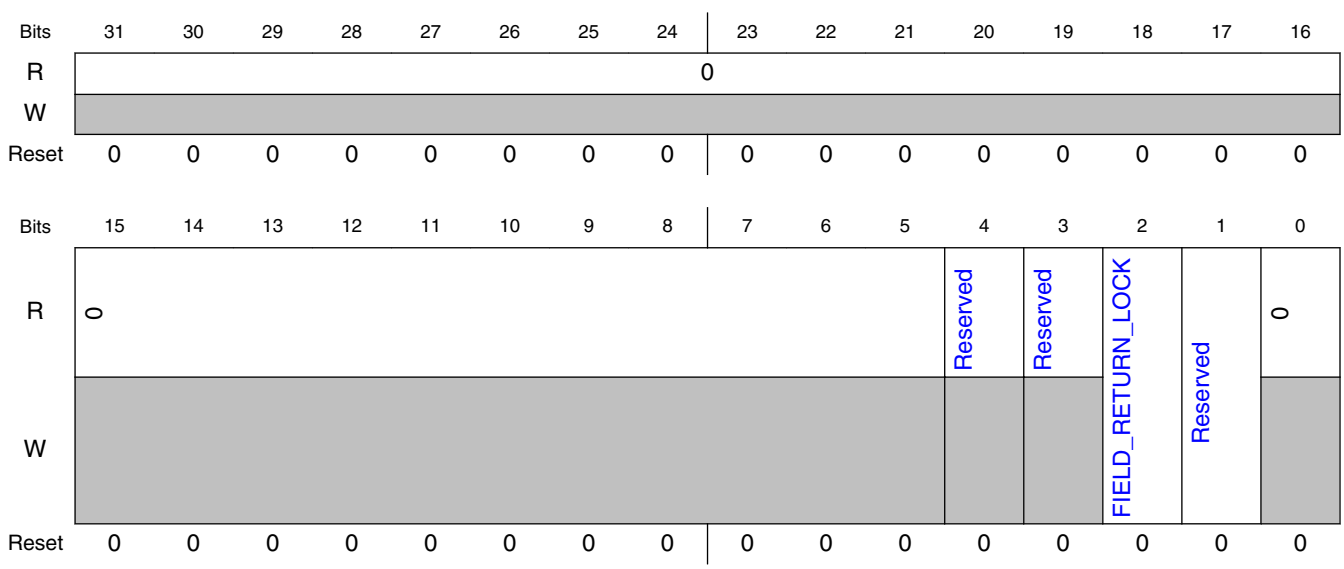
34.7.1.10.1 Offset

Register	Offset
HW_OCOTP_SW_STICKY	50h

34.7.1.10.2 Function

Some sticky bits are used by SW to lock some fuse area, shadow registers and other features.

34.7.1.10.3 Diagram



34.7.1.10.4 Fields

Field	Function
31-5 —	- Reserved
4	Reserved

Table continues on the next page...

Field	Function
—	
3 —	Reserved
2 FIELD_RETURN_LOCK	FIELD_RETURN_LOCK Shadow register write and OTP write lock for FIELD_RETURN fuse. When set, the writing of this region's shadow register and OTP fuse word are blocked.Once this bit is set, it is always high unless a reset is issued.
1 —	- Reserved
0 —	- Reserved

34.7.1.11 OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS)

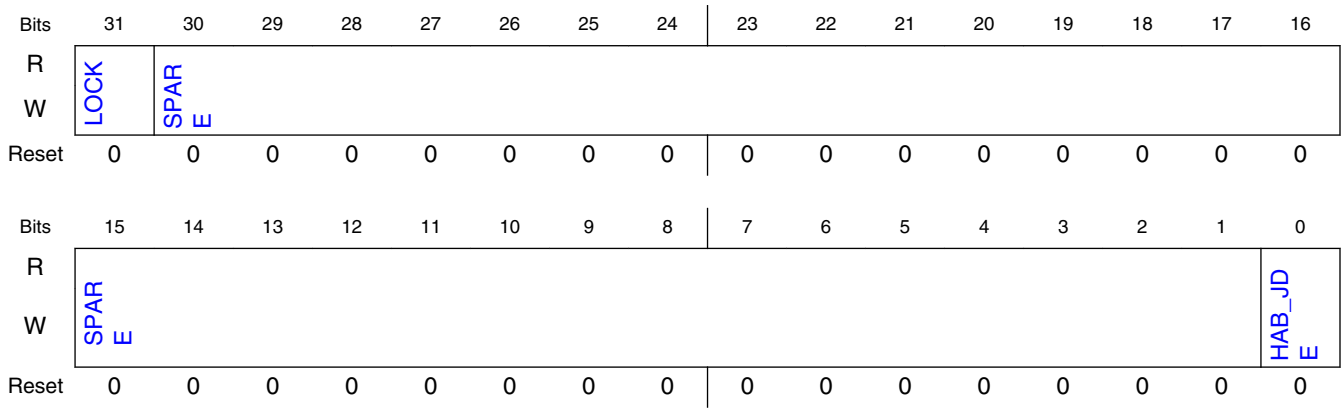
34.7.1.11.1 Offset

Register	Offset
HW_OCOTP_SCS	60h

34.7.1.11.2 Function

This register holds volatile configuration values that can be set and locked by trusted software. All values are returned to their default values after reset. Writing this register will set/clear/reverse all the bits of HW_OCOTP_SCS_SET/CLR/TOG respectively, which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_SCS.

34.7.1.11.3 Diagram



34.7.1.11.4 Fields

Field	Function
31 LOCK	LOCK When set, all the bits in this register are locked and can not be changed through SW programming. This bit is only reset after a reset is issued.
30-1 SPARE	SPARE Unallocated read/write bits for implementation specific software use.
0 HAB_JDE	HAB_JDE HAB JTAG Debug Enable. This bit is used by the HAB to enable JTAG debugging, assuming that a properly signed command to do so is found and validated by the HAB. The HAB must lock the register before passing control to the OS whether or not JTAG debugging has been enabled. Once JTAG is enabled by this bit, it can not be disabled unless the system is reset by reset. 0b - JTAG debugging is not enabled by the HAB (it may still be enabled by other mechanisms). 1b - JTAG debugging is enabled by the HAB (though this signal may be gated off).

34.7.1.12 OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS_SET)

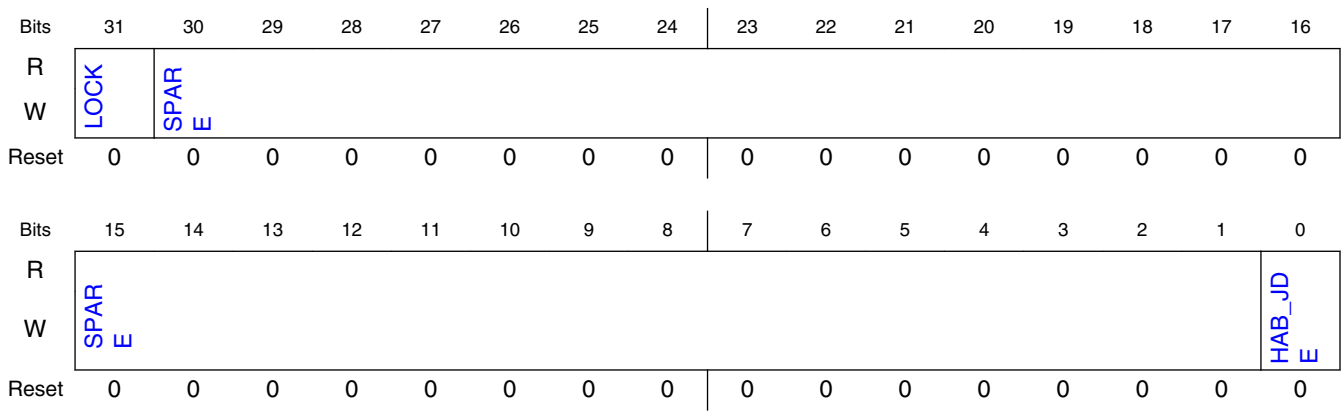
34.7.1.12.1 Offset

Register	Offset
HW_OCOTP_SCS_SET	64h

34.7.1.12.2 Function

This register holds volatile configuration values that can be set and locked by trusted software. All values are returned to their default values after reset. Writing this register will set/clear/reverse all the bits of HW_OCOTP_SCS_SET/CLR/TOG respectively, which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_SCS.

34.7.1.12.3 Diagram



34.7.1.12.4 Fields

Field	Function
31 LOCK	LOCK When set, all the bits in this register are locked and can not be changed through SW programming. This bit is only reset after a reset is issued.
30-1 SPARE	SPARE Unallocated read/write bits for implementation specific software use.
0 HAB_JDE	HAB_JDE HAB JTAG Debug Enable. This bit is used by the HAB to enable JTAG debugging, assuming that a properly signed command to do so is found and validated by the HAB. The HAB must lock the register before passing control to the OS whether or not JTAG debugging has been enabled. Once JTAG is enabled by this bit, it can not be disabled unless the system is reset by reset.

34.7.1.13 OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS_CLR)

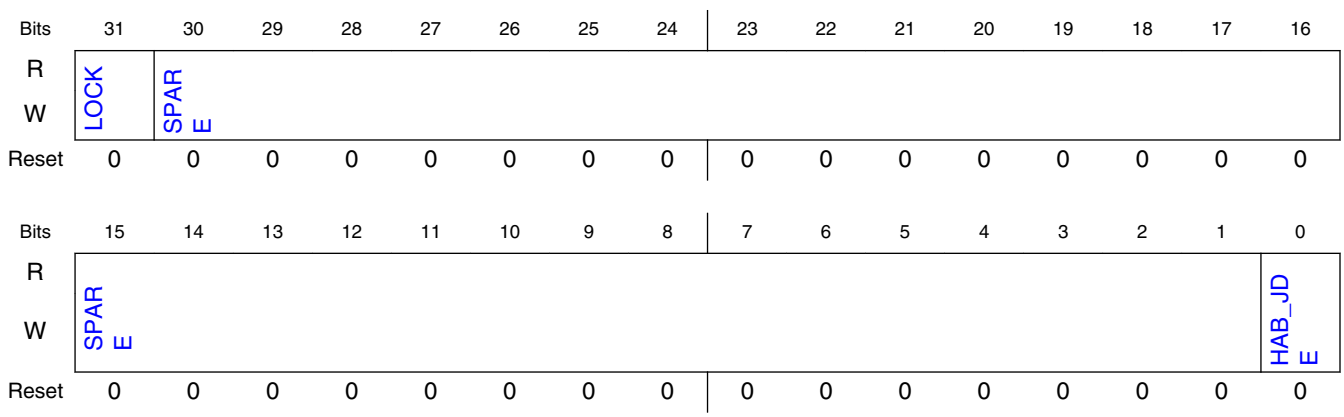
34.7.1.13.1 Offset

Register	Offset
HW_OCOTP_SCS_CLR	68h

34.7.1.13.2 Function

This register holds volatile configuration values that can be set and locked by trusted software. All values are returned to their default values after reset. Writing this register will set/clear/reverse all the bits of HW_OCOTP_SCS_SET/CLR/TOG respectively, which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_SCS.

34.7.1.13.3 Diagram



34.7.1.13.4 Fields

Field	Function
31 LOCK	LOCK When set, all the bits in this register are locked and can not be changed through SW programming. This bit is only reset after a reset is issued.
30-1 SPARE	SPARE Unallocated read/write bits for implementation specific software use.
0 HAB_JDE	HAB_JDE HAB JTAG Debug Enable. This bit is used by the HAB to enable JTAG debugging, assuming that a properly signed command to do so is found and validated by the HAB. The HAB must lock the register before passing control to the OS whether or not JTAG debugging has been enabled. Once JTAG is enabled by this bit, it can not be disabled unless the system is reset by reset.

34.7.1.14 OTP Controller Software Controllable Signals Register (HW_OCOTP_SCS_TOG)

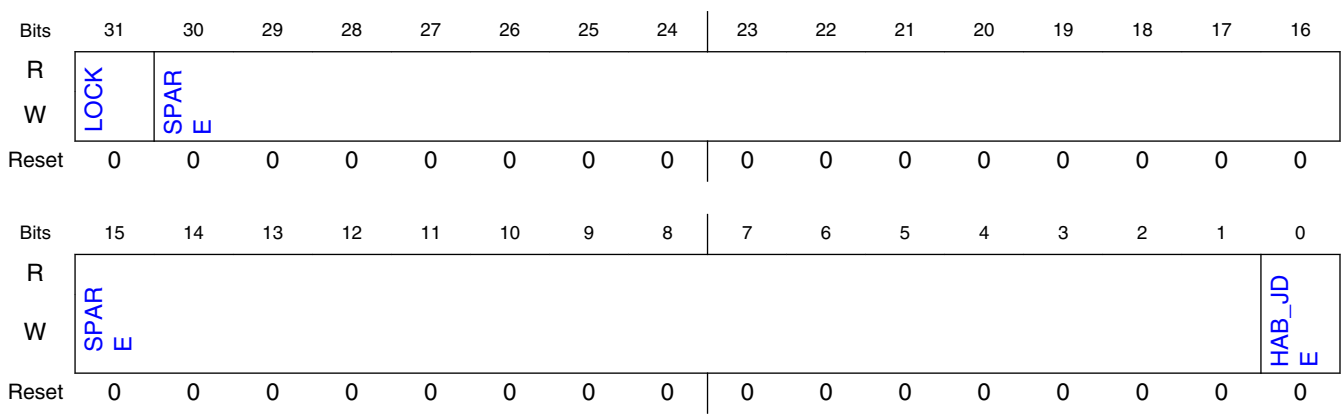
34.7.1.14.1 Offset

Register	Offset
HW_OCOTP_SCS_TOG	6Ch

34.7.1.14.2 Function

This register holds volatile configuration values that can be set and locked by trusted software. All values are returned to their default values after reset. Writing this register will set/clear/reverse all the bits of HW_OCOTP_SCS_SET/CLR/TOG respectively, which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_SCS.

34.7.1.14.3 Diagram



34.7.1.14.4 Fields

Field	Function
31 LOCK	LOCK When set, all the bits in this register are locked and can not be changed through SW programming. This bit is only reset after a reset is issued.
30-1 SPARE	SPARE Unallocated read/write bits for implementation specific software use.

Table continues on the next page...

Field	Function
0 HAB_JDE	HAB_JDE HAB JTAG Debug Enable. This bit is used by the HAB to enable JTAG debugging, assuming that a properly signed command to do so is found and validated by the HAB. The HAB must lock the register before passing control to the OS whether or not JTAG debugging has been enabled. Once JTAG is enabled by this bit, it can not be disabled unless the system is reset by reset.

34.7.1.15 OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS)

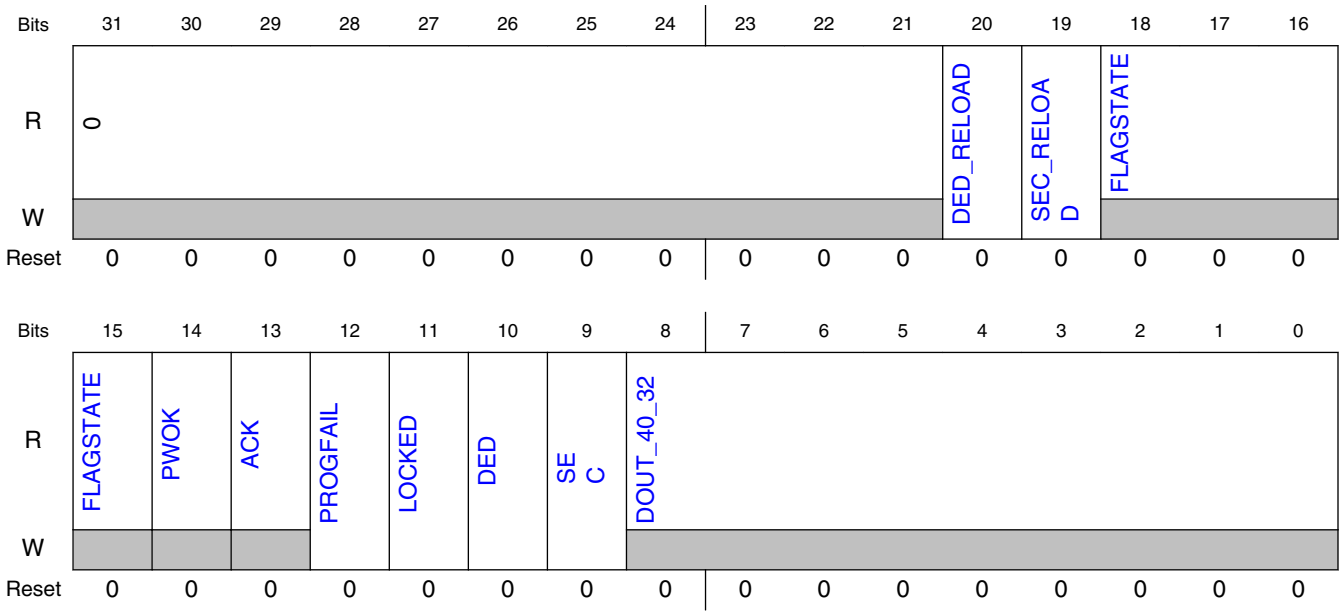
34.7.1.15.1 Offset

Register	Offset
HW_OCOTP_OUT_STATUS	90h

34.7.1.15.2 Function

This register stores output signals of OTP memory for programming and reading OTP memory. Writing this register will set/clear/reverse all the bits of HW_OCOTP_OUT_STATUS_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_OUT_STATUS.

34.7.1.15.3 Diagram



34.7.1.15.4 Fields

Field	Function
31-21 —	- Reserved
20 DED_RELOAD	DED_RELOAD Indicates double error detection ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
19 SEC_RELOAD	SEC_RELOAD Indicates single error correction ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
18-15 FLAGSTATE	FLAGSTATE Indicates OTP memory current state.
14 PWOK	PWOK Indicates that power vdd and vdd1 of OTP memory are in operating range.
13 ACK	ACK indicates operation acknowledgement from OTP memory
12 PROGFAIL	PROGFAIL Indicates programming is failed. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
11 LOCKED	LOCKED

Table continues on the next page...

Field	Function
	Indicates if word is locked or not when programming or reading. When this bit is asserted when programming one fuse word, it means current programming operation is failed since the fuse word is already locked. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
10 DED	DED Indicates 2 errors have been detected (no correction) when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
9 SEC	SEC Indicates one error has been corrected in data or ECC syndrome when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
8-0 DOUT_40_32	DOUT_40_32 Stored out[40:32] from OTP memory when reading one fuse word.

34.7.1.16 OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS_SET)

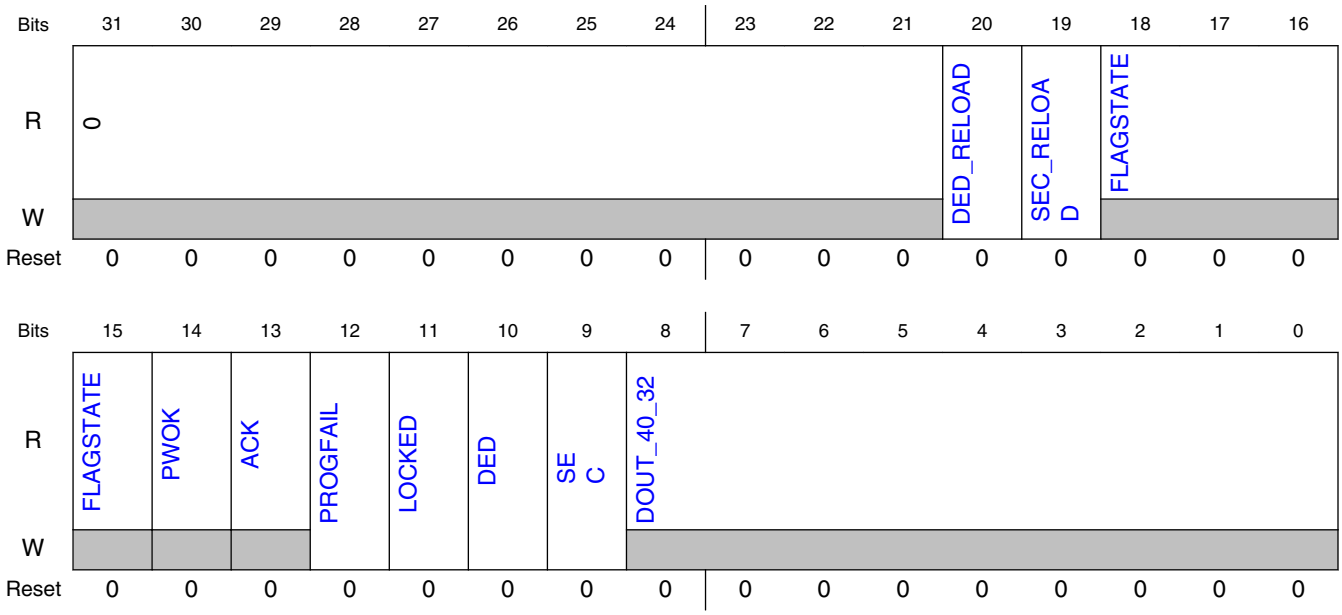
34.7.1.16.1 Offset

Register	Offset
HW_OCOTP_OUT_STATUS_SET	94h

34.7.1.16.2 Function

This register stores output signals of OTP memory for programming and reading OTP memory. Writing this register will set/clear/reverse all the bits of HW_OCOTP_OUT_STATUS_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_OUT_STATUS.

34.7.1.16.3 Diagram



34.7.1.16.4 Fields

Field	Function
31-21 —	- Reserved
20 DED_RELOAD	DED_RELOAD Indicates double error detection ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
19 SEC_RELOAD	SEC_RELOAD Indicates single error correction ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
18-15 FLAGSTATE	FLAGSTATE Indicates OTP memory current state.
14 PWOK	PWOK Indicates that power vdd and vdd1 of OTP memory are in operating range.
13 ACK	ACK indicates operation acknowledgement from OTP memory
12 PROGFAIL	PROGFAIL Indicates programming is failed. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
11 LOCKED	LOCKED

Table continues on the next page...

Field	Function
	Indicates if word is locked or not when programming or reading. When this bit is asserted when programming one fuse word, it means current programming operation is failed since the fuse word is already locked. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
10 DED	DED Indicates 2 errors have been detected (no correction) when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
9 SEC	SEC Indicates one error has been corrected in data or ECC syndrome when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
8-0 DOUT_40_32	DOUT_40_32 Stored out[40:32] from OTP memory when reading one fuse word.

34.7.1.17 OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS_CLR)

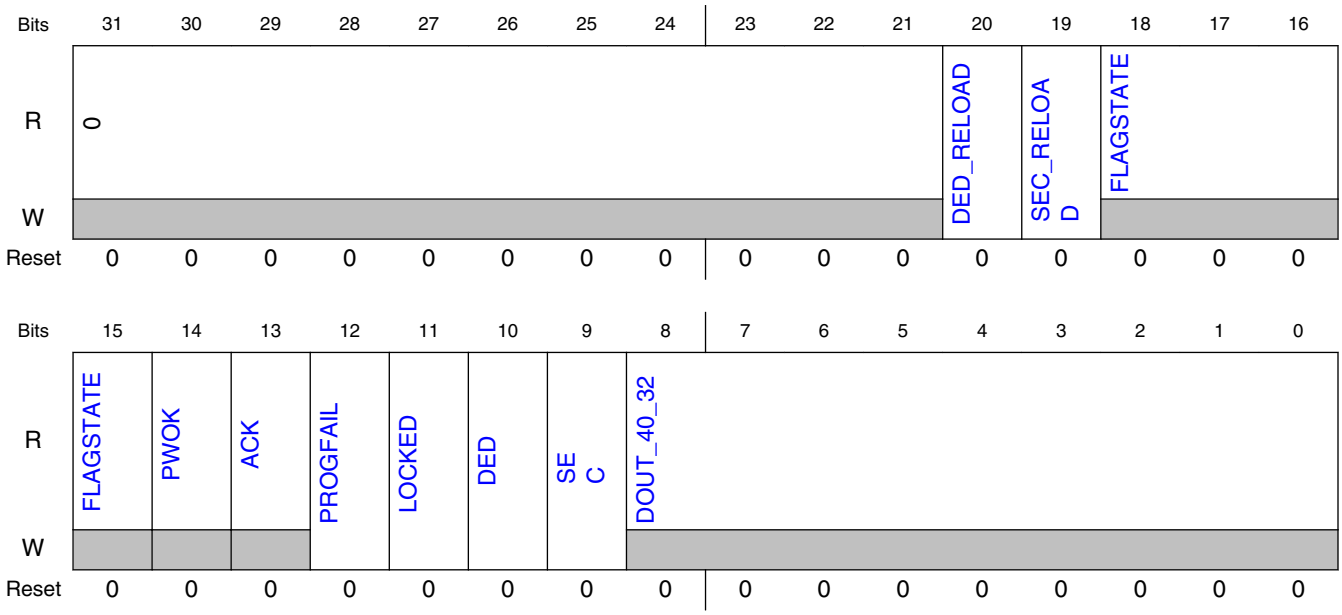
34.7.1.17.1 Offset

Register	Offset
HW_OCOTP_OUT_STATUS_CLR	98h

34.7.1.17.2 Function

This register stores output signals of OTP memory for programming and reading OTP memory. Writing this register will set/clear/reverse all the bits of HW_OCOTP_OUT_STATUS_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_OUT_STATUS.

34.7.1.17.3 Diagram



34.7.1.17.4 Fields

Field	Function
31-21 —	- Reserved
20 DED_RELOAD	DED_RELOAD Indicates double error detection ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
19 SEC_RELOAD	SEC_RELOAD Indicates single error correction ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
18-15 FLAGSTATE	FLAGSTATE Indicates OTP memory current state.
14 PWOK	PWOK Indicates that power vdd and vdd1 of OTP memory are in operating range.
13 ACK	ACK indicates operation acknowledgement from OTP memory
12 PROGFAIL	PROGFAIL Indicates programming is failed. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
11 LOCKED	LOCKED

Table continues on the next page...

Field	Function
	Indicates if word is locked or not when programming or reading. When this bit is asserted when programming one fuse word, it means current programming operation is failed since the fuse word is already locked. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
10 DED	DED Indicates 2 errors have been detected (no correction) when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
9 SEC	SEC Indicates one error has been corrected in data or ECC syndrome when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
8-0 DOUT_40_32	DOUT_40_32 Stored out[40:32] from OTP memory when reading one fuse word.

34.7.1.18 OTP Controller Output Status Register (HW_OCOTP_OUT_STATUS_TOG)

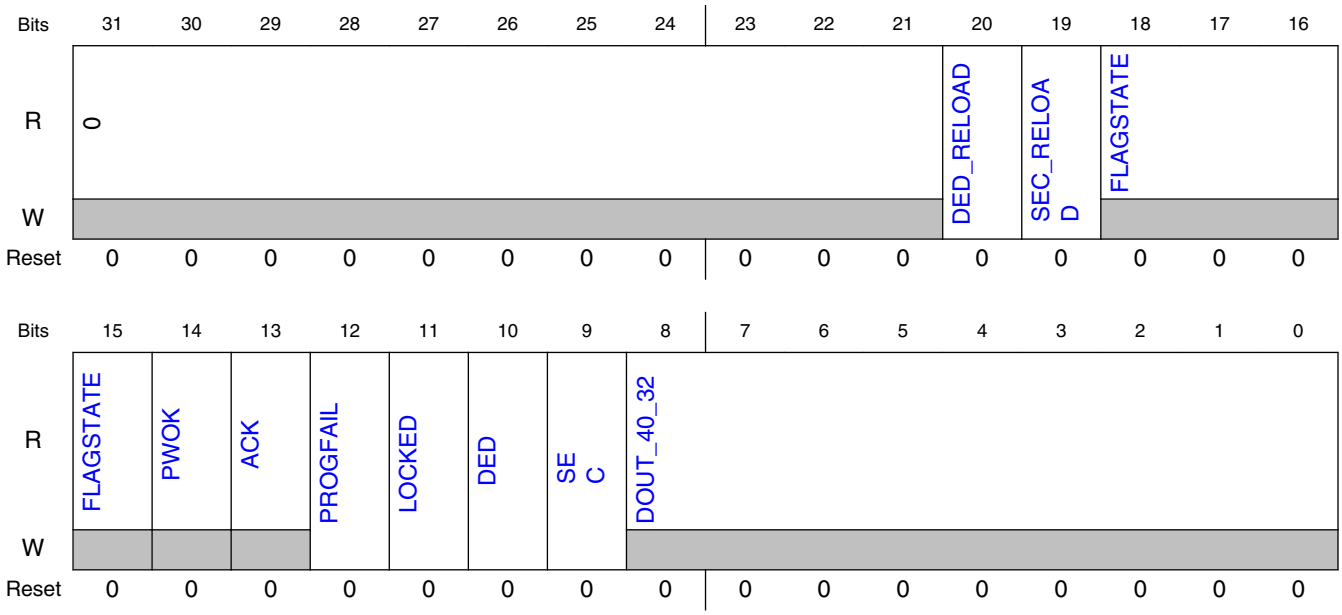
34.7.1.18.1 Offset

Register	Offset
HW_OCOTP_OUT_STATUS_TOG	9Ch

34.7.1.18.2 Function

This register stores output signals of OTP memory for programming and reading OTP memory. Writing this register will set/clear/reverse all the bits of HW_OCOTP_OUT_STATUS_SET/CLR/TOG respectively which are 1 in write data. Other bits will not change. Reading this register will return the value of HW_OCOTP_OUT_STATUS.

34.7.1.18.3 Diagram



34.7.1.18.4 Fields

Field	Function
31-21 —	- Reserved
20 DED_RELOAD	DED_RELOAD Indicates double error detection ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
19 SEC_RELOAD	SEC_RELOAD Indicates single error correction ever happens in reset reload or reload by setting HW_OCOTP_CTRL[RELOAD_SHADOWS] bit.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
18-15 FLAGSTATE	FLAGSTATE Indicates OTP memory current state.
14 PWOK	PWOK Indicates that power vdd and vdd1 of OTP memory are in operating range.
13 ACK	ACK indicates operation acknowledgement from OTP memory
12 PROGFAIL	PROGFAIL Indicates programming is failed. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
11 LOCKED	LOCKED

Table continues on the next page...

Field	Function
	Indicates if word is locked or not when programming or reading. When this bit is asserted when programming one fuse word, it means current programming operation is failed since the fuse word is already locked. This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
10 DED	DED Indicates 2 errors have been detected (no correction) when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
9 SEC	SEC Indicates one error has been corrected in data or ECC syndrome when reading one fuse word.This bit can only be set by the controller. Reset this bit by writing a 1 to HW_OCOTP_CTRL_CLR[ERROR] and not by a general write.
8-0 DOUT_40_32	DOUT_40_32 Stored out[40:32] from OTP memory when reading one fuse word.

34.7.1.19 OTP Controller Output Startword Register (HW_OCOTP_STARTWORD)

34.7.1.19.1 Offset

Register	Offset
HW_OCOTP_STARTWORD	A0h

34.7.1.19.2 Function

This register stores the output startword value from OTP memory.

34.7.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STARTWORD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.19.4 Fields

Field	Function
31-16 —	- Reserved
15-0 STARTWORD	STARTWORD Stores the output startword value from OTP memory

34.7.1.20 OTP Controller Version Register (HW_OCOTP_VERSION)

34.7.1.20.1 Offset

Register	Offset
HW_OCOTP_VERSION	B0h

34.7.1.20.2 Function

This register indicates the RTL version in use.

34.7.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.20.4 Fields

Field	Function
31-24	MAJOR

Table continues on the next page...

Field	Function
MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
23-16 MINOR	MINOR Fixed read-only value reflecting the MINOR field of the RTL version.
15-0 STEP	STEP Fixed read-only value reflecting the stepping of the RTL version.

34.7.1.21 OTP Controller Lock Control Register 0 (HW_OCOTP_LOCK0)

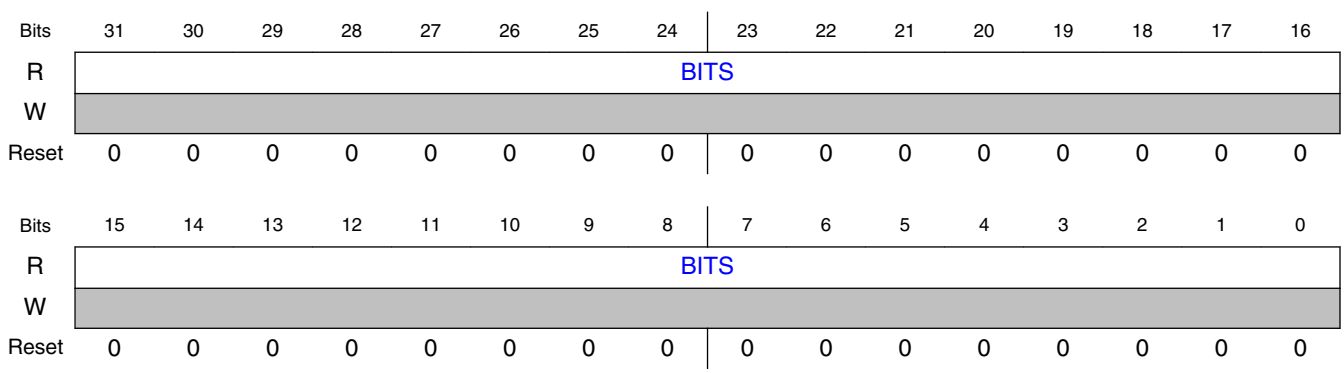
34.7.1.21.1 Offset

Register	Offset
HW_OCOTP_LOCK0	480h

34.7.1.21.2 Function

Shadowed memory mapped access to OTP Bank 1, word 0 (ADDR = 0x08).

34.7.1.21.3 Diagram



34.7.1.21.4 Fields

Field	Function
31-0 BITS	BITS Status of shadow register and OTP write/read lock .

34.7.1.22 OTP Controller Lock Control Register 1 (HW_OCOTP_LOCK1)

34.7.1.22.1 Offset

Register	Offset
HW_OCOTP_LOCK1	490h

34.7.1.22.2 Function

Shadowed memory mapped access to OTP Bank 1, word 1 (ADDR = 0x09).

34.7.1.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.22.4 Fields

Field	Function
31-0 BITS	BITS Status of shadow register and OTP write/read lock.

34.7.1.23 OTP Controller Lock Control Register 2 (HW_OCOTP_LOCK2)

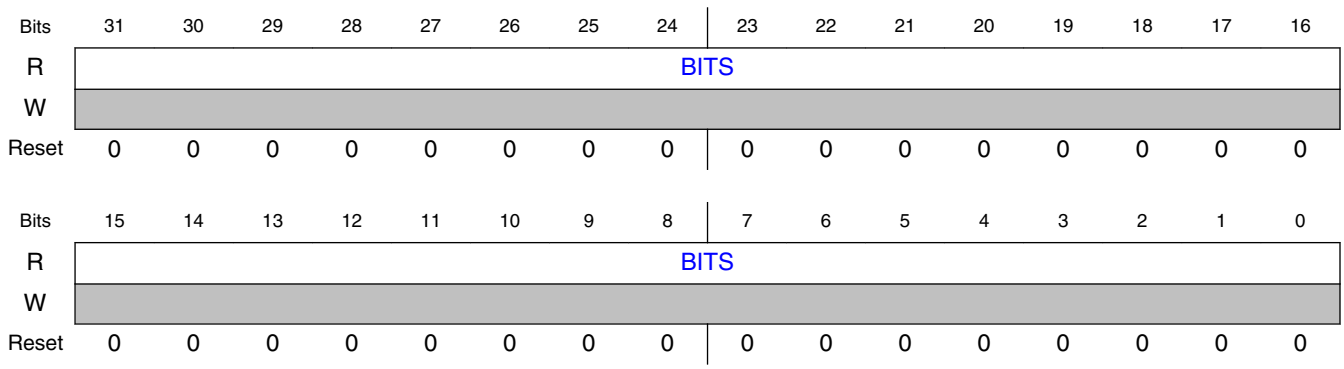
34.7.1.23.1 Offset

Register	Offset
HW_OCOTP_LOCK2	4A0h

34.7.1.23.2 Function

Shadowed memory mapped access to OTP Bank 1, word 2 (ADDR = 0x0a).

34.7.1.23.3 Diagram



34.7.1.23.4 Fields

Field	Function
31-0 BITS	BITS Status of shadow register and OTP write/read lock.

34.7.1.24 Value of OTP Bank1 Word3 (ID Info.) (HW_OCOTP_CFG0)

34.7.1.24.1 Offset

Register	Offset
HW_OCOTP_CFG0	4B0h

34.7.1.24.2 Function

This register contains bits of the Unique ID used as individual challenge by the secure JTAG challenge/response mechanism. Shadowed memory mapped access to OTP Bank 1, word 3 (ADDR = 0x0b).

34.7.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.24.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 1, word 3 (ADDR = 0x0b). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.25 Value of OTP Bank1 Word4 (ID Info.) (HW_OCOTP_CFG1)

34.7.1.25.1 Offset

Register	Offset
HW_OCOTP_CFG1	4C0h

34.7.1.25.2 Function

This register contains bits of the Unique ID used as individual challenge by the secure JTAG challenge/response mechanism. Shadowed memory mapped access to OTP Bank 1, word 4 (ADDR = 0x0c).

34.7.1.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.25.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 1, word 4 (ADDR = 0x0c). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.26 Value of OTP Bank1 Word5 (ID Info.) (HW_OCOTP_CFG2)

34.7.1.26.1 Offset

Register	Offset
HW_OCOTP_CFG2	4D0h

34.7.1.26.2 Function

This register contains bits of the Unique ID used as individual challenge by the secure JTAG challenge/response mechanism. Shadowed memory mapped access to OTP Bank 1, word 5 (ADDR = 0x0d).

34.7.1.26.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.26.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 1, word 5 (ADDR = 0x0d). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.27 Value of OTP Bank1 Word6 (ID Info.) (HW_OCOTP_CFG3)

34.7.1.27.1 Offset

Register	Offset
HW_OCOTP_CFG3	4E0h

34.7.1.27.2 Function

This register contains bits of the Unique ID used as individual challenge by the secure JTAG challenge/response mechanism. Shadowed memory mapped access to OTP Bank 1, word 6 (ADDR = 0x0e).

34.7.1.27.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.27.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 1, word 6 (ADDR = 0x0e). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.28 Value of OTP Bank1 Word7 (ID Info.) (HW_OCOTP_CFG4)

34.7.1.28.1 Offset

Register	Offset
HW_OCOTP_CFG4	4F0h

34.7.1.28.2 Function

This register contains bits of the Unique ID used as individual challenge by the secure JTAG challenge/response mechanism. Shadowed memory mapped access to OTP Bank 1, word 7 (ADDR = 0x0f).

34.7.1.28.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.28.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 1, word 7 (ADDR = 0x0f). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.29 Value of OTP Bank2 Word0 (Boot Configuration Info.) (HW_OCOTP_BOOT0)

34.7.1.29.1 Offset

Register	Offset
HW_OCOTP_BOOT0	500h

34.7.1.29.2 Function

Shadowed memory mapped access to OTP Bank 2, word 0 (ADDR = 0x10).

34.7.1.29.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.29.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 2, word 0 (ADDR = 0x10). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.30 OTP Controller Boot Configuration Register 3 (HW_OCOTP_BOOT3)

34.7.1.30.1 Offset

Register	Offset
HW_OCOTP_BOOT3	530h

34.7.1.30.2 Function

Shadowed memory mapped access to OTP Bank 2, word 3 (ADDR = 0x13).

34.7.1.30.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.30.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 2, word 3 (ADDR = 0x13). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.31 OTP Controller Boot Configuration Register 4 (HW_OCOTP_BOOT4)

34.7.1.31.1 Offset

Register	Offset
HW_OCOTP_BOOT4	540h

34.7.1.31.2 Function

Shadowed memory mapped access to OTP Bank 2, word 4 (ADDR = 0x14).

34.7.1.31.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.31.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 2, word 4 (ADDR = 0x14). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.32 OTP Controller Boot Configuration Register 5 (HW_OCOTP_BOOT5)

34.7.1.32.1 Offset

Register	Offset
HW_OCOTP_BOOT5	550h

34.7.1.32.2 Function

Shadowed memory mapped access to OTP Bank 2, word 5 (ADDR = 0x15).

34.7.1.32.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.32.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 2, word 5 (ADDR = 0x15). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.33 OTP Controller Boot Configuration Register 6 (HW_OCOTP_BOOT6)

34.7.1.33.1 Offset

Register	Offset
HW_OCOTP_BOOT6	560h

34.7.1.33.2 Function

Shadowed memory mapped access to OTP Bank 2, word 6 (ADDR = 0x16).

34.7.1.33.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.33.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 2, word 6 (ADDR = 0x16). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.34 OTP Controller Boot Configuration Register 7 (HW_OCOTP_BOOT7)

34.7.1.34.1 Offset

Register	Offset
HW_OCOTP_BOOT7	570h

34.7.1.34.2 Function

Shadowed memory mapped access to OTP Bank 2, word 7 (ADDR = 0x17).

34.7.1.34.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.34.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 2, word 7 (ADDR = 0x17). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.35 OTP Controller Analog Register 2 (HW_OCOTP_ANA2)

34.7.1.35.1 Offset

Register	Offset
HW_OCOTP_ANA2	5E0h

34.7.1.35.2 Function

Shadowed memory mapped access to OTP bank 3, word 6 (ADDR = 0x1e).

34.7.1.35.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.35.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP bank 3, word 6 (ADDR = 0x1e). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.36 OTP Controller Analog Register 3 (HW_OCOTP_ANA3)

34.7.1.36.1 Offset

Register	Offset
HW_OCOTP_ANA3	5F0h

34.7.1.36.2 Function

Shadowed memory mapped access to OTP bank 3, word 7 (ADDR = 0x1f).

34.7.1.36.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.36.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP bank 3, word 7 (ADDR = 0x1f). These bits become read-only after corresponding lock fuse bit is set.

34.7.1.37 Value of OTP Bank9 Word1 (HW_OCOTP_PAD_MISC1)

34.7.1.37.1 Offset

Register	Offset
HW_OCOTP_PAD_MISC1	890h

34.7.1.37.2 Function

Shadowed memory mapped access to OTP Bank 9, word 1 (ADDR = 0x49).

34.7.1.37.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.37.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 9, word 1 (ADDR = 0x49).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.38 Value of OTP Bank9 Word3 (HW_OCOTP_PAD_MISC3)

34.7.1.38.1 Offset

Register	Offset
HW_OCOTP_PAD_MISC3	8B0h

34.7.1.38.2 Function

Shadowed memory mapped access to OTP Bank 9, word 3 (ADDR = 0x4b).

34.7.1.38.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.38.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 9, word 3 (ADDR = 0x4b).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.39 Value of OTP Bank24 Word0 (GP6) (HW_OCOTP_GP60)

34.7.1.39.1 Offset

Register	Offset
HW_OCOTP_GP60	1000h

34.7.1.39.2 Function

Shadowed memory mapped access to OTP Bank 24, word 0 (ADDR = 0xc0).

34.7.1.39.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.39.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 0 (ADDR = 0xc0).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.40 Value of OTP Bank24 Word1 (GP6) (HW_OCOTP_GP61)

34.7.1.40.1 Offset

Register	Offset
HW_OCOTP_GP61	1010h

34.7.1.40.2 Function

Shadowed memory mapped access to OTP Bank 24, word 1 (ADDR = 0xC1).

34.7.1.40.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.40.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 1 (ADDR = 0xC1).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.41 Value of OTP Bank24 Word2 (GP6) (HW_OCOTP_GP62)

34.7.1.41.1 Offset

Register	Offset
HW_OCOTP_GP62	1020h

34.7.1.41.2 Function

Shadowed memory mapped access to OTP Bank 24 word 2 (ADDR = 0xc2).

34.7.1.41.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.41.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 2 (ADDR = 0xc2).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.42 Value of OTP Bank24 Word3 (GP6) (HW_OCOTP_GP63)

34.7.1.42.1 Offset

Register	Offset
HW_OCOTP_GP63	1030h

34.7.1.42.2 Function

Shadowed memory mapped access to OTP Bank 24, word 3 (ADDR = 0xc3).

34.7.1.42.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.42.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 3 (ADDR = 0xc3).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.43 Value of OTP Bank24 Word4 (GP6) (HW_OCOTP_GP64)

34.7.1.43.1 Offset

Register	Offset
HW_OCOTP_GP64	1040h

34.7.1.43.2 Function

Shadowed memory mapped access to OTP Bank 24, word 4 (ADDR = 0xc4).

34.7.1.43.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.43.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 4 (ADDR = 0xc4).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.44 Value of OTP Bank24 Word5 (GP6) (HW_OCOTP_GP65)

34.7.1.44.1 Offset

Register	Offset
HW_OCOTP_GP65	1050h

34.7.1.44.2 Function

Shadowed memory mapped access to OTP Bank 24, word 5 (ADDR = 0xc5).

34.7.1.44.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.44.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 5 (ADDR = 0xc5).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.45 Value of OTP Bank24 Word6 (GP6) (HW_OCOTP_GP66)

34.7.1.45.1 Offset

Register	Offset
HW_OCOTP_GP66	1060h

34.7.1.45.2 Function

Shadowed memory mapped access to OTP Bank 24, word 6 (ADDR = 0xc6).

34.7.1.45.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.45.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 6 (ADDR = 0xc6).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.46 Value of OTP Bank24 Word7 (GP6) (HW_OCOTP_GP67)

34.7.1.46.1 Offset

Register	Offset
HW_OCOTP_GP67	1070h

34.7.1.46.2 Function

Shadowed memory mapped access to OTP Bank 24, word 7 (ADDR = 0xc7).

34.7.1.46.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.46.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 24, word 7 (ADDR = 0xc7).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.47 Value of OTP Bank25 Word0 (GP7) (HW_OCOTP_GP70)

34.7.1.47.1 Offset

Register	Offset
HW_OCOTP_GP70	1080h

34.7.1.47.2 Function

Shadowed memory mapped access to OTP Bank 25, word 0 (ADDR = 0xc8).

34.7.1.47.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.47.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 0 (ADDR = 0xc8).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.48 Value of OTP Bank25 Word1 (GP7) (HW_OCOTP_GP71)

34.7.1.48.1 Offset

Register	Offset
HW_OCOTP_GP71	1090h

34.7.1.48.2 Function

Shadowed memory mapped access to OTP Bank 25, word 1 (ADDR = 0xc9).

34.7.1.48.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.48.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 1 (ADDR = 0xc9).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.49 Value of OTP Bank25 Word2 (GP7) (HW_OCOTP_GP72)

34.7.1.49.1 Offset

Register	Offset
HW_OCOTP_GP72	10A0h

34.7.1.49.2 Function

Shadowed memory mapped access to OTP Bank 25, word 2 (ADDR = 0xca).

34.7.1.49.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.49.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 2 (ADDR = 0xca).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.50 Value of OTP Bank25 Word3 (GP7) (HW_OCOTP_GP73)

34.7.1.50.1 Offset

Register	Offset
HW_OCOTP_GP73	10B0h

34.7.1.50.2 Function

Shadowed memory mapped access to OTP Bank 25, word 3 (ADDR = 0xcb).

34.7.1.50.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.50.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 3 (ADDR = 0xcb).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.51 Value of OTP Bank25 Word4 (GP7) (HW_OCOTP_GP74)

34.7.1.51.1 Offset

Register	Offset
HW_OCOTP_GP74	10C0h

34.7.1.51.2 Function

Shadowed memory mapped access to OTP Bank 25, word 4 (ADDR = 0xcc).

34.7.1.51.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.51.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 4 (ADDR = 0xcc).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.52 Value of OTP Bank25 Word5 (GP7) (HW_OCOTP_GP75)

34.7.1.52.1 Offset

Register	Offset
HW_OCOTP_GP75	10D0h

34.7.1.52.2 Function

Shadowed memory mapped access to OTP Bank 25, word 5 (ADDR = 0xcd).

34.7.1.52.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.52.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 5 (ADDR = 0xcd).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.53 Value of OTP Bank25 Word6 (GP7) (HW_OCOTP_GP76)

34.7.1.53.1 Offset

Register	Offset
HW_OCOTP_GP76	10E0h

34.7.1.53.2 Function

Shadowed memory mapped access to OTP Bank 25, word 6 (ADDR = 0xce).

34.7.1.53.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.53.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 6 (ADDR = 0xce).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.54 Value of OTP Bank25 Word7 (GP7) (HW_OCOTP_GP77)

34.7.1.54.1 Offset

Register	Offset
HW_OCOTP_GP77	10F0h

34.7.1.54.2 Function

Shadowed memory mapped access to OTP Bank 25, word 7 (ADDR = 0xcf).

34.7.1.54.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.54.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 25, word 7 (ADDR = 0xcf).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.55 Value of OTP Bank26 Word0 (GP8) (HW_OCOTP_GP80)

34.7.1.55.1 Offset

Register	Offset
HW_OCOTP_GP80	1100h

34.7.1.55.2 Function

Shadowed memory mapped access to OTP Bank 26, word 0 (ADDR = 0xD0).

34.7.1.55.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.55.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 0 (ADDR = 0xD0).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.56 Value of OTP Bank26 Word1 (GP8) (HW_OCOTP_GP81)

34.7.1.56.1 Offset

Register	Offset
HW_OCOTP_GP81	1110h

34.7.1.56.2 Function

Shadowed memory mapped access to OTP Bank 26, word 1 (ADDR = 0xD1).

34.7.1.56.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.56.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 1 (ADDR = 0xD1).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.57 Value of OTP Bank26 Word2 (GP8) (HW_OCOTP_GP82)

34.7.1.57.1 Offset

Register	Offset
HW_OCOTP_GP82	1120h

34.7.1.57.2 Function

Shadowed memory mapped access to OTP Bank 26, word 2 (ADDR = 0xD2).

34.7.1.57.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.57.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 2 (ADDR = 0xD2).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.58 Value of OTP Bank26 Word3 (GP8) (HW_OCOTP_GP83)

34.7.1.58.1 Offset

Register	Offset
HW_OCOTP_GP83	1130h

34.7.1.58.2 Function

Shadowed memory mapped access to OTP Bank 26, word 3 (ADDR = 0xD3).

34.7.1.58.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.58.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 3 (ADDR = 0xD3).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.59 Value of OTP Bank26 Word4 (GP8) (HW_OCOTP_GP84)

34.7.1.59.1 Offset

Register	Offset
HW_OCOTP_GP84	1140h

34.7.1.59.2 Function

Shadowed memory mapped access to OTP Bank 26, word 4 (ADDR = 0xD4).

34.7.1.59.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.59.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 4 (ADDR = 0xD4).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.60 Value of OTP Bank26 Word5 (GP8) (HW_OCOTP_GP85)

34.7.1.60.1 Offset

Register	Offset
HW_OCOTP_GP85	1150h

34.7.1.60.2 Function

Shadowed memory mapped access to OTP Bank 26, word 5 (ADDR = 0xD5).

34.7.1.60.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.60.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 5 (ADDR = 0xD5).

34.7.1.61 Value of OTP Bank26 Word6 (GP8) (HW_OCOTP_GP86)

34.7.1.61.1 Offset

Register	Offset
HW_OCOTP_GP86	1160h

34.7.1.61.2 Function

Shadowed memory mapped access to OTP Bank 26, word 6 (ADDR = 0xD6).

34.7.1.61.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.61.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 6 (ADDR = 0xD6).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.62 Value of OTP Bank26 Word7 (GP8) (HW_OCOTP_GP87)

34.7.1.62.1 Offset

Register	Offset
HW_OCOTP_GP87	1170h

34.7.1.62.2 Function

Shadowed memory mapped access to OTP Bank 26, word 7 (ADDR = 0xD7).

34.7.1.62.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.62.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 26, word 7 (ADDR = 0xD7).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.63 Value of OTP Bank27 Word0 (GP9) (HW_OCOTP_GP90)

34.7.1.63.1 Offset

Register	Offset
HW_OCOTP_GP90	1180h

34.7.1.63.2 Function

Shadowed memory mapped access to OTP Bank 27, word 0 (ADDR = 0xD8).

34.7.1.63.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.63.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 0 (ADDR = 0xD8).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.64 Value of OTP Bank27 Word1 (GP9) (HW_OCOTP_GP91)

34.7.1.64.1 Offset

Register	Offset
HW_OCOTP_GP91	1190h

34.7.1.64.2 Function

Shadowed memory mapped access to OTP Bank 27, word 1 (ADDR = 0xD9).

34.7.1.64.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.64.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 1 (ADDR = 0xD9).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.65 Value of OTP Bank27 Word2 (GP9) (HW_OCOTP_GP92)

34.7.1.65.1 Offset

Register	Offset
HW_OCOTP_GP92	11A0h

34.7.1.65.2 Function

Shadowed memory mapped access to OTP Bank 27, word 2 (ADDR = 0xDA).

34.7.1.65.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.65.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 2 (ADDR = 0xDA).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.66 Value of OTP Bank27 Word3 (GP9) (HW_OCOTP_GP93)

34.7.1.66.1 Offset

Register	Offset
HW_OCOTP_GP93	11B0h

34.7.1.66.2 Function

Shadowed memory mapped access to OTP Bank 27, word 3 (ADDR = 0xDB).

34.7.1.66.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.66.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 3 (ADDR = 0xDB).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.67 Value of OTP Bank27 Word4 (GP9) (HW_OCOTP_GP94)

34.7.1.67.1 Offset

Register	Offset
HW_OCOTP_GP94	11C0h

34.7.1.67.2 Function

Shadowed memory mapped access to OTP Bank 27, word 4 (ADDR = 0xDC).

34.7.1.67.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.67.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 4 (ADDR = 0xDC).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.68 Value of OTP Bank27 Word5 (GP9) (HW_OCOTP_GP95)

34.7.1.68.1 Offset

Register	Offset
HW_OCOTP_GP95	11D0h

34.7.1.68.2 Function

Shadowed memory mapped access to OTP Bank 27, word 5 (ADDR = 0xDD).

34.7.1.68.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.68.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 5 (ADDR = 0xDD).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.69 Value of OTP Bank27 Word6 (GP9) (HW_OCOTP_GP96)

34.7.1.69.1 Offset

Register	Offset
HW_OCOTP_GP96	11E0h

34.7.1.69.2 Function

Shadowed memory mapped access to OTP Bank 27, word 6 (ADDR = 0xDE).

34.7.1.69.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.69.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 6 (ADDR = 0xDE).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.70 Value of OTP Bank27 Word7 (GP9) (HW_OCOTP_GP97)

34.7.1.70.1 Offset

Register	Offset
HW_OCOTP_GP97	11F0h

34.7.1.70.2 Function

Shadowed memory mapped access to OTP Bank 27, word 7 (ADDR = 0xDF).

34.7.1.70.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.70.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 27, word 7 (ADDR = 0xDF).These bits become read-only after corresponding lock fuse bit is set.

34.7.1.71 Value of OTP Bank28 Word7 (HW_OCOTP_GP107)

34.7.1.71.1 Offset

Register	Offset
HW_OCOTP_GP107	1270h

34.7.1.71.2 Function

Shadowed memory mapped access to OTP Bank 28, word 7 (ADDR = 0xE7).

34.7.1.71.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BITS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

34.7.1.71.4 Fields

Field	Function
31-0 BITS	BITS Reflects value of OTP Bank 28, word 7 (ADDR = 0xE7).These bits become read-only after corresponding lock fuse bit is set.

Chapter 35
Reset and Boot

35.1 Reset

Resetting the device provides a way to start processing from a known set of initial conditions. System reset begins with the on-chip regulators in full regulation and system clocking generation from the internal reference FIRC clock. The CMC[1] and RMC, [1] modules control the reset sequence of the device. There are various reset sources on this device. They can be categorized into three groups:

- Power-On Reset (POR)
- System Reset
- Local Reset

Reset types and their sources are described in more detail in the following sections.

i.MX 7ULP is designed in a way that a POR or System Reset event on the A7 power domain only resets that domain, while a POR or System Reset event on the M4 power domain will reset both power domains.

NOTE

A7 Reset is also mapped as interrupt on M4 domain, so in case A7 Reset must reset M4, it could be done via M4 ISR. Additionally, A7 Reset can automatically affect M4 domain when the A7_TO_M4_RESET_EN fuse is blown.

NOTE

If a reset that only affects the Application domain (that is, does not propagate in any way to the Real Time domain) takes place and the Real Time domain's bus clock is gated (Real Time domain in PSTOP1 or more restrictive modes), then that

1. MSMC, CMC, and RMC are 3 separate but tightly coupled blocks. The memory map documented in MSMC chapter encompasses features related to all 3 blocks, being actually implemented into the CMC block alone. Therefore, there might be some references to CMC and RMC blocks throughout this document.

(Application domain) reset sequence does not conclude. In order to avoid such a scenario, the following two solutions are available:

- Enable the Application domain's reset to propagate to the Real Time domain (by blowing the A7_TO_M4_RESET_EN fuse), or
- Configure Real Time domain to wake up upon an Application domain reset (by setting MU_A CR[RAIE] and properly configuring the corresponding IRQ). Real Time domain can re-enter the mode it was before being woken up by any of these procedures once the Application domain's reset sequence has been concluded (MU_A SR[RS] == 0), if necessary.

35.1.1 Power-On Reset

A Power-On Reset (POR) will reset the entire domain(s) affected by that reset source. All data is lost. A POR event will be initiated by one of the following conditions:

- When power is initially applied to the device.
- When supply voltage has risen above the Low Voltage Detect (LVD) threshold, after the LVD condition was detected. There are four LVD sources that could trigger POR reset:
  - A7 LDO LVD monitor: Trigger POR reset to A7 domain
  - M4 LDO LVD monitor: Trigger POR reset to M4 and A7 domains
  - M4 MEM LDO LVD monitor: Trigger POR reset to M4 and A7 domains
  - PMC POR monitor: Trigger POR reset to M4 and A7 domains

  Following a POR event, the POR and LVD flag bit will be set in the Reset Status register.

35.1.2 System Reset

A system reset can be initiated by one of the following sources:

Table 35-1. M4 system reset sources

Reset source	Reset name	Comments
System_Reset_A7	SYS_RESET_A7	ORed version of all system reset sources from A7. Only enabled when A7_TO_M4_RESET_EN fuse is blown.
Power-On-Reset	POR_M4	—

Table continues on the next page...

Table 35-1. M4 system reset sources (continued)

Reset source	Reset name	Comments
Reset pin (External)	RESET_M4_b	—
Software JTAG	SW_JTAG_RST	The JTAG module generates a system reset when certain IR codes are selected. This functional reset is asserted when EXTEST, HIGHZ and CLAMP instructions are active. The reset source from the JTAG module is released when any other IR code is selected.
WDOG (M4)	WDOG_M4_RST	—
Loss of Clock (LOC)	LOC_EXT_XTL	Loss of Clock on external Crystal
Core Software Reset	SW_RST_M4	Software reset initiated by CM4
CPU Exception	CPU_EXP_M4	—
Low Leakage Wakeup Unit (LLWU)	WAKEUP_M4	Reset due to wakeup from CM4 Power down modes
Stop Mode Ack Error	STOP_ACK_M4	—

Table 35-2. A7 system sources

Reset source	Reset name	Comments
Power-On-Reset	POR_A7	—
Reset pin (External)	RESET_A7_b	—
System_Reset_M4	SYS_RESET_M4	ORed version of all system reset sources from M4
WDOG (A7)	WDOG_A7_RST	—
Core Software Reset	SW_RST_A7	Software reset initiated by CA7
CPU Exception	CPU_EXP_A7	—
Secure WDOG	WDOG_RST	Optional reset
Stop Mode Ack Error	STOP_ACK_A7	—
Power Down Wakeup	VLLS_WAKEUP_A7	Wake-up from VLLS/Power down mode via MU

NOTE

Even though all source clock control logic resides in M4, there could be cases where M4 is running from FIRC clock while A7 is running from XTAL, so A7 must know LOC for MHz Crystal.

The reset pin filter supports filtering from the IP bus clock. In the low power modes when the IP bus clock is not present, this logic either switches to bypass operation or has continued filtering operation depending on the filtering mode selected.

The RMC[2] provides the option for delaying watchdog reset request from either watchdog module. The reset request output from watchdog module will start delay timer. The timer countdown period can be configured during the system start up for a fixed set of options. The default value of the timer countdown period is 0. Upon receiving a watchdog reset request, the RMC will generate a watchdog interrupt to both CPUs. Software cannot cancel the delay timer after it started the countdown. When delay timer reaches 0, the watchdog reset output resets the system.

A System Reset event triggered by an A7 source would reset that power domain. While a System Reset event triggered by an M4 source would reset both domains except for certain components that are not affected by a specific System Reset source.

35.1.3 Local Resets

A variety of resets are generated to reset certain modules or logics of the chip. There are separate resets for each power domain. They are asserted the same way by their respective reset sources with the exception of VLLS wakeup.

Table 35-3. Local Resets

Local Reset	Assertion Sources	Resets What?
VBAT POR	Asserts on a VBAT POR reset source.	Resets only the modules within the VBAT power domain. These VBAT power domain modules are not affected by the other resets.
POR Only	Asserts on the POR reset source only.	Resets the PMC and System Register File.
Chip POR Not VLLS	Asserts on POR and LVD reset sources.	Resets parts of the MSMC and SIM, and also resets some low power peripherals (like LPTMR).
Chip POR	Asserts on POR, LVD, and VLLS Wakeup reset sources.	Resets the Reset Pin Filter registers and parts of the SIM and SCG.
Chip Reset Not VLLS	Asserts on all reset sources except a VLLS Wakeup that does not occur via the RESET_b pin.	Resets parts of the MSMC, LLWU, and other modules that remain powered on during VLLS mode.
Chip	Asserts on all reset sources.	Resets the remaining modules that are not reset by the other MCU reset types.

2. MSMC, CMC, and RMC are 3 separate but tightly coupled blocks. The memory map documented in MSMC chapter encompasses features related to all 3 blocks, being actually implemented into the CMC block alone. Therefore, there might be some references to CMC and RMC blocks throughout this document.

35.2 System boot

35.2.1 Overview

There are two ROMs on i.MX 7ULP: one running on Cortex-A7 core and another on Cortex-M4 core.

The A7 and M4 processors on i.MX 7ULP start booting from a power on reset. The M4 core always starts first and holds off the A7 core until the M4 core determines the boot modes from BOOT_MODE[1:0] pin settings and boot configurations from boot configuration pin/fuses settings. The boot ROMs support features such as: selectable boot sources, setting device configurations and secure boot. The boot ROMs are responsible for reading the settings of boot pins (boot mode pins and boot configuration pins) and OTP values to determine the behavior of the boot flow.

The main features of the ROMs include:

- Support for booting from various boot devices
- Serial downloader support (USB OTG, A7 ROM only)
- Device configuration data (DCD) and plugin
- Digital signature and encryption based High Assurance Boot (HAB)
- Wake-up from low power modes

The boot ROM supports the following boot devices:

- QuadSPI flash for M4 ROM
- SD/eMMC for A7 ROM.

Both boot ROMs use the state of BOOT_MODE and eFUSEs to determine the boot device. For development purposes, eFUSEs used to determine the boot device may be overridden by using GPIO pin inputs (boot configuration pins).

The device configuration data (DCD) feature allows boot ROM code to obtain SOC configuration data from an external Program Image residing on the boot device. As an example, DCD can be used to program the DDR controller for optimal settings improving the boot performance. DCD is restricted to memory areas and peripheral addresses that are considered essential for boot purposes (see Write Data Command).

A key feature of the boot ROM is the ability to perform a secure boot or High Assurance Boot (HAB). This is supported by the HAB security library which is a subcomponent of the ROM code. HAB uses a combination of hardware and software together with a Public Key Infrastructure (PKI) protocol to protect the system from executing unauthorized programs. Before the HAB allows a user's image to execute, the image must be signed. The signing process is done during the image build process by the private key holder and

the signatures are then included as part of the final Program Image. If configured to do so, the ROM verifies the signatures using the public keys included in the Program Image. In addition to supporting digital signature verification to authenticate Program Images, Encrypted boot is also supported. Encrypted boot can be used to prevent cloning of the Program Image directly off the boot device. A secure boot with HAB can be performed on all boot devices supported on the chip in addition to the Serial Downloader. The HAB library in the boot ROM also provides API functions, allowing additional boot chain components (bootloaders) to extend the secure boot chain. The out-of-fab setting for SEC_CONFIG is the Open configuration in which the ROM/HAB performs image authentication, but all authentication errors are ignored and the image is still allowed to execute.

NOTE

For details on security features, see i.MX 7ULP Security Reference Manual

35.2.2 High level description of Boot: Boot mode and boot flow

35.2.2.1 Boot modes

The boot sequence on i.MX 7ULP is determined and controlled through the following mechanisms:

- **BOOT_MODE [1:0]**: Type of boot is controlled by the boot mode pins "BOOT_MODE [1:0]" sampled at exit of reset (stored in Boot Mode Register in CMC [3] module). The available boot modes are: Boot From Fuses, serial boot via USB, and Internal Boot. See the table below for settings.

Table 35-4. i.MX 7ULP boot modes

BOOT_MODE[1:0]	Boot Mode	Boot details
00	Boot from fuses	Boot Configuration is loaded from fuse instead of boot configuration pins. If BT_FUSE_SEL=0, boot flow will jump to Serial Downloader.

Table continues on the next page...

3. MSMC, CMC, and RMC are 3 separate but tightly coupled blocks. The memory map documented in MSMC chapter encompasses features related to all 3 blocks, being actually implemented into the CMC block alone. Therefore, there might be some references to CMC and RMC blocks throughout this document.

Table 35-4. i.MX 7ULP boot modes (continued)

BOOT_MODE[1:0]	Boot Mode	Boot details
01	Serial Downloader	A7 core will go into USB serial download mode (by SDP protocol), as a high-speed, HID device. M4 core will go into an endless loop.
10	Internal Boot Mode	• BT_FUSE_SEL=1 : Additional Boot configuration will be loaded from fuses. • BT_FUSE_SEL=0 : Additional Boot configuration will be loaded from GPIO. This mode is intended for development only.
11	Reserved	—

- Additional boot configuration is obtained via either programmable e-fuses or by pins value sampled during reset. The boot pins are sampled one CM4 bus clock cycle before the RESET0_b/RESET1_b signal is de-asserted. The selection between the two options, is done based on the value of BT_FUSE_SEL fuse, as follows.
  - In case BT_FUSE_SEL is blown, all boot options are configured by efuses. Boot ROM code software may read the values from the Boot Mode Register, or from the e-fuses, via the OCOTP_CTRL module.
  - In case BT_FUSE_SEL is left unblown and BOOT_CFG[1:0]=01 or 10, the various boot options are determined by sample of dedicated pins at the exit of reset. Every e-fuse option is overridden with a dedicated pin(s), such that same functionality is available for both boot options. See the fuse map attached with this reference manual for boot option pins. For this case, regardless of fuse values, Boot ROM code must read the options' values from Boot Mode Register of the CMC[3] module.

NOTE

Boot pins, including Boot Mode pins and Boot Config pins, are sampled and stored into CMC Mode register on every cold reset (POR, VLLS exiting, etc). After these input pins are sampled, their subsequent state does not affect the contents of the CMC Mode register. Specifically, Boot Mode pins and M4 ROM Boot Config pins (BT0_CFG0-15) are sampled on every M4 domain reset and the A7 ROM Boot Config pins (BT1_CFG0-15) are sampled on every A7 domain reset.

35.2.2.1.1 Boot From Fuses Mode (BOOT_MODE[1:0] = 00b)

A value of 00b in the BOOT_MODE[1:0] register selects the Boot From Fuses mode.

This mode is similar to the Internal Boot mode described in Internal Boot Mode (BOOT_MODE[1:0] = 0b10) with one difference. In this mode the GPIO boot override pins are ignored. The boot ROM code uses the boot eFUSE settings only. This mode also supports a secure boot using HAB.

If set to Boot From Fuses, the boot flow is controlled by the BT_FUSE_SEL eFUSE value.

- If BT_FUSE_SEL = 0, indicating that the boot device (for example, Flash, SD/MMC) has not yet been programmed, the boot flow jumps directly to the Serial Downloader.
- If BT_FUSE_SEL = 1, the normal boot flow is followed, where the ROM attempts to boot from the selected boot device.

The first time a board is used, the default eFUSEs may be configured incorrectly for the hardware on the platform, or most likely, boot fuses not been blown. In such a case, the Boot ROM code may try to boot from a device that does not exist. This may cause an electrical/logic violation on some pads. Using Boot From Fuses mode addresses this problem.

Setting BT_FUSE_SEL=0 in Boot From Fuse Mode forces the ROM code to jump directly to the Serial Downloader. This allows a bootloader to be downloaded which can then provision the boot device with a Program Image and blow the BT_FUSE_SEL and the other boot configuration eFUSEs. After reset, the Boot ROM code determines that BT_FUSE_SEL is blown (BT_FUSE_SEL = 1) and the ROM code performs internal boot according to the new eFUSE settings. This allows a user to set BOOT_MODE[1:0]=00b on a production device and burn fuses on the same device (by forcing entry to the Serial Downloader), without changing the value of BOOT_MODE[1:0] or pullups/pulldowns on the BOOT_MODE pins.

35.2.2.1.2 Serial downloader

The Serial Downloader provides a means to download a Program Image to the chip over USB connection, usually for the manufacturing scenario.

In this mode, A7 ROM programs WDOG1 for a time-out specified by fuse WDOG Timeout Select (See attached fuse map for details) if WDOG_ENABLE eFuse is 1 and continuously polls for USB connection. If no activity is found on USBOTG2 and the watchdog timer expires, the A7 core is reset. If valid SDP command be received by A7 ROM though USB, A7 ROM will execute them. Note that some SDP commands can be disabled by fuse (SDP_READ_DISABLE), some may need authentication if HAB closed (for SDP_JUMP), and USB SDP feature can be totally disabled by fuse SDP_DISABLE.

NOTE

After being loaded, the downloaded image is responsible to manage the watchdog resets properly if the watchdog been ever enabled.

NOTE

Serial Downloader is available on A7 ROM only, and available on USBOTG2 only.

The following figure shows the USB serial download boot flow.

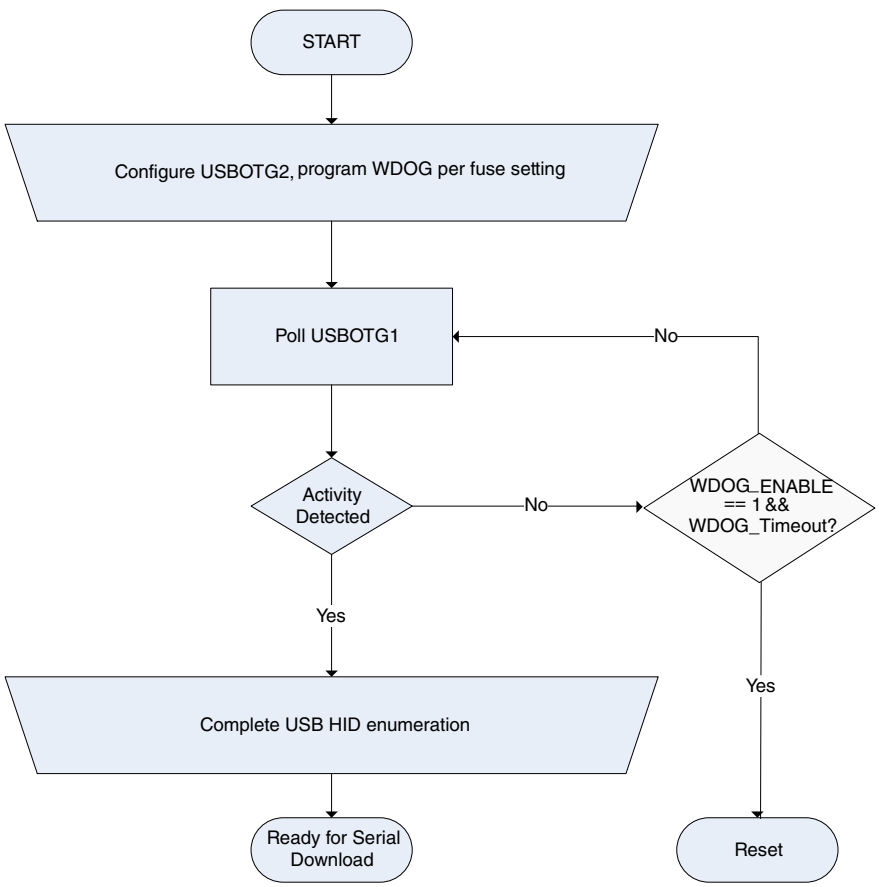


Figure 35-1. Serial download sequence

35.2.2.1.3 Internal Boot Mode (BOOT_MODE[1:0] = 0b10)

A value of 0b10 in the BOOT_MODE[1:0] register selects the internal boot mode. In this mode, the processor continues to execute boot code from the internal boot ROM. The boot code performs hardware initialization, loads the Program Image from the chosen boot device, performs image validation using the HAB library (see Boot security

settings), and then jumps to an address derived from the Program Image. If any error occurs during internal boot, the boot code jumps to the Serial Downloader (see Serial downloader). A secure boot using the HAB is possible in all the three boot modes.

When set to internal boot, the boot flow may be controlled by a combination of eFUSE settings with an option of overriding the fuse settings using General Purpose I/O (GPIO) pins. The GPIO Boot Select FUSE (BT_FUSE_SEL) determines whether the ROM uses GPIO pins for a select number of configuration parameters or eFUSEs in this mode. See Table 35-6 for more details.

- If BT_FUSE_SEL = 1, all boot options are controlled by the eFUSEs described in Table 35-6
- If BT_FUSE_SEL = 0, specific boot configuration parameters may be set using GPIO pins rather than eFUSEs. The fuses that can be overridden when in this mode are indicated in the GPIO column of Table 35-6. See Table 35-8 for GPIO overrides.

The use of GPIO overrides is intended for development since these pads are used for other purposes in deployed products. NXP recommends controlling the boot configuration by eFUSEs in deployed products and reserving the use of the GPIO mode for development and testing purposes only.

35.2.2.2 Boot flow

35.2.2.2.1 M4 boot flow

The M4 processor starts booting from the internal ROM and initializes the QSPI controller for booting from serial flash memory. Any errors occurring during this boot procedure will lead to serial download via the designated interface.

35.2.2.2.2 A7 boot flow

The A7 processor starts booting from the internal ROM and initializes the memory controller for the selected memory device. Any errors occurring during this boot procedure will lead to serial download via the designated interface.

35.2.2.2.3 Boot flow configurations

Depending on the setting of BT0_CFG [1:0], there are three different boot flows on i.MX 7ULP, listed in the table below.

Table 35-5. Boot flow configuration

BT0_CFG[1:0]	Boot Flow	Description
00	Single Memory Boot	• M4 ROM kicks A7 (lets A7 core start to run) and waits until M4 image been provisioned by A7 image. • A7 ROM loads A7 image from eMMC/SD, then A7 image loads M4 image and sets M4 entry point.
x1	Low Power Boot	M4 ROM will not kick A7 domain in this mode.
10	Dual Memory Boot	The M4 loads from the serial flash through the QSPI interface, and the A7 starts (Kicked by M4 ROM) loading from the eMMC/SDmemory through the uSDHC0/1 interface

35.2.2.2.3.1 Dual memory boot

In this boot mode, the two processors on i.MX 7ULP boot independently of each other, except for any software-imposed handshaking inserted explicitly to synchronize the process. The M4 loads from the serial flash through the QSPI interface, and the A7 starts loading from the eMMC memory through the uSDHC0 interface.

Under normal operation, each of the two processors boots using the High-Assurance Boot (HAB) contained within each of their boot ROMs. The HAB verifies the authenticity of the code in external memory, then allows it to execute. i.MX 7ULP also supports an HAB-bypass mode in which the processors can boot directly from external memory.

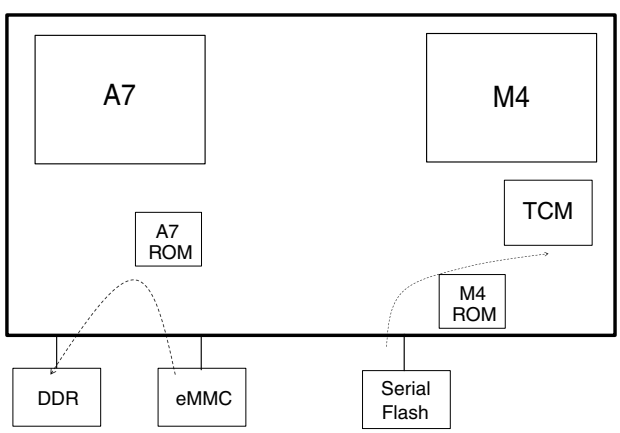


Figure 35-2. Dual Memory Boot

35.2.2.2.3.2 Single memory boot

The A7 core loads the M4 core in this mode.

1. The A7 will start loading from the eMMC memory.
2. The A7 loads the M4 internal memory and starts the M4 once it is ready.
3. If HAB is enabled, the A7 ROM will verify the authenticity of M4 code and its code.

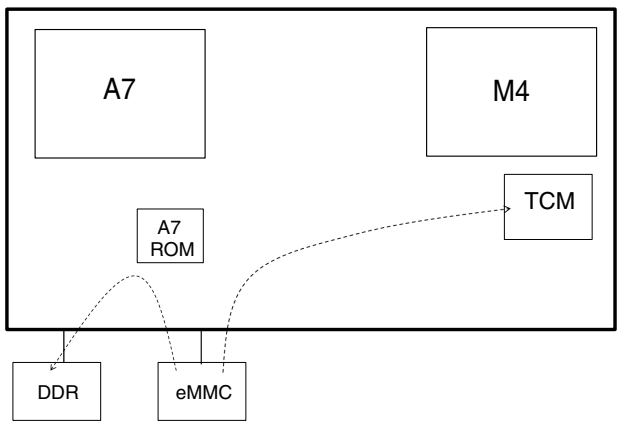


Figure 35-3. Single Memory Boot

35.2.2.2.3.3 Low power boot

In Low Power Boot boot mode, only the M4 core will boot up and run on the FIRC clock frequency to conserve battery energy. The boot sequence of the M4 in Lower Power Boot is similar to its normal boot sequence, except the processor, bus, and peripherals operate on the FIRC clock or a divided version of the FIRC clock.

35.2.2.3 Boot security settings

Following are the boot security settings specified by SEC_CONFIG[1:0]/ FIELD_RETURN fuses and provisioning the chip life cycle. For more details on security features and security fuses, see the i.MX 7ULP Security Reference Manual.

Available boot security settings include:

- **Closed**: This level is intended for use with shipping secure products. All HAB functions are executed and security hardware is initialized (the Security Controller, or SNVS, enters Secure state), DCD is processed if present, and the program image is authenticated by HAB prior to its execution. All detected errors will be logged, and

the boot flow aborted with control passing to the serial downloader (For M4 ROM, control passing to endless loop). At this level, execution does not leave the internal ROM unless the target executable image has been authenticated.

- **Open**: This level is intended for use in non-secure products or during the development phases of a secure product. All HAB functions are executed as for a closed device. Security hardware is initialized (except the SNVS is left in Non-Secure state), DCD is processed if present, and the program image is authenticated by HAB prior to its execution. All detected errors will be logged, but have no influence on the boot flow, which continues as if the errors did not occur. This configuration is useful for secure product development, since the Program Image will run even if the authentication data is missing or incorrect, and the error log can be examined to determine the cause of authentication failure.
- Field Return: This level is intended for parts returned from shipped products.

NOTE

If the DIR_BT_DIS eFuse is not blown, authentication may be bypassed. In this case the system is not secure.

35.2.3 Boot configuration and eFuse

This section describes the external inputs that control the behavior of the Boot ROM code. This includes boot device selection, boot device configuration (SD bus width, speed, etc.), and so on. In general, the source for this configuration comes from eFUSEs embedded inside the chip. However, certain configuration parameters can be sourced from GPIO pins allowing further flexibility during the development process.

35.2.3.1 Boot eFuse description

All Boot eFuses are included in fuse map attached with this Reference Manual or i.MX 7ULP Security Reference Manual. Here is a comprehensive list of the configuration parameters that the ROM uses.

Table 35-6. Configuration parameters or eFUSEs used by ROM1

FUSE	Definition	GPIO	Shipped Value	Settings
A7 LDO Enable	Once blown, M4 ROM will enable A7 LDO by setting PMC_0_CTRL[LDOEN] bit when kicking A7. This fuse value shall be aligned with power design of the board.	N	0	0—Disable 1—Enable

Table continues on the next page...

Table 35-6. Configuration parameters or eFUSEs used by ROM1 (continued)

M4_LOW_FREQ_BT		N	0	0—Normal frequency 1—Half frequency
A7_LOW_FREQ_BT		N	0	0—Normal frequency 1—Half frequency
A7 D Cache Disable	If blown, A7 ROM will not enable D-cache/MMU when authenticating image.	N	0	0—Enable 1—Disable
M4 D Cache Disable	If blown, M4 ROM will not enable D-cache when authenticating image.	N	0	0—Enable 1—Disable
WDOG Timeout Select	To select WDOG timeout value, valid when WDOG_ENABLE been blown.	N	0	00—32s 01—16s 10—8s 11—4s Others—0.5s
A7 WDOG_ENABLE		N	0	0—Disabled 1—Enabled
M4 WDOG_ENABLE		N	0	0—Disabled 1—Enabled
A7 I-Cache Disable		N	0	0—Enable 1—Disable
M4 I-Cache Disable		N	0	0—Enable 1—Disable
LP Boot		Y	0	0—No Low Power Boot 1—Boot from M4 with A7 on demand
Dual Boot		Y	0	0—Boot from eMMC 1—Boot from A7/eMMC and M4/QSPI
Infinite-Loop (Debug use only)	This is for debug purpose only, and can be disabled if DIR_BT_DIS been blown.	Y	0	0—Disable 1—Enable
M4 boot interface		Y	0	0—QSPI Others—reserved for future use
External OSC Freq Selection	**NOTE:** This setting must be aligned with the frequency of external OSC on the board, otherwise the BootROM may not work properly.	Y	0	00—24 MHz 01—30 MHz 10—19.2 MHz 11—26 MHz
uSDHC device type		Y	0	0—eMMC 1—SD

Table continues on the next page...

Table 35-6. Configuration parameters or eFUSEs used by ROM1 (continued)

A7 boot interface		Y	0	000—uSDHC0 001—uSDHC1 Others—reserved
SEC_CONFIG[0]	Security Configuration as defined in Boot security settings	N	1	SEC_CONFIG[1:0] mapping: 00—Reserved 01—Open (allows any program image, even if authentication fails) 1x—Closed (Program image executes only if authenticated)
SEC_CONFIG[1]		N	0	
DIR_BT_DIS	Disables the NXP reserved modes	N/A	0	0—The reserved NXP modes are enabled. 1—The reserved NXP modes are disabled. This fuse must be blown for normal operation.
BT_FUSE_SEL	In internal Boot mode • BOOT_MODE[1:0] = 10, the BT_FUSE_SEL fuse determines whether the boot settings indicated by a Yes in the GPIO column are controlled by GPIO pins or eFUSE settings in the On-Chip OTP Controller(OCOTP). In Boot From Fuse mode • BOOT_MODE[1:0] = 00, BT_FUSE_SEL fuse indicates whether bit configuration eFuses have been programmed .	N	0	If BOOT_MODE[1:0] = 0b10 • 0—Bits of SBMR are overridden by GPIO pins. • 1—Specific bits of CMC0_MODE[15:0] and CMC1_MODE[15:0] are controlled by eFUSE settings If BOOT_MODE[1:0] = 0b00 • 0—BOOT configuration eFuses are not yet programmed. Boot flow jumps to downloader • 1—BOOT configuration eFuses have been programmed. Regular boot flow is performed. .
SOSC Source Selection	To select the source of SOSC	N	0	0—Crystal 1—External Clock generator

1. See the attached fusemap for more details.

NOTE

See Boot Device section for the detail of fuses relevant to the device.

The following table presents the recommended settings for the lock fuses.

Table 35-7. Recommended settings for lock fuses

Fuse name	Number of fuses	Fuse function	Settings
BOOT_CFG_LOCK	2	Perform lock on BOOT related fuses	00—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) x1—Lock (Overwrite Protect) 1x—Lock (Write Protect)
GPR9_LOCK	1	Lock for General Purpose fuse register #1 (GP9)	0—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) 1—Lock (Write Protect, Overwrite Protect)
GPR8_LOCK	1	Lock for General Purpose fuse register #1 (GP8)	0—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) 1—Lock (Write Protect, Overwrite Protect)
GPR7_LOCK	1	Lock for General Purpose fuse register #1 (GP7)	0—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) 1—Lock (Write Protect, Overwrite Protect)
GPR6_LOCK	1	Lock for General Purpose fuse register #1 (GP6)	0—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) 1—Lock (Write Protect, Overwrite Protect)
PMU_LOCK	2	Lock for Banks 10 and 28 fuses	00—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) x1—Lock (Write Protect) 1x—Lock (Overwrite Protect) **NOTE:** PMU_LOCK is not locked by NXP and must be locked by the user if required.
ANALOG_LOCK	2	Lock for USB_VID, USB0_PID, USB1_PID fuses	00—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) x1—Lock (Write Protect)

Table continues on the next page...

Table 35-7. Recommended settings for lock fuses (continued)

Fuse name	Number of fuses	Fuse function	Settings
			1x—Lock (Overwrite Protect)
PAD_MISC_LOCK	2	Lock for PAD_SETTINGS, PT*_RANGE_CONTROL_RST_VALUE fuses	00—Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded IIM register) x1—Lock (Write Protect) 1x—Lock (Overwrite Protect)

35.2.3.2 Boot configuration

Each of the CM4 and CA7 domains of i.MX 7ULP includes 16 boot configuration pins that can be used to load boot configuration during "*Boot from Pins*" mode (BOOT_MODE [1:0] = "10")

Table 35-8. GPIO override assignments

M4 Boot Configuration Pins		**A7 Boot Configuration Pins**	
GPIO	*Boot Configuration*	*GPIO*	*Boot Configuration*
PTA0	BT0_CFG0	PTF0	BT1_CFG0
PTA1	BT0_CFG1	PTF1	BT1_CFG1
PTA2	BT0_CFG2	PTF2	BT1_CFG2
PTA3	BT0_CFG3	PTF3	BT1_CFG3
PTA4	BT0_CFG4	PTF4	BT1_CFG4
PTA5	BT0_CFG5	PTF5	BT1_CFG5
PTA6	BT0_CFG6	PTF6	BT1_CFG6
PTA7	BT0_CFG7	PTF7	BT1_CFG7
PTA8	BT0_CFG8	PTF8	BT1_CFG8
PTA9	BT0_CFG9	PTF9	BT1_CFG9
PTA10	BT0_CFG10	PTF10	BT1_CFG10
PTA11	BT0_CFG11	PTF11	BT1_CFG11
PTA12	BT0_CFG12	PTF12	BT1_CFG12
PTA13	BT0_CFG13	PTF13	BT1_CFG13
PTA14	BT0_CFG14	PTF14	BT1_CFG14
PTA15	BT0_CFG15	PTF15	BT1_CFG15
SEC_CONFIG[0]	BT0_CFG16	SEC_CONFIG[0]	BT1_CFG16
SEC_CONFIG[1]	BT0_CFG17	SEC_CONFIG[1]	BT1_CFG17
Reserved	BT0_CFG18	Reserved	BT1_CFG18
DIR_BT_DIS	BT0_CFG19	DIR_BT_DIS	BT1_CFG19

Table continues on the next page...

Table 35-8. GPIO override assignments (continued)

BT_FUSE_SEL	BT0_CFG20	BT_FUSE_SEL	BT1_CFG20
Reserved	BT0_CFG21	Reserved	BT1_CFG21
Reserved	BT0_CFG22	Reserved	BT1_CFG22
Reserved	BT0_CFG23	Reserved	BT1_CFG23
Reserved	BT0_CFG24	Reserved	BT1_CFG24
Reserved	BT0_CFG25	Reserved	BT1_CFG25
Reserved	BT0_CFG26	Reserved	BT1_CFG26
Reserved	BT0_CFG27	Reserved	BT1_CFG27
Reserved	BT0_CFG28	Reserved	BT1_CFG28
Reserved	BT0_CFG29	Reserved	BT1_CFG29
BOOT_MODE[0]	BT0_CFG30	BOOT_MODE[0]	BT1_CFG30
BOOT_MODE[1]	BT0_CFG31	BOOT_MODE[1]	BT1_CFG31

Each individual domain includes 32 boot configurations bits (BT0[1]_CFG[31:0]. When the “boot from Pins” is selected BT0[1]_CFG[15:0] always default to taking the values from the pins while rest of the boot configuration (BT0[1]_CFG[31:16]) always come from fuses irrespective of state of BOOT_MODE[1:0].

BT0_CFG[15:0] is mapped to PTA[15:0] that is part of CM4 power domain. BT1_CFG [15:0] is mapped to PTF[15:0] that is part of CA7 power domain. This ensures that when CA7 domain is powered gated, M4 can still boot and load boot configuration from its individual pins(PTA[15:0]).

NOTE

BOOT_MODE [1:0] pins apply to both CM4/CA7 domain, however the value of BOOT_MODE [1:0] should always be consistent between both the domains. For the cases where CA7 exits from power down mode, internal logic will ignore the values on the BOOT_MODE [1:0] pins and would rather latch the values from CM4 domain. Specific to Boot_mode[1:0] = “10” individual boot configuration (BT1[15:0]) for CA7 will still come from the pins(PTF[14:0]).

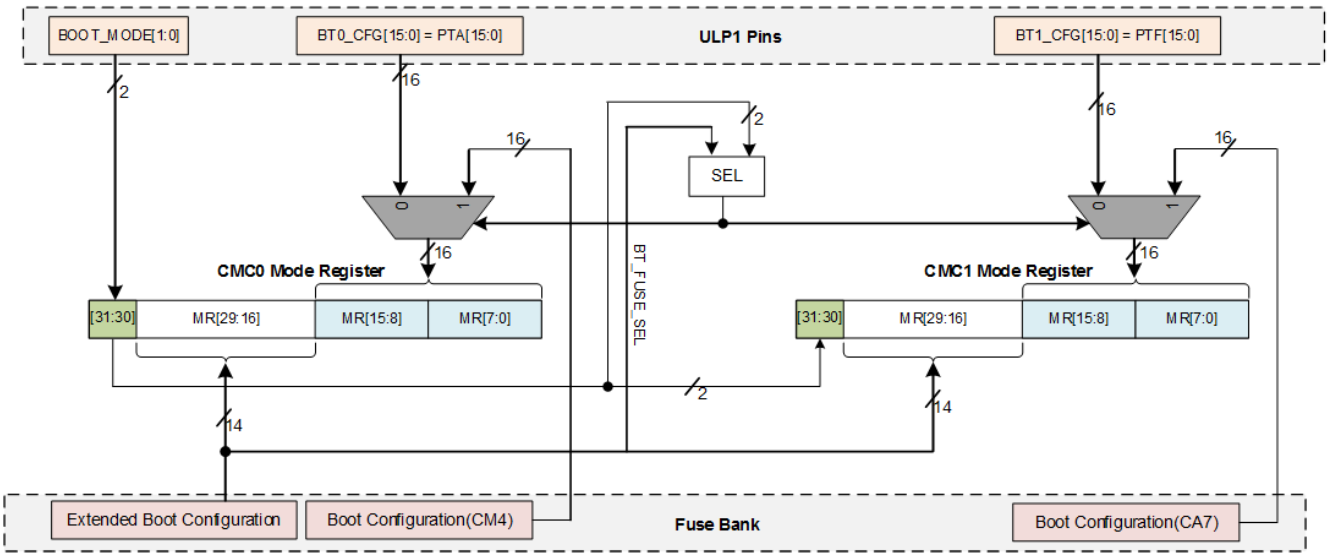


Figure 35-4. Boot configuration

35.2.4 Platform initialization in ROM

This section describes the details on the ROM and provides initialization details. This includes details on:

- The ROM Memory Map
- The RAM Memory Map
- On-chip blocks that the ROM should make use of or change POR register default values
- Clock options
- MMU/cache/LMEM
- Exception handling and interrupt handling

35.2.4.1 Internal ROM/RAM memory map

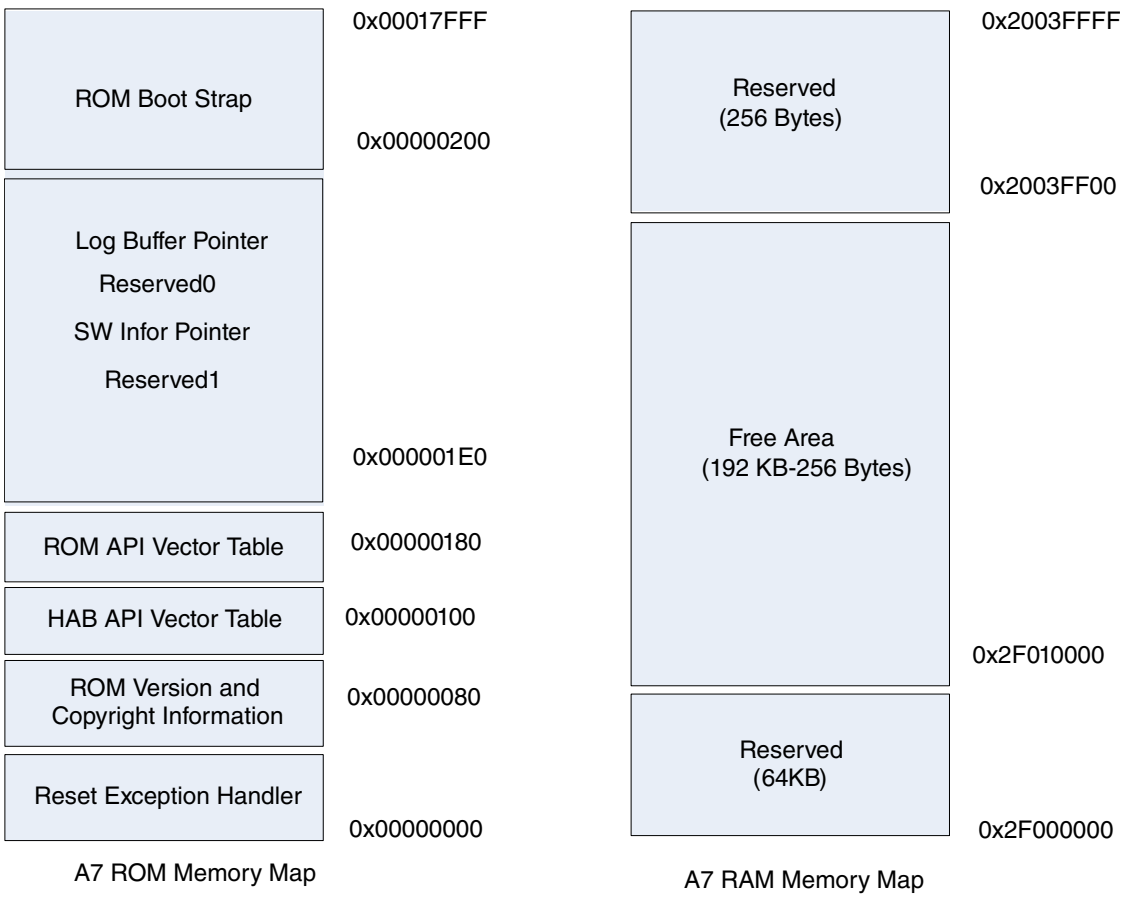


Figure 35-5. A7 ROM/RAM memory layout

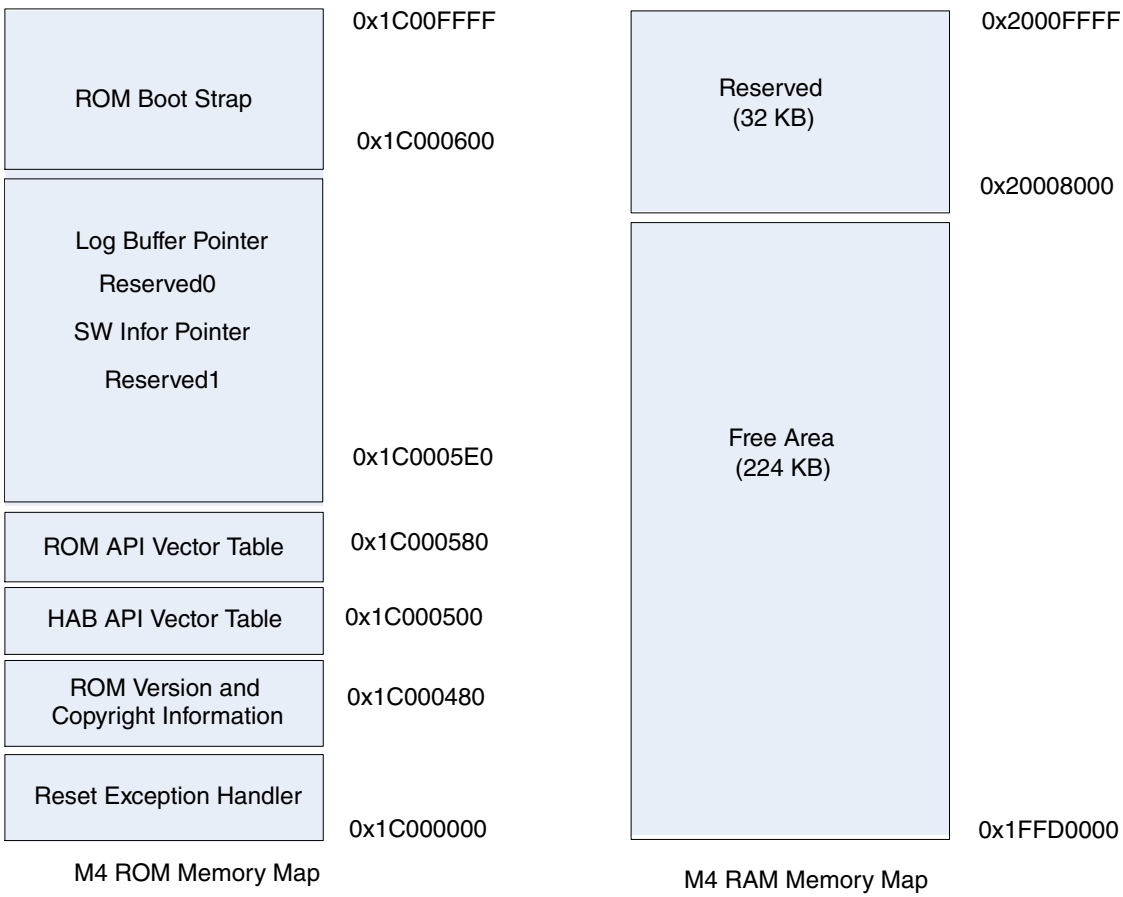


Figure 35-6. M4 ROM/RAM memory layout

NOTE

For both M4 and A7, the entire RAM region (including the reserved RAM) can be freely used after boot in case ROM/HAB APIs are not being called in a post boot stage. Special attention need to be paid in M4 suspend/resume case: If M4 application uses reserved RAM, it needs to back up the reserved RAM on suspend and restore them on resume, because ROM will overwrite the reserved RAM on resume.

35.2.4.2 Boot block activation

The boot ROM affects a number of different hardware blocks which are activated and play a vital role in the boot flow. The ROM configures and uses the following blocks during the boot process. Note that the blocks actually used depend on the boot mode and boot device selection.

Following is the list of blocks used by M4 ROM:

- IOMUXC - I/O Multiplexer Control
- LMEM - Local Memory Controller
- QuadSPI - QuadSPI Flash controller
- OTFAD – On-the-fly AES Decryptor
- WDOG-0 - Watchdog Timer
- CMC0 - Core Mode Controller
- SCG0 – System Clock Generator
- LPSPI – Low Power Serial Peripheral Interface
- RGPIO2P – Rapid General-Purpose Input and Output with 2 Ports
- SIM - System Integration Module
- OCOTP - On-Chip One-Time-Programmable Controller
- SNVS - Secure Non-Volatile Storage

Following is the list of blocks used by A7 ROM:

- IOMUXC - I/O Multiplexer Control
- CAAM - Cryptographic Acceleration and Assurance Module
- CMC1 - Core Mode Controller
- SCG1 – System Clock Generator
- USBOTG2 - Used for serial download of a boot device provisioning program
- USDHC - Ultra Secure Digital Host Controller
- WDOG-1 - Watchdog Timer
- LPSPI – Low Power Serial Peripheral Interface
- RGPIO2P – Rapid General-Purpose Input and Output with 2 Ports
- SIM - System Integration Module
- OCOTP - On-Chip One-Time-Programmable Controller

35.2.4.3 Boot clock options

i.MX 7ULP defaults to 48 MHz FIRC as the initial boot clock as an option for low power boot. However, hardware allows additional options to clock the system slower to reduce the boot clock frequency to be able to provide further options to the application if low power becomes higher priority than boot time.

Table 35-9. M4 boot clock options

Boot Mode	Clock Source (Fuse)	QSPI_DIV(part of IP)	Fuse Controls[1] (Fuse/Dividers)	M4 Core/ System	QSPI Clock	Notes
LP Boot[2]	FIRC (48 MHz)	Div 1, 2, or 4	DIVCORE = 0	48 MHz/48 MHz	12/24/48 MHz	3
LP Boot	FIRC (48 MHz)	Div 1, 2, or 4	DIVCORE =1	24 MHz/24 MHz	12/24/48 MHz	3

Table continues on the next page...

Table 35-9. M4 boot clock options (continued)

Boot Mode	Clock Source (Fuse)	QSPI_DIV(part of IP)	Fuse Controls[1] (Fuse/Dividers)	M4 Core/ System	QSPI Clock	Notes
LP Boot	FIRC (48 MHz)	Div 1, 2, or 4	DIVCORE =0	48 MHz/48 MHz	12/24/48 MHhz	3
LP Boot	FIRC(48 MHz)	Div 1, 2, or 4	DIVCORE =1	24 MHz/24 MHz	12/24/48 MHz	3
Dual/Single Boot[4], [5]	SPLL[PFD0] (246 MHz)	Div 1 or 2	M4_LOW_FREQ_BT =0 DIVCORE=1 (DIV/2) DIVPLAT =0(DIV/1)	123 MHz/123 MHz	SDR: 12/48/72/80 MHz DDR: 12/45/63.5/69.5 MHz	3
Dual/Single Boot	SPLL[PFD0] (246 MHz)	Div 1 or 2	M4_LOW_FREQ_BT =1 DIVCORE=3 (DIV/4) DIVPLAT =0(DIV/1)	61.5 MHz/ 61.5 MHz	SDR: 12/48/72/80 MHz DDR: 12/45/63.5/69.5 MHz	3

1. FIRC_DIV set to 1 (Divide by 1) for all boot cases.
2. LP Boot selection is based on "*LP Boot*" boot configuration fuse
3. QSPI clock can be changed in QSPI Configuration header
4. Based on "*dual boot*" fuse. Only selectable when "*LP Boot*" fuse is unblown.
5. For Dual/Single boot option, customer can select/change clock settings based on QSPI configuration header.

Table 35-10. A7 boot clock options

Boot Mode	Clock Source	Fuse Controls (Fuse/Dividers)	A7 Core	NIC0/NIC1	DDR Clock	Comments
Dual Boot	A7 SPLL[PFD0] (413 MHz)	NIC0DIV=1 NIC1DIV=0 A7_LOW_FREQ_BT = 0	413 MHz	NIC0 = DDR/2 =176 MHz NIC1 = DDR/2 =176 MHz	APLL[PFD0] (352 MHz)	See Table 35-11 for SD clock options
Dual Boot	A7 SPLL[PFD0] (413 MHz)	NIC0DIV=1 NIC1DIV=0 A7_LOW_FREQ_BT = 1	206.5 MHz	NIC0 = DDR/2 = 176 MHz NIC1 = DDR/2 =176 MHz	APLL[PFD0] (352 MHz)	See Table 35-11 for SD clock options

Table 35-11. SD boot clock options

Modes	Clock Source	PCC DIVIDER	Internal DIV	SD Clock
Identification	AP PLL3[PFD0] (352 MHz)	DIV = 1 (Div by 2)	SDCLKFS=0x20, DVS=0x8, Div by (64 * 9)	305.5 KHz
MMC Normal	AP PLL3[PFD0] (352 MHz)	DIV = 1 (Div by 2)	SDCLKFS=0x10, DVS=0x4, Div by (2 * 5)	17.6 MHz
MMC DDR Normal	AP PLL3[PFD0] (352 MHz)	DIV = 1 (Div by 2)	SDCLKFS=0x00, DVS=0x3, Div by (2*4)	22 MHz

Table continues on the next page...

Table 35-11. SD boot clock options (continued)

Modes	Clock Source	PCC DIVIDER	Internal DIV	SD Clock
SD Normal/SDR12	AP PLL3[PFD0] (352 MHz)	DIV = 1 (Div by 2)	SDCLKFS=0x01, DVS=0x3, Div by (2 * 4)	22 MHz
MMC High	AP PLL3[PFD0] (352 MHz)	DIV = 1 (Div by 2)	SDCLKFS=0x00, DVS=0x4, Div by (1 * 5)	35.2 MHz
SD High/SDR25	AP PLL3[PFD0] (352 MHz)	DIV = 1 (Div by 2)	SDCLKFS=0x01, DVS=0x1, Div by (2 * 2)	44 MHz
MMC DDR high	AP PLL3[PFD0] (352 MHz)	DIV = 1 (Div by 2)	SDCLKFS=0x00, DVS=0x1, Div by (2 * 2)	44 MHz

35.2.4.4 MMU and cache

By default, both A7/M4 ROM enable I cache at the start of image download, unless been disabled by fuse A7_I-Cache_Disable/M4_I-Cache_Disable.

Both A7 and M4 ROM always enable MMU/MPU at very beginning. By default, ROM code sets all memories (except ROM code) as non-executable. Before valid entry point provisioned (for HAB closed, which means image be authenticated by ROM), ROM code will remove the non-executable by disabling MMU/MPU. So, if boot fails and customer loads some images by JTAG (or jump to QSPI NOR flash), the memories are non-executable. Customers can disable MMU/MPU by JTAG and then run the loadable image.

By default, A7's L1 data cache and L2 cache are enabled during image authentication unless these are disabled by fuse A7 D Cache Disable and M4 D Cache Disable. LMEM data cache is also enabled for M4 boot. Once HAB authentication completes, A7 ROM disables the L1 data cache and L2 cache, and M4 ROM disables the LMEM data cache.

Enabling the MMU and data cache when booting non-securely with SEC_CONFIG=Open, and setting the CSF pointer in the Image Vector Table to NULL, has no impact on the boot performance. With this configuration it is recommended to blow A7_D-Cache_Disable/M4_D-Cache_Disable fuses.

35.2.4.5 Exception handling

Once exception is thrown on A7 side during boot phase, A7 ROM goes into USB serial download mode, and if thrown on M4 side, M4 ROM will go into an endless loop.

After boot, the program image can overwrite the vectors as required.

35.2.4.6 Interrupt handling during boot

No special interrupt handling routines are required during the boot process. Interrupts are disabled during boot ROM execution on both A7 and M4 ROM.

35.2.4.7 SIM DGO GPR used by ROM

NOTE

SIM DGO general-purpose registers (GPRs) are in M4 "Always ON" domain with values retained in lowest power modes (M4 VLLS), where the core supply is not available.

SIM DGO_GPR	Usage by ROM	Description
SIM_DGO_GP1	M4 resume entry point	To store M4 entry point for resuming from low power mode. Must be in M4 TCM range
SIM_DGO_GP2	Argument for M4 resume	M4 ROM stores this value into CPU register R0 when jumping to M4 entry point. This is optional and can be used as other purpose if argument is not needed when resuming.
SIM_DGO_GP3	A7 core0 resume entry point	To store A7 entry point for a resuming from low power mode.
SIM_DGO_GP4	Argument for A7 core0 resume	A7 ROM stores this value into CPU register R0 when jumping to A7 entry point. This is optional and can be used as other purpose if argument is not needed when resuming.
SIM_DGO_GP5	Bits 31-8: Reserved for future BootROM Bits 7-4: Attack Counter for A7 ROM Bits 3-0: Attack Counter for M4 ROM	If ROM code detects any attack by hacker, the ROM code will add these 'counter' into SNVS domain and then issue a WDOG reset. After been reset, ROM goes into infinite loop on non-closed part, and issues WDOG reset immediately on closed part.
SIM_DGO_GP6	Reserved for future BootROM	—
SIM_DGO_GP7	M4 entry point	Provided by A7 image in Single Boot mode

Table continues on the next page...

SIM_DGO_GP8	Argument for M4	Provided by A7 image in Single Boot mode. This is optional and can be used as other purpose if argument is not needed when resuming.
SIM_DGO_GP9	Restored Boot_CFG	SW can override BOOT_CFG and issue a SW reset; then BootROM will retrieve boot configuration here. Will be disabled if DIR_BT_DIS blown.
SIM_DGO_GP10	Boot ROM internal Flag	See Table 35-12
SIM_DGO_GP11	Reserved	—

Table 35-12. Boot ROM Internal Flag (SIM_DGO_GP10) Definition

Bit position	Name	Description
Bit 31	Reserved	—
30	Secondary Boot	This bit identifies which image should be used, primary or secondary. Used only for boot device that supports secondary boot. For this device, secondary boot is only supported on SD/eMMC boot.
29	Reserved	—
28	WDOG_RESET_BOOT	Set by bootable image before issuing a WDOG reset, to indicate ROM to retrieve boot configuration from SIM_DGO_GP9 instead of from CMC0/1 Mode register. Can be disabled if DIR_BT_DIS is blown.
27-26	Reserved	—
25	A7 resume failure flag	If A7 resume failed (usually an invalid resume entry been provisioned), A7 ROM will set this bit to indicate such failure and then issues a WDOG reset.
24	M4 resume failure flag	If M4 resume failed (usually an invalid resume entry been provisioned), M4 ROM will set this bit to indicate such failure and then issues a WDOG reset.
23-0	Reserved	—

35.2.5 Boot device

35.2.5.1 Expansion device

35.2.5.1.1 Expansion Device eFUSE Configuration

SD/MMC/eSD/eMMC boot can be performed using either uSDHC ports, based on setting of the BOOT_CFG1[11-9] (A7 Boot Interface Selection) fuse or its associated GPIO input value at boot, and SD/MMC/eSD/eMMC boot supported on A7 ROM side only.

All uSDHC ports support eMMC4.4 and eMMC4.5 fast boot. See the table below for details.

Table 35-13. uSDHC Boot eFUSE Descriptions

Fuse	Config	Definition	GPIO	Shipped Value	Settings
BOOT_CFG1[11:9]	OEM	A7 Boot Interface Selection	Yes	000	**A7 Boot interface** 000—uSDHC0 001—uSDHC1 others—Reserved
BOOT_CFG1[8]	OEM	SD/MMC Selection	Yes	0	**uSDHC device type** 0—eMMC 1—SD/eSD
BOOT_CFG1[7:6]	OEM	MMC/eMMC Bus Width Selection	Yes	00	**MMC/eMMC Bus width Selection** 00—4-bit 01—8-bit 10—4-bit DDR 11—8-bit DDR
BOOT_CFG1[5]	OEM	MMC/eMMC Speed Selection	Yes	0	**MMC/eMMC speed selection** 0—Normal 1—High
BOOT_CFG1[7:5]	OEM	SD Speed Selection	Yes	00	**SD speed selection** 000—Normal/SDR12 001—High/SDR25 Others—Reserved
BOOT_CFG1[4]	OEM	• eMMC fast boot Selection • SD Loopback Clock Source Selection	Yes	0	**eMMC fast boot** 0—Disable 1—Enable **SD Loopback Clock Source Selection** 0—Through SD pad 1—Direct
0x550[31]	OEM	USDHC DLL Selection	No	0	**uSDHC DLL Selection** 0—DLL Slave Mode 1—DLL Override Mode
0x550[30:24]	OEM	USDHC DLL Number	No	000000	**uSDHC DLL Number**

Table continues on the next page...

Table 35-13. uSDHC Boot eFUSE Descriptions (continued)

Fuse	Config	Definition	GPIO	Shipped Value	Settings
				0	Delay target for uSDHC DLL: It is applied to slave mode target delay or override mode target delay depending on DLL Override fuse bit value
0x550[23]	OEM	Enable/Disable SDMMC Manufacture mode	No	0	**Disable SDMMC Manufacture mode** 0—Enable. 1—Disable
0x550[22]	OEM	uSDHC DLL Enabled selection	No	0	**uSDHC DLL Enable** 0—Disable DLL for SD/eMMC 1—Enable DLL for SD/eMMC
0x550[21]	OEM	USDHC Pad Setting Override Enabled	No	0	**USDHC Override Pad Settings** 0—Disable pad settings override 1—Override pad settings by PAD_SETTINGS value
0x550[20]	OEM	Enable or Disable USDHC device power reset	No	0	**Power cycle enable** 0—No power cycle 1—Enabled
0x550[19:8]	OEM	USDHC Pad Settings Number	No	000000 000000	**Bits shift** Bit[0]: PS Bit[1]: PE Bit[2]: SRE Bit[3]: PFE Bit[4]: ODE Bit[6:5]: DSE Bit[8:7]: ODT Bit[9]: IBE Bit[10]: OBE Bit[11]: PV
0x550[7:6]	OEM	Calibration pace number	No	00	**SD Calibration Step** 00—1 01—1 10—2 11—3
0x550[5:4]	OEM	USDHC power cycle interval selection	No	00	**uSDHC Power Cycle Interval** 00—20 ms 01—10 ms

Table continues on the next page...

Table 35-13. uSDHC Boot eFUSE Descriptions (continued)

Fuse	Config	Definition	GPIO	Shipped Value	Settings
					10—5 ms 11—2.5 ms
0x550[3]	OEM	USDHC Power cycle delay number selection, select wait time after reset	No	0	**USDHC Power Cycle Delay** 0—5 ms 1—2.5 ms
0x550[2]	OEM	USDHC power reset polarity selection	No	0	**USDHC Power On Polarity** 0—Low 1—High
0x550[1]	OEM	USDHC0 voltage selection	No	0	**USDHC0 IO Voltage selection** 0—3.3 V 1—1.8 V
0x550[0]	OEM	Disable/Enable eMMC acknowledge in fast boot case	No	0	**Fast Boot Acknowledge Disable:** 0—Boot Ack Disabled 1—Boot Ack Enabled
0x560[15]	OEM	USDHC1 voltage selection	No	0	**uSDHC1 IO Voltage Selection** 0—3.3 V 1—1.8 V

Boot code supports following standards. ROM doesn't support booting with 1-bit bus width for all kinds of SD/eSD/MMC/eMMC devices.

- MMCv4.4 or less
- eMMCv4.4 or less
- SDv2.0 or less
- eSDv2.10 rev-0.9, with or without FAST_BOOT.

MMC/SD/eSD/SDXC/eMMC can be connected to any of the USDHC blocks and can be booted by copying 4 KB of data from MMC/SD/eSD/eMMC device to internal RAM. After checking the Image Vector Table header value (0xD1) from Program Image, the ROM code performs a DCD check. After successful DCD extraction, the ROM code extracts from Boot Data Structure the destination pointer and length of image to be copied to RAM device from where code execution occurs.

The maximum image size to load in SD/MMC boot is 32 MB. This is due to the limited number of uSDHC ADMA Buffer Descriptors allocated by ROM.

NOTE

The Initial 4 KB of Program Image must contain the IVT, DCD and the Boot Data structures.

NOTE

i.MX 7ULP A7 ROM supports SDXC card (SD3.0) in normal SD(SD2.0) compatible way.

Table 35-14. SD/MMC Frequencies

	SD	MMC	MMC (DDR Mode)
Identification (KHz)	305.5 kHz		
Normal Speed Mode (MHz)	22	17.6	22
High Speed Mode (MHz)	44	35.2	44

NOTE

The boot ROM code reads application image length and application destination pointer from image.

35.2.5.1.2 MMC and eMMC Boot

The following table provides MMC and eMMC boot details.

Table 35-15. MMC and eMMC Boot Details

Normal Boot Mode	During initialization (normal boot mode) the MMC frequency is set to 305.5 kHz. When the MMC card enters the identification portion of the initialization, voltage validation is performed and the ROM boot code checks high voltage settings and card capacity. The ROM boot code supports both high capacity and low capacity MMC/eMMC cards. After initialization phase is complete, the ROM boot code switches to a higher frequency (17.6 MHz in Normal boot mode or 35.2 MHz in High Speed mode). eMMC is also interfaced via USDHC and follows the same flow as MMC. The boot partition can be selected for an MMC4.x card after the card initialization is complete. The ROM code reads the BOOT_PARTITION_ENABLE field in the Ext_CSD[179] to get the boot partition to be set. If there is no boot partition mentioned in BOOT_PARTITION_ENABLE field or the user partition has been mentioned, ROM boots from the user partition.
eMMC4.3 or eMMC4.4 Device Supporting Special Boot Mode	If using an eMMC4.3 or eMMC4.4 device supporting special boot mode, it can be initiated by pulling the CMD line low. If BOOT ACK is enabled, the eMMC4.3/eMMC4.4 device sends the BOOT ACK via DATA lines and ROM can read the BOOT ACK [S010E] to identify the eMMC4.3/eMMC4.4 device. eMMC4.3/eMMC4.4 device with "Boot mode" feature can only be supported via ESDHCV3-3 and with or without BOOT

Table continues on the next page...

Table 35-15. MMC and eMMC Boot Details (continued)

	ACK. If BOOT ACK is enabled, ROM waits 50 ms to get the BOOT ACK and if BOOT ACK is received by ROM. If BOOT ACK is disabled, ROM waits 1 second for data. If BOOT ACK or data was received then eMMC4.3/eMMC4.4 is booted in "Boot mode", otherwise eMMC4.3/eMMC4.4 boots as a normal MMC card from the selected boot partition. This boot mode can be selected by BOOT_CFG1[4] (Fast Boot) fuse. BOOT ACK is selected by BOOT_CFG2[1].
eMMC4.4 Device	If using eMMC4.4 device, Double Data Rate (DDR) mode can be used. This mode can be selected by BOOT_CFG2[7:5] (Bus Width) fuse.

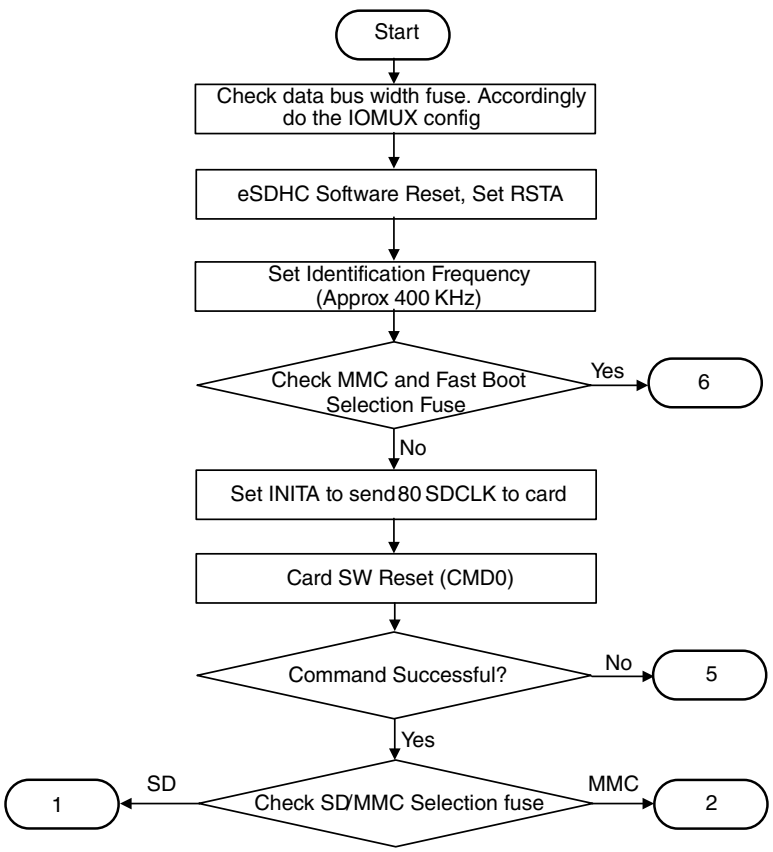


Figure 35-7. Expansion Device Boot Flow (1 of 6)

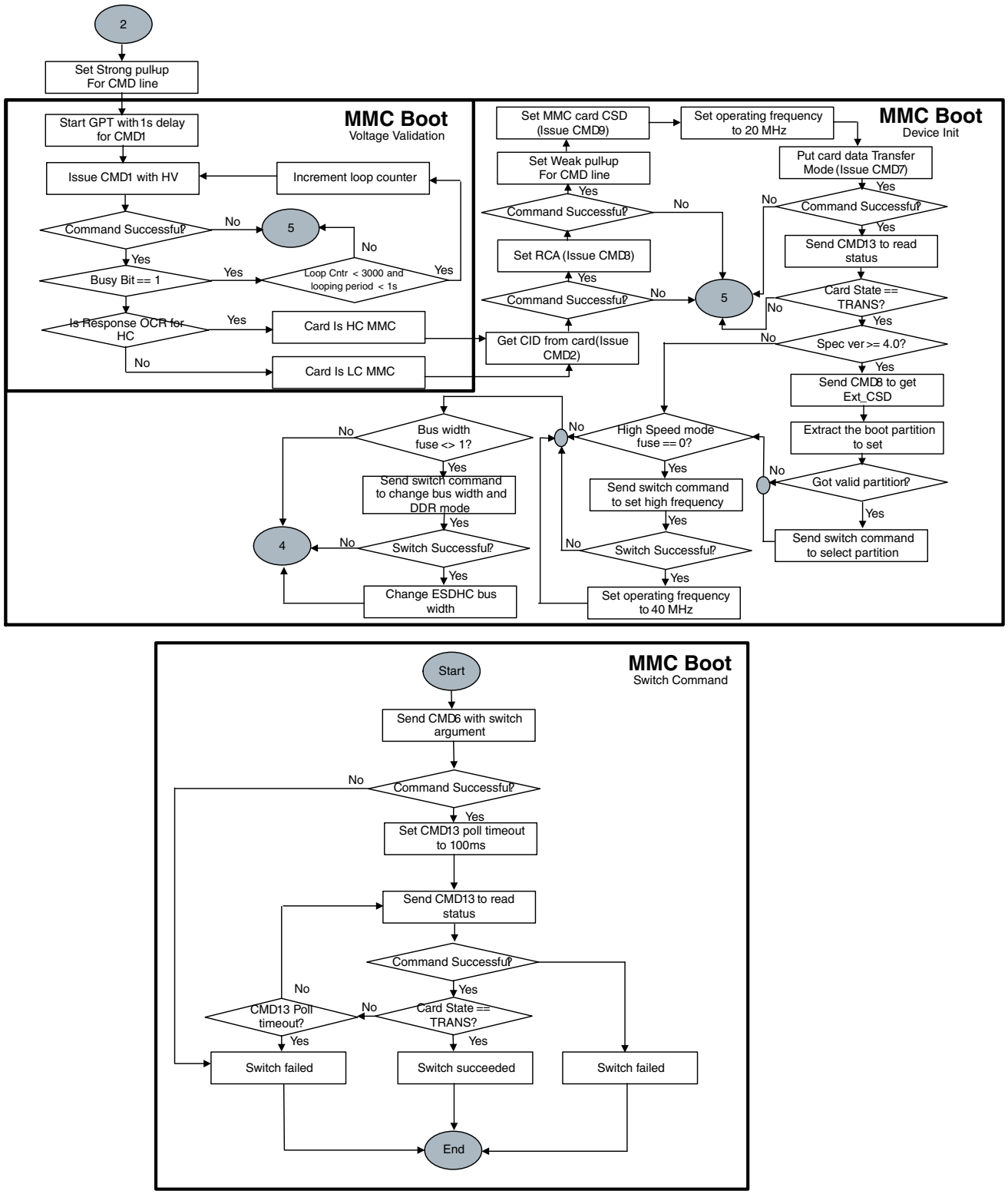


Figure 35-8. Expansion Device (MMC) Boot Flow (2 of 6)

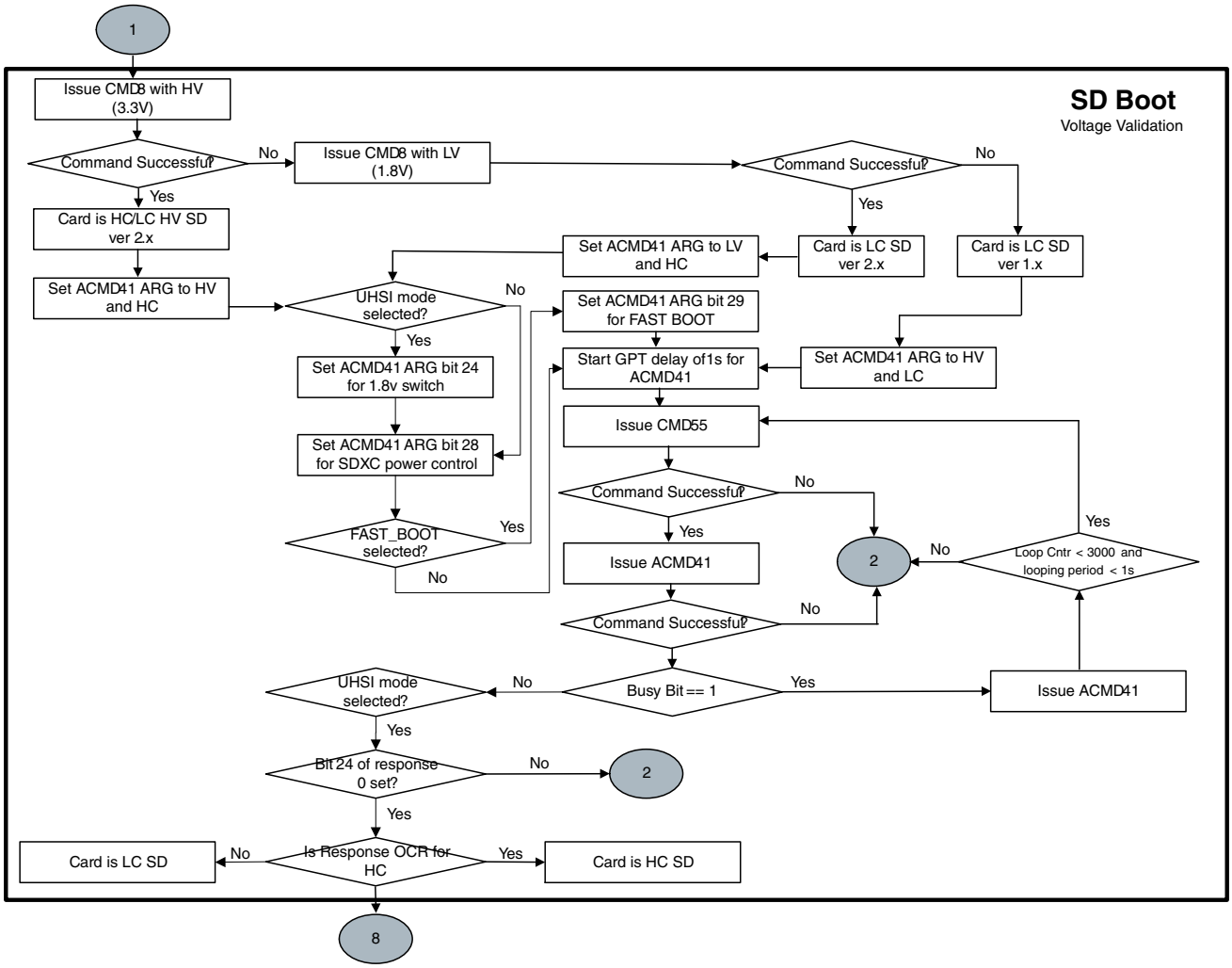


Figure 35-9. Expansion Device (SD/eSD/SDXC) Boot Flow (3 of 6) Part 1

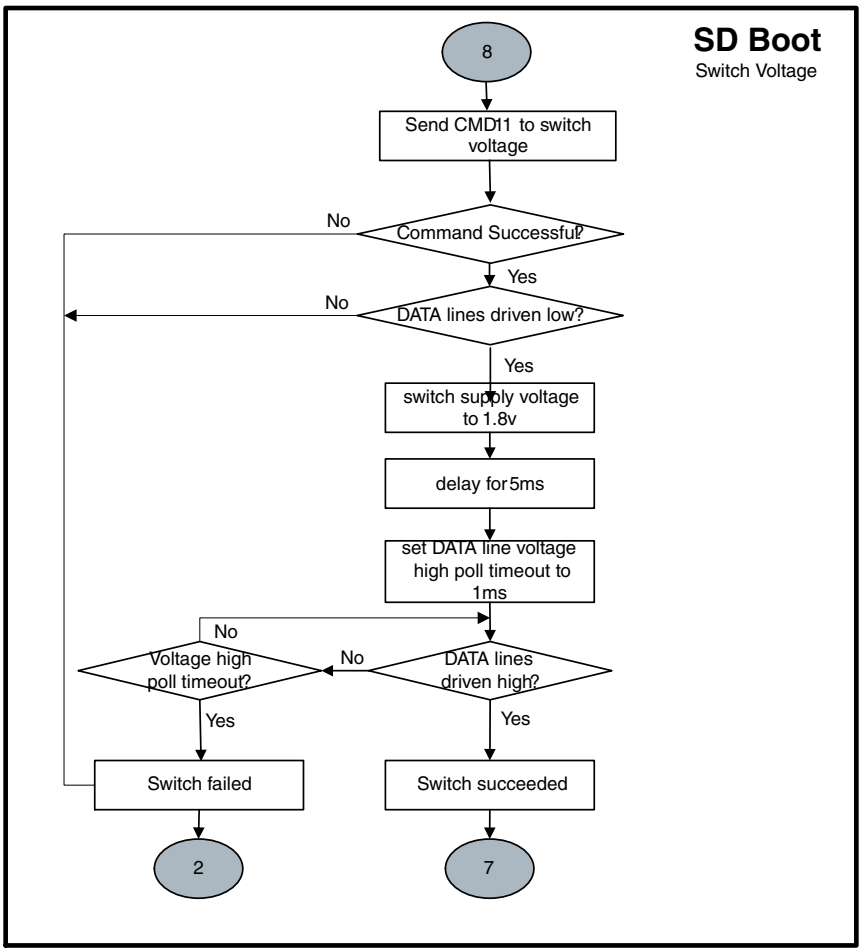


Figure 35-10. Expansion Device (SD/eSD/SDXC) Boot Flow (3 of 6) Part 2

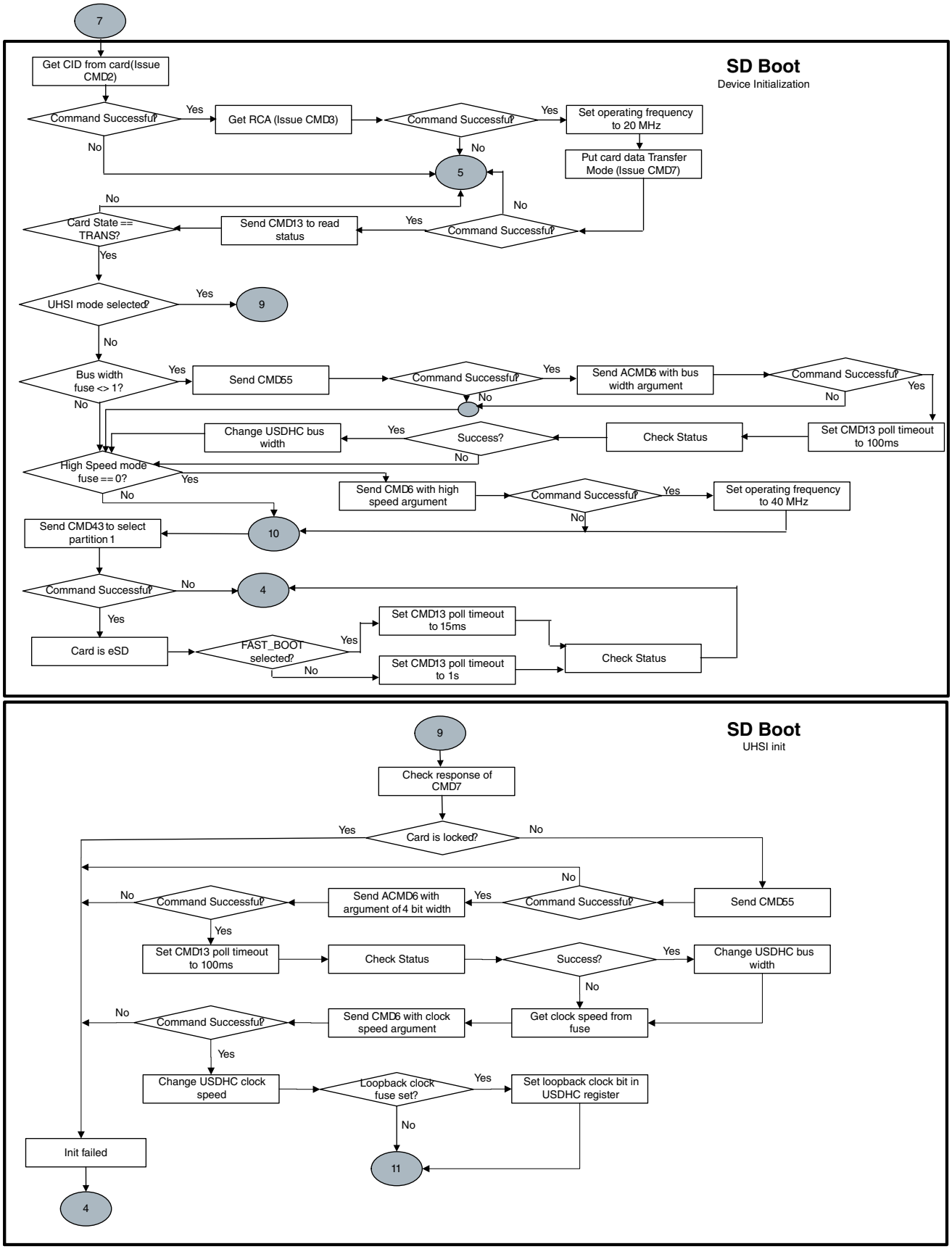


Figure 35-11. Expansion Device (MMCSD/eSD/SDXC) Boot Flow (4 of 6)

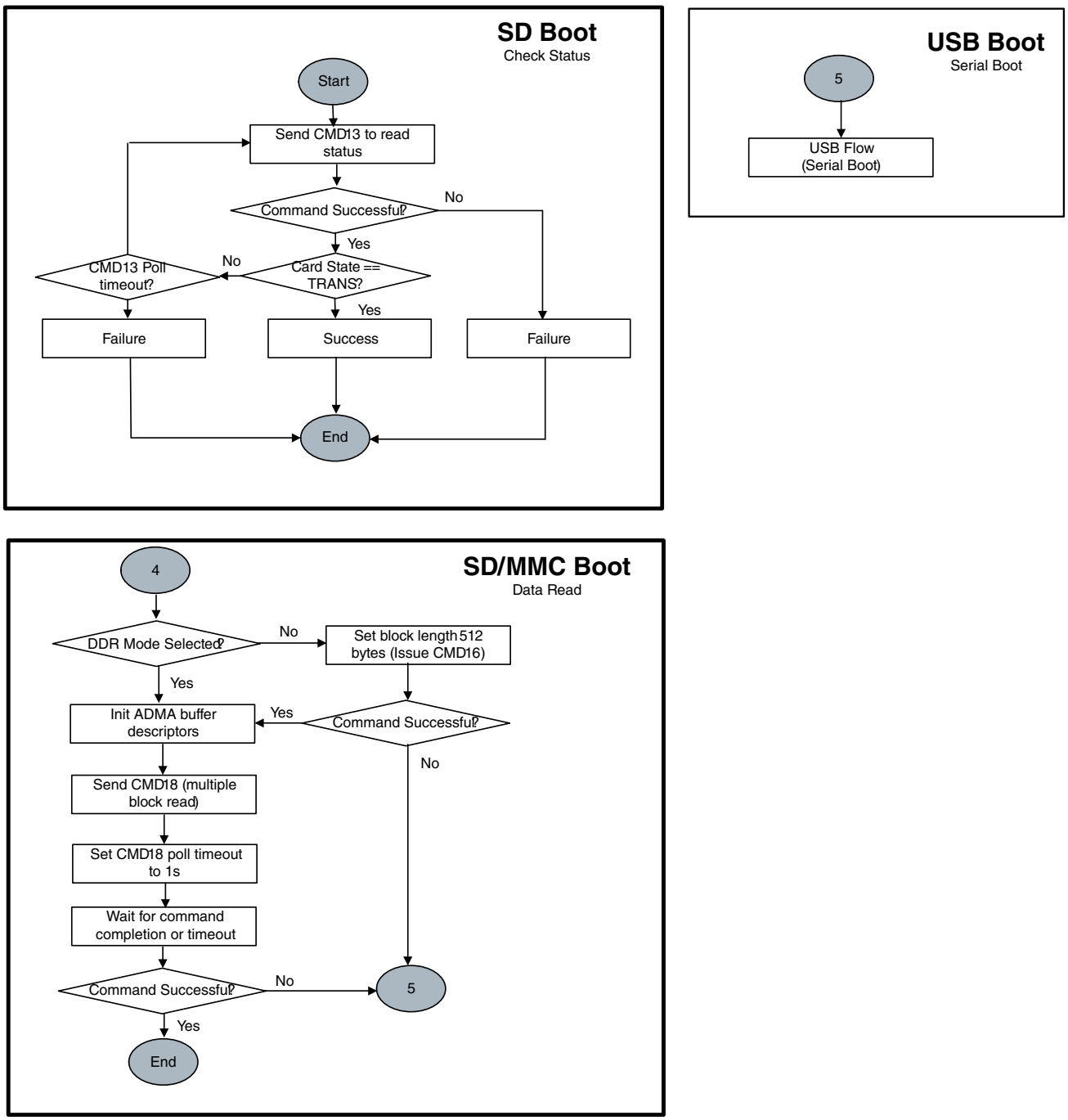
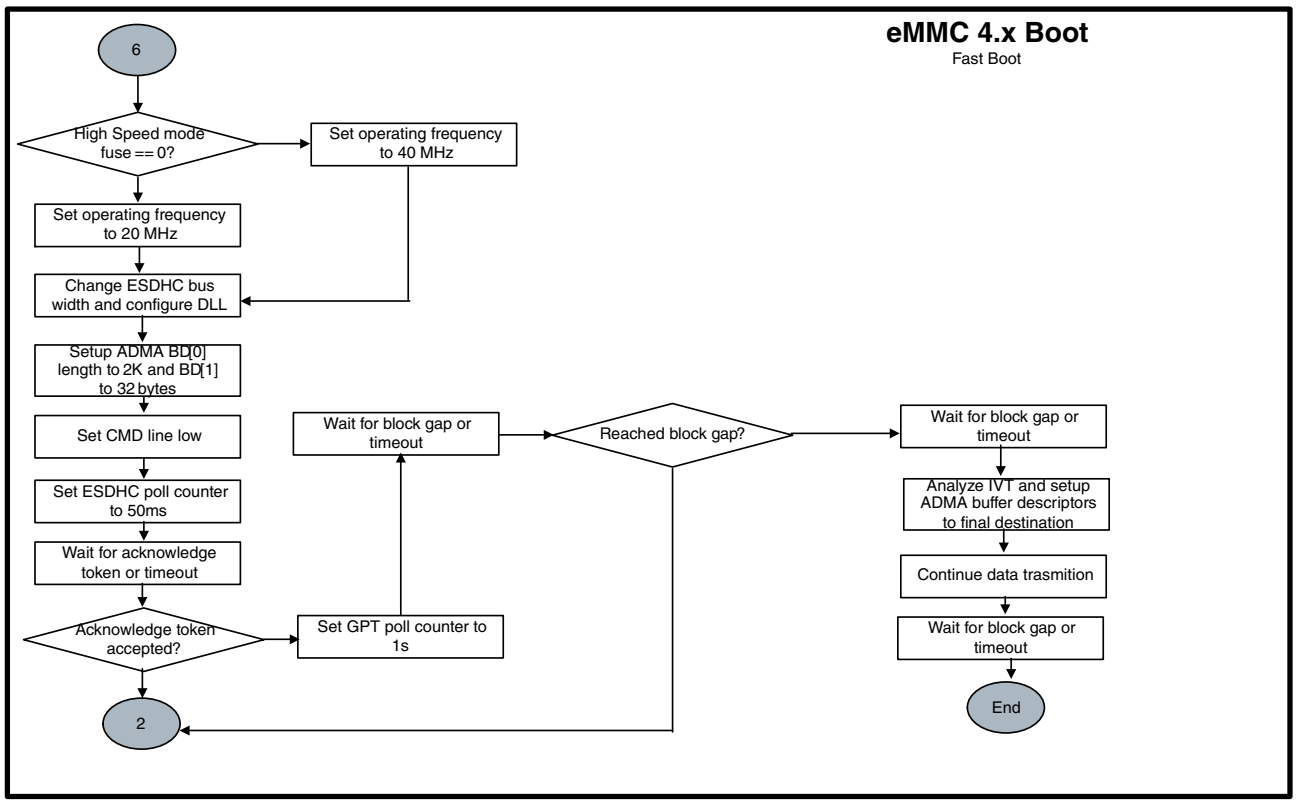


Figure 35-12. Expansion Device (SD/eSD) Boot Flow (5 of 6)



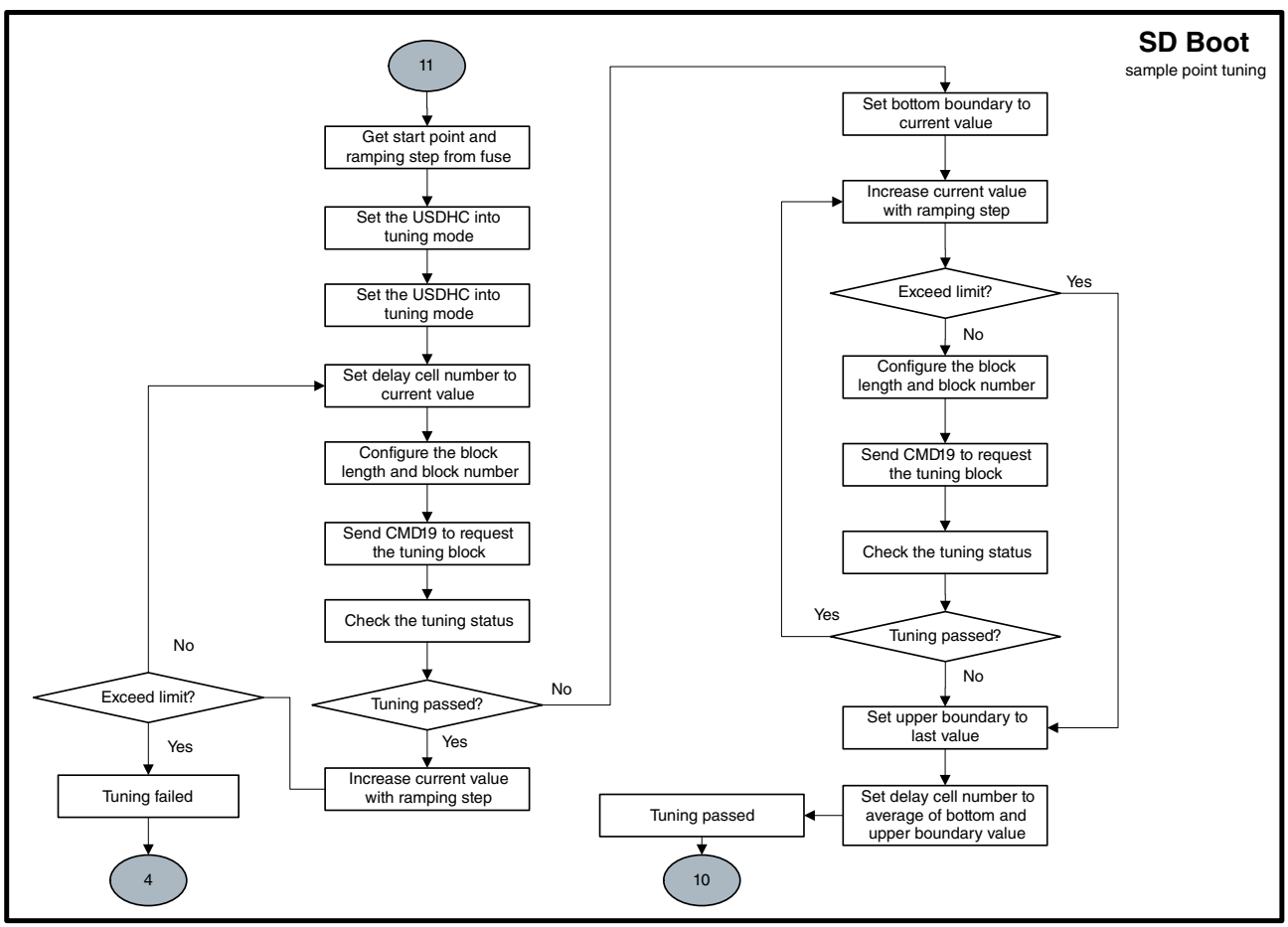


Figure 35-13. Expansion Device Boot Flow (6 of 6)

35.2.5.1.3 SD, eSD and SDXC

NOTE

A7 ROM supports SD3.0 card in SD2.0 compatible way, thus SDXC card will be taken as a normal SD2.0 card.

After the normal boot mode initialization begins, the SD/eSD/SDXC frequency is set to 305.5 kHz. During the identification phase, SD/eSD/SDXC card voltage validation is performed. During voltage validation, boot code first checks with high voltage settings; if that fails, it checks with low voltage settings. The capacity of the card is also checked. Boot code supports high capacity and low capacity SD/eSD/SDXC cards after voltage validation card initialization is done.

During card initialization, the ROM boot code attempts to set the boot partition for all SD, eSD, and SDXC devices. If this fails, the boot code assumes the card is a normal SD card or SDXC card. If it does not fail, the boot code assumes it is an eSD card. After the initialization phase is over, boot code switches to a higher frequency (22 MHz in Normal Speed mode or 44 MHz in High Speed Mode). ROM also supports FAST_BOOT mode booting from eSD card. This mode can be selected by BOOT_CFG1[4] (Fast Boot) fuse described in Table 35-13.

UHSI calibration start value (MMC_DLL_DLY[6:0]) and step value (BOOT_CFG2[7:5]) can be set to optimize the sample point tuning process.

If SD Power Cycle Enable eFuse is 1, ROM will set SD_RST pad low, wait 5ms and then set SD_RST pad high. If SD_RST pad is connected to SD power supply enable logic on board, it enables power cycle of SD card. This may be crucial when SD logic is in 1.8 V states and must be reset to 3.3 V states.

SDR50 and SDR104 boot are not supported on the USDHC0 and USDHC1 ports because there are no reset signals for those ports when connected in the IOMUX.

35.2.5.1.4 IOMUX Configuration for SD/MMC

Table 35-16. IOMUX Configuration for uSDHC

Signal	uSDHC0	uSDHC1
CLK	PTD2.alt8	PTE2.alt8
CMD	PTD1.alt8	PTE3.alt8
DATA0	PTD10.alt8	PTE1.alt8
DATA1	PTD9.alt8	PTE0.alt8
DATA2	PTD8.alt8	PTE5.alt8
DATA3/CD_B	PTD7.alt8	PTE4.alt8 (DATA3 only)
DATA4	PTD6.alt8	PTE6.alt8

Table continues on the next page...

Table 35-16. IOMUX Configuration for uSDHC (continued)

Signal	uSDHC0	uSDHC1
DATA5	PTD5.alt8	PTE7.alt8
DATA6	PTD4.alt8	PTE8.alt8
DATA7	PTD3.alt8	PTE9.alt8
VSELECT	No VSELECE signal	PTE14.alt8
RESET_B	PTD0.alt1	PTE11.alt1

35.2.5.1.5 Redundant Boot Support for Expansion Device

ROM supports redundant boot for expansion device. Primary or Secondary image is selected depending on PERSIST_SECONDARY_BOOT setting provisioned in the GPRs.

If PERSIST_SECONDARY_BOOT is 0, the boot ROM uses address 0x0 for primary image.

If PERSIST_SECONDARY_BOOT is 1, the boot ROM will read secondary image table from address 0x200 on boot media and will use address specified in the table for the secondary image.

Table 35-17. Secondary Image Table Format

Reserved (chipNum)
Reserved (driveType)
tag
firstSectorNumber
Reserved (sectorCount)

Where:

- tag: used as indication of valid secondary image table. Must be 0x00112233.
- firstSectorNumber is the first 512B sector number of the secondary image.

For secondary image support, the primary image must reserve space for secondary image table. See the figure below for typical structures layout on expansion device.

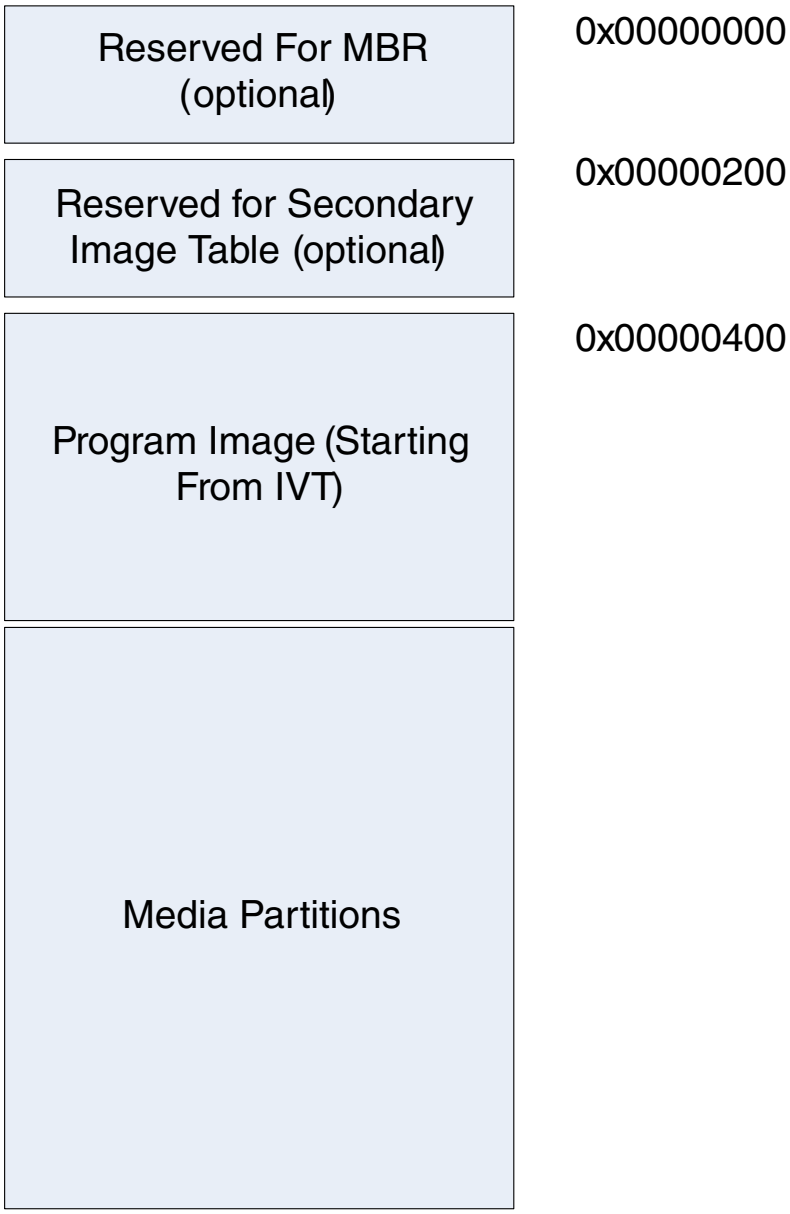


Figure 35-14. Expansion Device Structures Layout

For Closed mode, if there are failures during primary image authentication, the boot ROM will turn on PERSIST_SECONDARY_BOOT bit and perform software reset. (After software reset, secondary image will be used.)

35.2.5.2 QuadSPI serial flash memory boot

35.2.5.2.1 QuadSPI eFUSE Configuration

Table 35-18. QuadSPI efUSE Configuration

Fuse	Config	Definition	GPIO	Shipped value	Settings
BOOT_CFG0[15:14]	OEM	QSPI instance selection	Yes	00	**QSPI instance** 00—QSPI0 Others—Reserved
BOOT_CFG0[13:12]	OEM	QSPI device type Selection	Yes	00	**QSPI device type** 00—Flash with 3B READ (0x03) default supported 01—Hyperflash 1.8 V 10—Hyperflash 3.3 V 11—Flash with 4B READ (0x13) default supported

35.2.5.2.2 QuadSPI Serial Flash BOOT Operation

The M4 Boot ROM attempts to boot from QuadSPI flash if the BOOT_CFG0[15:14] fuses are programmed to 00 as shown in Table 35-18. The ROM will initialize the requested the QuadSPI Interface as selected in Fuse bit BOOT_CFG0[13:12] in the Table 35-18. QuadSPI interface initialization is a two step process.

The ROM expects the QuadSPI configuration parameters as explained in the QuadSPI Configuration Parameters to be present in the Serial Flash memory from offset 0x1000 of serial flash of length 512 bytes. The ROM reads these configuration parameters using the default read command configured in the LUT of the QuadSPI interface with SCLOCK operating at 18 MHz.

In the second step, ROM configures the selected QuadSPI interface with the configuration parameters read from the serial flash and starts the boot procedure. See Table 35-19 for details regarding QuadSPI configuration parameters and to the QuadSPI boot flow chart for detailed boot flow chart of QuadSPI.

Both booting an XIP and non XIP image is supported from serial flash. For XIP boot, the image has to be built for QuadSPI address space and for non XIP the image can be built to execute from TCM.

For QUAD mode boot, the Boot ROM expects the Quad Enable bit inside the QSPI Flash to be already set before booting starts. Therefore, the QUAD enable bit must be set in the non-volatile register of the flash at the time of programming.

NOTE

If the SPI flash device requires quad enable command, it can be sent via configuration structure fields: device_quad_mode_en, device_cmd, write_cmd_ipcr, write_enable_ipcr, busy_bit_offset, read_status_ipcr.

35.2.5.2.3 QuadSPI Configuration Parameters

The QuadSPI Configuration Parameters Table is built in boot image at fixed offset 0x400 from QSPI NOR A1 base address (368 bytes). Table below lists various QuadSPI Configuration Parameters.

Table 35-19. QuadSPI Configuration Parameters

Name	Offset	Size in Bytes	Description
DQS Loopback	0	4	DQS LoopBack Mode to enable Dummy Pad, 0 - Disable, 1 - Enable
Hold Delay	4	4	Hold Delay for QSPI[0,1] A/B Value / QSPI1 B / QSPI1 A 00 / Disable / Disable 01 / Disable / Enable 10 / Enable / Disable 11 / Enable / Enable
Reserved	8	4	Reserved to 0
Reserved	12	4	Reserved to 0
device_quad_mode_en	16	4	Send Quad enable command to SPI device.
device_cmd	20	4	Command to send to SPI device.
write_cmd_ipcr	24	4	IPCR register value for write command
write_enable_ipcr	28	4	IPCR register value for Enable
Chip Select hold time	32	4	This is chip select hold time in terms of Serial clock (For Example 1 serial clock cycle 0-15).
Chip Select setup time	36	4	Chip select setup time in terms of Serial clock (For example 1 serial clock).
Serial Flash A1 size	40	4	Serial Flash A1 size in units of bytes
Serial Flash A2 size	44	4	Serial Flash A2 size in units of bytes
Reserved	48	-	-
Reserved	52	-	-
Serial Clock Frequency	56	4	This is serial clock frequency select parameter.

Table continues on the next page...

Table 35-19. QuadSPI Configuration Parameters (continued)

<table>
<tr><th>Name</th><th>Offset</th><th>Size in Bytes</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><th>Value</th><th>Clock</th></tr>
<tr><td>00</td><td>12 MHz for both STR/DDR</td></tr>
<tr><td>01</td><td>45 MHz for DTR or 48 MHz for STR</td></tr>
<tr><td>02</td><td>63.5 MHz for DTR or 72 MHz for STR</td></tr>
<tr><td>03</td><td>69.5 MHz for DTR or 90 MHz for STR</td></tr>
<tr><td>04-06</td><td>NA</td></tr>
</table>
</td></tr>
<tr><td>busy_bit_offset</td><td>60</td><td>4</td><td>SPI Flash device busy bit offset in its status register, used for enabling Quad mode of SPI device</td></tr>
<tr><td>Mode of operation of serial Flash</td><td>64</td><td>4</td><td>This field describes the mode of operation of Serial flash
<table>
<tr><th>Value</th><th>Mode</th></tr>
<tr><td>01</td><td>Single</td></tr>
<tr><td>02</td><td>Dual</td></tr>
<tr><td>04</td><td>Quad</td></tr>
<tr><td>08</td><td>Octal</td></tr>
</table>
</td></tr>
<tr><td>Serial Flash Port B Selection</td><td>68</td><td>4</td><td>Port A is always available. This field informs the device ROM the availability of Port B.
0 – Port B is not used
1 – Port B is used</td></tr>
<tr><td>Dual Data Rate mode enable</td><td>72</td><td>4</td><td>This field enables the device ROM to enable DDR mode.
0 – DDR mode is disabled
1 – DDR mode is enabled</td></tr>
<tr><td>Data Strobe Signal enable in Serial Flash</td><td>76</td><td>4</td><td>This field enables Data Strobe signal in Serial Flash which supports it.
0 – Disable DQS
1 – Enable DQS</td></tr>
<tr><td>Reserved</td><td>80</td><td>-</td><td>-</td></tr>
<tr><td>CS1 on Port A</td><td>84</td><td>4</td><td>This field enables CS1 on port A
0 – Disable CS1 on Port A
1 – Enable CS1 on Port A</td></tr>
<tr><td>CS1 on Port B</td><td>88</td><td>4</td><td>This field enables CS1 on port B
0 – Disable CS1 on Port B
1 – Enable CS1 on Port B</td></tr>
<tr><td>Full Speed Phase Selection</td><td>92</td><td>4</td><td>Select the edge of the sampling clock valid for full speed commands:
0: Select sampling at non-inverted clock
1: Select sampling at inverted clock
This bit is also used to shift the dqs_enable when DQS mode is selected</td></tr>
</table>

Table continues on the next page...

Table 35-19. QuadSPI Configuration Parameters (continued)

Name	Offset	Size in Bytes	Description
Full Speed Delay Selection	96	4	Select the delay w.r.t. the reference edge for the sample point valid for full speed commands: 0: One clock cycle delay 1: Two clock cycles delay This bit is also used to shift the dqs_enable when DQS mode is selected
DDR Sampling Point	100	4	Select the sampling point for incoming data when serial flash is in DDR mode. **NOTE:** Valid Values are (b000-b111)
LUT program sequence	104	256	256 Bytes of Look up table program sequence. ROM programs the LUT of QuadSPI with this parameter supplied. It assumes that the optimize read command sequence which will be used to read data from Serial flash and fill the AHB buffer is programmed at index 0.
read_status_ipcr	360	4	IPCR value of Read Status Reg
enable_dqs_phase	364	4	Enable DQS Phase
Reserved	368	36	Not used
dqs_pad_setting_override	404	4	DQS pin pad setting override
sclk_pad_setting_override	408	4	SCLK pin pad setting override
data_pad_setting_override	412	4	DATA pins pad setting override
cs_pad_setting_override	416	4	CS pins pad setting override
dqs_loopback_internal	420	4	0: dqs loopback from pad 1: dqs loopback internally
dqs_phase_sel	424	4	dqs phase selection
dqs_fa_delay_chain_sel	428	4	dqs fa delay chain selection
dqs_fb_delay_chain_sel	432	4	dqs fb delay chain selection
sclk_fb_delay_chain_sel	440	4	sclk fb delay chain selection
Reserved	444	64	Reserved
tag	508	4	End flag of QSPI parameters area

35.2.5.2.4 IOMUX Configuration for QSPI Devices

The QSPI interface uses dedicated contacts on the device. The contacts assigned to the data signals used by QSPI are shown in the table below.

Signal	QSPI1
SCLK	PTB15.alt8
SCLK_B	PTB14.alt9
DQS	PTB9.alt8
SS0_B	PTB8.alt8
SS1_B	PTB7.alt8
DATA0	PTB19.alt8
DATA1	PTB18.alt8
DATA2	PTB17.alt8
DATA3	PTB16.alt8
DATA4	PTB13.alt8
DATA5	PTB6.alt8
DATA6	PTB5.alt8
DATA7	PTB4.alt8

35.2.5.2.5 QSPI NOR flash

M4 ROM supports encrypted boot on QSPI NOR flash using OTFAD hardware. For details, see the OTFAD chapter in i.MX 7ULP Security Reference Manual.

35.2.5.2.6 QuadSPI boot flow chart

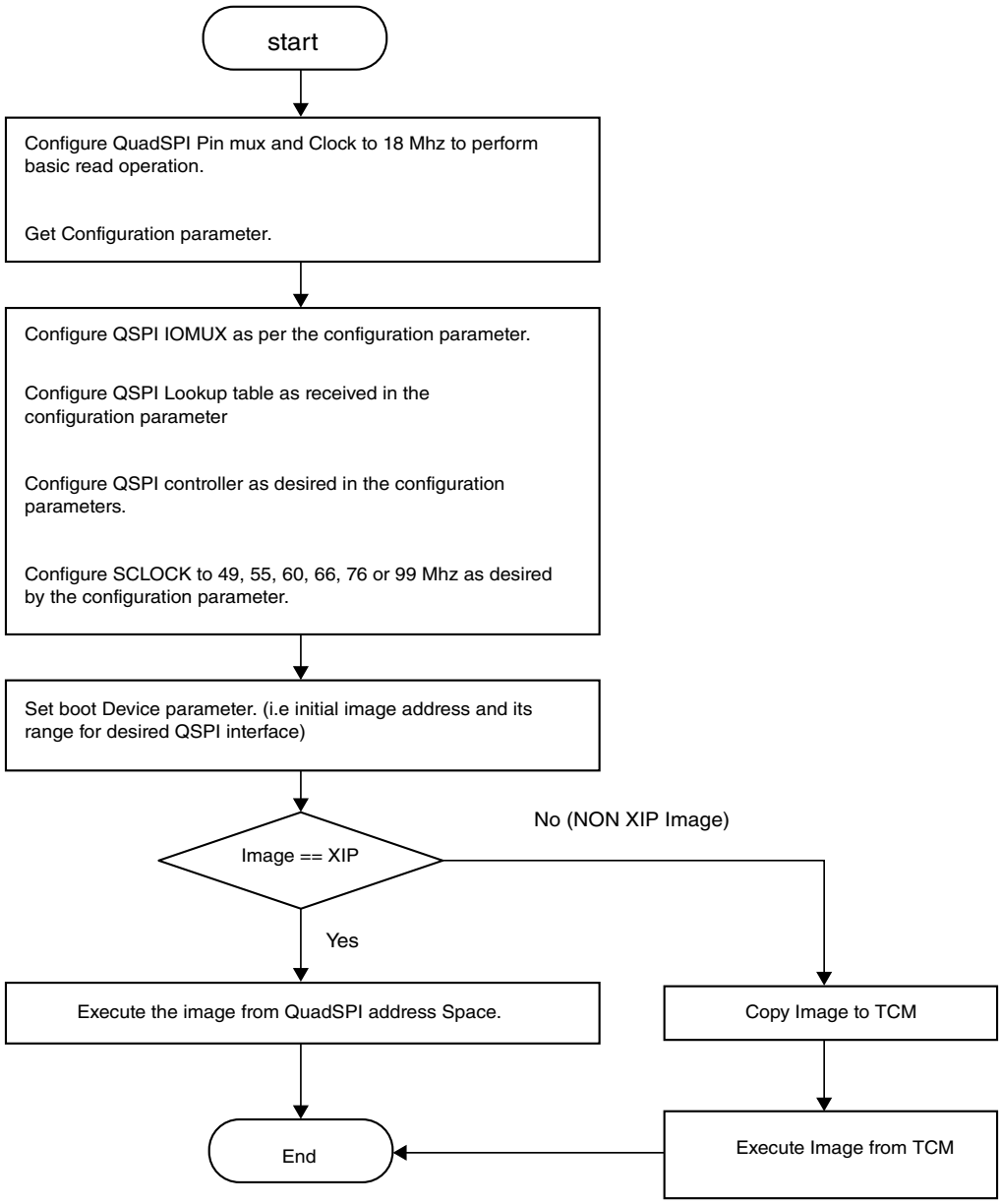


Figure 35-15. QuadSPI boot flow chart

NOTE

If flash is configured for "High performance mode (where command is generated only once)" in LUT program sequence, then external reset should be routed to flash reset to allow rebooting in case of any device reset other than Power On Reset. Also this high performance mode must be exited by application before any Low power mode entry where the device is supposed to reboot from QSPI flash on Low power mode exit. In general, any preserved configuration in external flash will not be understood by device after reset.

35.2.5.3 Recovery flash on LPSPI flash

This device supports recovery devices. If primary boot device fails, A7 ROM and M4 ROM will try to boot from recovery device using one of LPSPI ports, if not been disabled by fuse 'Disable M4 SPI Recovery boot' or 'Disable A7 SPI Recovery boot'.

35.2.5.3.1 LPSPI eFuse configuration

Fuse	Config	Definition	GPIO	Shipped value	Settings
0x560[13]	OEM	Disable/Enable M4 SPI boot	No	0	**Disable M4 SPI Recovery boot** 0—Disable 1—Enable
0x560[12]	OEM	Disable/Enable A7 SPI boot	No	0	**Disable A7 SPI Recovery boot** 0—Disable 1—Enable
0x560[11:10]	OEM	M4 SPI recovery boot port selection	No	00	**M4 Recovery Port Select:** 00—LPSPI0 01—LPSPI1 Others—Reserved
0x560[9:8]	OEM	M4 SPI recover boot CS selection	No	00	**M4 Recovery CS select (SPI only):** 00—CS#0 01—CS#1 10—CS#2 11—CS#3
0x560[7:6]	OEM	A7 SPI recovery boot port selection	No	00	**A7 Recovery Port Select:** 00—LPSPI2 01—LPSPI3 Others—Reserved
0x560[5:4]	OEM	A7 SPI recover boot CS selection	No	00	**A7 Recovery CS select (SPI only):** 00—CS#0 01—CS#1 10—CS#2 11—CS#3

Table continues on the next page...

Fuse	Config	Definition	GPIO	Shipped value	Settings
0x560[3]	OEM	SPI recovery boot addressing width selection	No	0	**Recovery SPI Addressing:** 0—3-bytes (24-bit) 1—2-bytes (16-bit)
0x560[2]	OEM	SPI CS active polarity selection	No	0	**SPI SS Active** 0—Active low 1—Active high
0x560[1]	OEM	SPI clock active polarity selection	No	1	**SPI SCLK Active** 0—Active high 1—Active low
0x560[0]	OEM	SPI clock data phase selection	No	0	**SPI Clock/Data Phase Control** 0—Tx on falling, Rx on rising 1—Tx on rising, Rx on falling

35.2.5.3.2 LPSPI pin muxing and pad setting

Table 35-20. LPSPI IOMUX configuration

Signal	LPSPI0	LPSPI1	LPSPI2	LPSP3
SCK	PTA6.alt3	PTA14.alt3	PTC10.alt3	PTC18.alt3
SIN	PTA4.alt3	PTA12.alt3	PTC8.alt3	PTC16.alt3
SOUT	PTA5.alt3	PTA13.alt3	PTC9.alt3	PTC17.alt3
PCS0	PTA3.alt3	PTA15.alt3	PTC11.alt3	PTC19.alt3
PCS1	PTA0.alt3	PTA8.alt3	PTC4.alt3	PTC12.alt3
PCS2	PTA1.alt3	PTA9.alt3	PTC5.alt3	PTC13.alt3
PCS3	PTA2.alt3	PTA10.alt3	PTC6.alt3	PTC14.alt3

35.2.6 Program image

This section describes the data structures that are required to be included in a user's program image. A program image consists of:

- Image vector table—A list of pointers located at a fixed address that the ROM examines to determine where other components of the program image are located
- Boot data—A table indicating the program image location, program image size in bytes, and the plugin flag

- Device configuration data—IC configuration data
- User code and data

35.2.6.1 Image Vector Table and Boot Data

The Image Vector Table (IVT) is the data structure that the ROM reads from the boot device supplying the program image containing the required data components to perform a successful boot.

The IVT includes the program image entry point, a pointer to Device Configuration Data (DCD) and other pointers used by the ROM during the boot process.The ROM locates the IVT at a fixed address that is determined by the boot device connected to the Chip. The IVT offset from the base address and initial load region size for each boot device type is defined in the table below. The location of the IVT is the only fixed requirement by the ROM. The remainder or the image memory map is flexible and is determined by the contents of the IVT.

Table 35-21. Image Vector Table Offset and Initial Load Region Size

Boot Device Type	Image Vector Table Offset	Initial Load Region Size
QSPI NOR	4 Kbyte = 0x1000 bytes	Entire Image Size
NAND	1 Kbyte = 0x400 bytes	4 Kbyte
SD/MMC/eSD/eMMC	1 Kbyte = 0x400 bytes	4 Kbyte
SPI Flash	1 Kbyte = 0x400 bytes	4 Kbyte

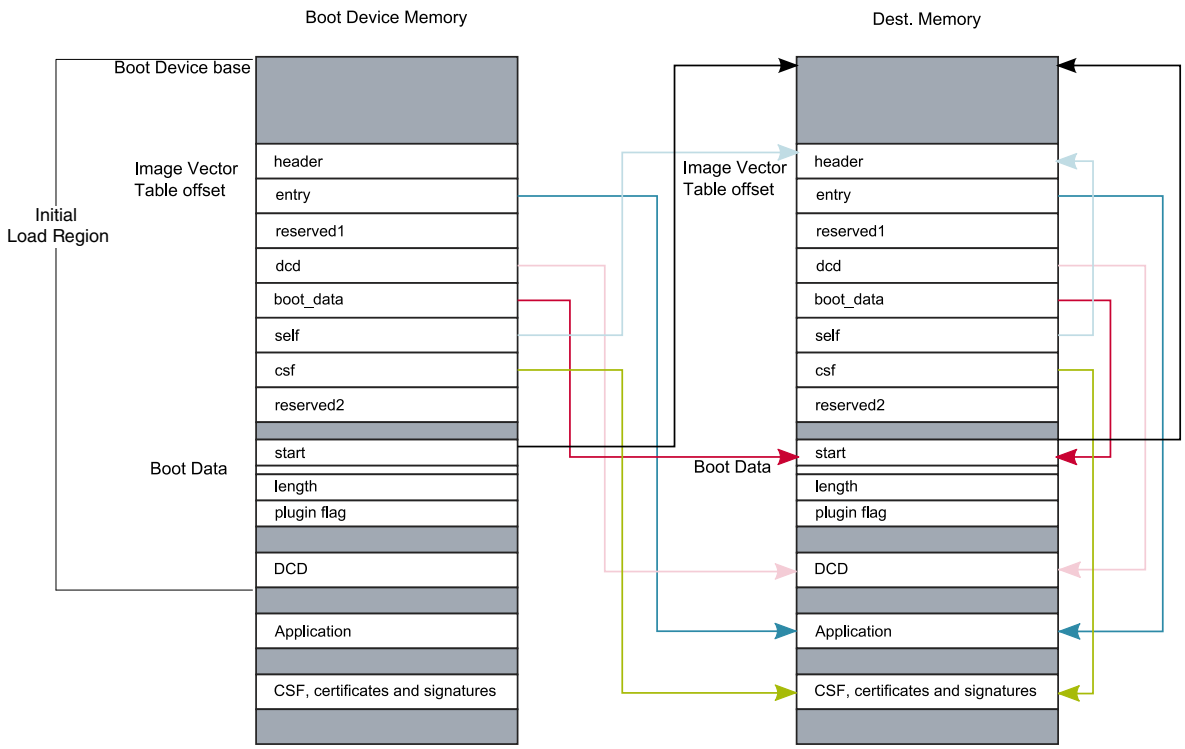


Figure 35-16. Image Vector Table

35.2.6.1.1 Image Vector Table Structure

The IVT has the following format where each entry is a 32 bit word:

Table 35-22. IVT Format

header
entry: Absolute address of the first instruction to execute from the image
reserved1: Reserved and should be zero
dcd: Absolute address of the image DCD. The DCD is optional so this field may be set to NULL if no DCD is required. See Device Configuration Data (DCD) for further details on DCD.
boot data: Absolute address of the Boot Data
self: Absolute address of the IVT. Used internally by the ROM
csf: Absolute address of Command Sequence File (CSF) used by the HAB library. See High Assurance Boot (HAB) for details on secure boot using HAB. This field must be set to NULL when not performing a secure boot
reserved2: Reserved and should be zero

Figure 35-17 shows the IVT header format:

Figure 35-17. IVT Header Format

where:

```
Tag: A single byte field set to 0xD1
Length: a two byte field in big endian format containing the overall length of the IVT,
in bytes, including the header. (the length is fixed and must have a value of
32 bytes)
Version: A single byte field set to 0x40 or 0x41
```

35.2.6.1.2 Boot Data Structure

The Boot Data must follow the format defined in the table found here, each entry is a 32-bit word.

Table 35-23. Boot Data Format

start	Absolute address of the image
length	Size of the program image
plugin	Plugin flag (see Plugin Image)

35.2.6.2 Device Configuration Data (DCD)

Upon reset, the Chip uses the default register values for all peripherals in the system. However, these settings typically are not ideal for achieving optimal system performance and there are even some peripherals that must be configured before they can be used.

The DCD is configuration information contained in a Program Image, external to the ROM, that the ROM interprets to configure various peripherals on the Chip.

For example, some components such as DDR require some sequence of register programming as part of configuration before it is ready to be used. The DCD feature can be used to program the MMDC registers to the optimal settings.

The ROM determines the location of the DCD table based on information located in the Image Vector Table (IVT). See Image Vector Table and Boot Data for more details. The DCD table shown below is a big endian byte array of the allowable DCD commands. The maximum size of the DCD is limited to 1768 bytes.

NOTE

ROM code does not allow DCD running once ROM jumps to boot image (even for plug-in scenario).

Figure 35-18. DCD Data Format

The DCD header is 4 bytes with the following format:

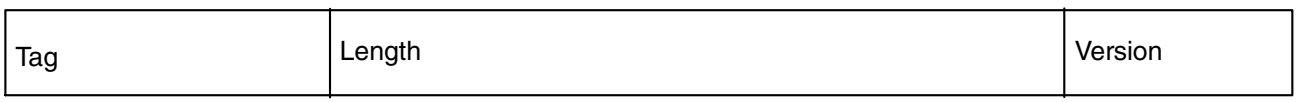


Figure 35-19. DCD Header Format

where:

```
Tag: A single byte field set to 0xD2
Length: a two byte field in big endian format containing the overall length of the DCD,
in bytes, including the header
Version: A single byte field set to 0x41
```

35.2.6.2.1 Write Data Command

The Write Data Command is used to write a list of given 1-, 2- or 4-byte values or bitmasks to a corresponding list of target addresses.

The format of Write Data Command, again a big endian byte array, is shown in the table below.

Table 35-24. Write Data Command Format

Tag	Length	Parameter
Address		
Value/Mask		
[Address]		

Table continues on the next page...

Table 35-24. Write Data Command Format (continued)

[Value/Mask]
...
[Address]
[Value/Mask]

where:

```
Tag: A single byte field set to 0xCC
Length: A two byte field in big endian format containing the length of the Write Data
Command, in bytes, including the header
Address: target address to which data should be written
Value/Mask: data value or bitmask to be written to preceding address
```

The Parameter field is a single byte divided into bitfields as follows:

Table 35-25. Write Data Command Parameter field

7	6	5	4	3	2	1	0
flags					bytes		

where

```
bytes: width of target locations in bytes. Either 1, 2 or 4
flags: control flags for command behavior.
Data Mask = bit 3: if set, only specific bits may be overwritten at target address
(otherwise all bits may be overwritten)
Data Set = bit 4: if set, bits at the target address overwritten with this flag
(otherwise it is ignored)
```

One or more target address and value/bitmask pairs can be specified. The same bytes and flags parameters apply to all locations in the command.

When successful, this command writes to each target address in accordance with the flags as follows:

Table 35-26. Interpretation of Write Data Command Flags

"Mask"	"Set"	Action	Interpretation
0	0	*address = val_msk	Write value
0	1	*address = val_msk	Write value
1	0	*address &= ~val_msk	Clear bitmask
1	1	*address |= val_msk	Set bitmask

NOTE

If any of the target addresses do not have the same alignment as the data width indicated in the parameter field, none of the values are written.

If any of the values is larger or any of the bitmasks is wider than permitted by the data width indicated in the parameter field, none of the values are written.

If any of the target addresses do not lie within an allowed region, none of the values are written. The list of allowable blocks and target addresses for the Chip are given below.

Table 35-27. Valid DCD Address Ranges

Address range	Start address	Last Address
MMDC registers	0x40AB0000	0x40ABFFFF
IOMUXC0 registers	0x4103D000	0x4103DFFF
IOMUXC1 registers	0x40AC0000	0x40ACFFF
IOMUXC DDR registers	0x40AD0000	0x40ADFFFF
PCC3 registers	0x40B30000	0x40B3FFFF
SCG1 registers	0x403E0000	0x403EFFFF
LPTPM5 registers	0x40260000	0x4026FFFF

35.2.6.2.2 Check Data Command

The Check Data Command is used to test for a given -1, 2- or 4-byte bitmasks from a source address.

The Check Data Command is a big endian byte array with format shown in the table below.

Table 35-28. Check Data Command Format

Tag	Length	Parameter
Address		
Mask		
[Count]		

where:

```
Tag: A single byte field set to 0xCF
Length: A two byte field in big endian format containing the length of the Check Data
Command, in bytes, including the header
Address: source address to test
Mask: bit mask to test
Count: optional poll count. If count is not specified this command will poll indefinitely
until the exit condition is met. If count = 0, this command behaves as for NOP.
```

The Parameter field is a single byte divided into bitfields as follows:

Table 35-29. Check Data Command Parameter field

7	6	5	4	3	2	1	0
flags					bytes		

where

```
bytes: width of target locations in bytes. Either 1, 2 or 4
flags: control flags for command behavior.
Data Mask = bit 3: if set, only specific bits may be overwritten at target address
(otherwise all bits may be overwritten)
Data Set = bit 4: if set, bits at the target address overwritten with this flag
(otherwise it is ignored)
```

This command polls the source address until either the exit condition is satisfied, or the poll count is reached. The exit condition is determined by the flags as follows:

Table 35-30. Interpretation of Check Data Command Flags

"Mask"	"Set"	Action	Interpretation
0	0	(*address & mask) == 0	All bits clear
0	1	(*address & mask) == mask	All bits set
1	0	(*address & mask)!= mask	Any bit clear
1	1	(*address & mask)!= 0	Any bit set

NOTE

If the source address does not have the same alignment as the data width indicated in the parameter field, the value is not read.

If the bitmask is wider than permitted by the data width indicated in the parameter field, the value is not read.

35.2.6.2.3 NOP Command

This command has no effect.

The format of NOP Command is a big endian four byte array as shown in the table below.

Table 35-31. NOP Command Format

Tag	Length	Undefined

where:

```
Tag: A single byte field set to 0xC0
Length: A two byte field in big endian containing the length of the NOP Command in bytes.
Fixed to a
```

```
value of 4.
Undefined: This byte is ignored and can be set to any value.
```

35.2.6.2.4 Unlock Command

The Unlock Command is used to prevent specific engine features being locked when exiting ROM.

The format of Unlock Command, again a big endian byte array, is shown in the table below.

Table 35-32. Unlock Command Format

Tag	Length	Eng
Value		
Value		
...		
Value		

where:

```
Tag: A single byte field set to 0xB2
Eng: Engine to be left unlocked.
Values: [optional] unlock values required by engine.
```

NOTE

This command may not be used in DCD structure if the SEC_CONFIG is configured as closed.

35.2.7 Plugin Image

The ROM supports a limited number of boot devices. When using other devices as a boot source (for example, Ethernet, CDROM, or USB), then the supported boot device must be used (typically serial ROM) as firmware to provide the missing boot drivers. Additionally, the plugin can be customized to support boot drivers, which is more flexible when doing device initialization, such as condition judging, delay assertion, or to apply custom settings to the boot device and memory system.

In addition to standard images, the chip also supports plugin images. Plugin images return execution to the ROM whereas a standard image does not.

The boot ROM detects the image type using the plugin flag of the boot data structure (see Boot Data Structure). If the plugin flag is 1, then the ROM uses the image as a plugin function. The function must initialize the boot device and copy the program image to the

final location. At the end the plugin function must return with the program image parameters. (See High level description of Boot: Boot mode and boot flow for details about boot flow).

The boot ROM authenticates the plugin image prior to running the plugin function and then authenticates the program image.

The plugin function must follow the API described below:

typedef BOOLEAN (*plugin_download_f)(void **start, size_t *bytes, UINT32 *ivt_offset);

ARGUMENTS PASSED:

- start - Image load address on exit.
- bytes - Image size on exit.
- ivt_offset - Offset in bytes of the IVT from the image start address on exit.

RETURN VALUE:

- 1 - on success
- 0 - on failure

35.2.8 Serial Downloader

The Serial Downloader provides a means to download a Program Image to the chip over USB serial connection. This feature is available on A7 ROM only.

In this mode the ROM programs WDOG1 for a time-out specified by fuse WDOG Time-out Select (See the attached fusemap for details) if WDOG_ENABLE eFuse is 1 and continuously polls for USB connection. If no activity is found on USBOTG2 and the watchdog timer expires, the Arm core is reset.

NOTE

After being loaded, the downloaded image is responsible to manage the watchdog resets properly.

Figure 35-1 shows USB boot flow.

35.2.8.1 USB

USB support is composed of the USBO2 (USB OTG core controller, compliant with the USB 2.0 specification) and the USBPHY (HS USB transceiver).

The ROM supports the USB OTG port for boot purposes. The other USB ports on the chip are not supported for boot purposes.

The USB Driver is implemented as a USB HID class. A collection of 4 HID reports are used to implement SDP protocol for data transfers as described in Table 35-33.

Table 35-33. USB HID Reports

Report ID (first byte)	Transfer Endpoint	Direction	Length	Description
1	control OUT	Host to device	17 bytes	SDP command from host to device
2	control OUT	Host to device	Up to 1025 bytes	Data associated with report 1 SDP command
3	interrupt	Device to host	5 bytes	HAB security configuration. Device sends 0x12343412 in closed mode and 0x56787856 in open mode.
4	interrupt	Device to host	Up to 65 bytes	Data in response to SDP command in report 1

35.2.8.1.1 USB Configuration Details

The USB OTG function device driver supports a high speed (HS for UTMI) non-stream mode with a maximal packet size of 512 B and a low-level USB OTG function.

The VID/PID and strings for USB device driver are listed in the table below.

Table 35-34. VID/PID and Strings for USB Device Driver

Descriptor	Value
VID	Defined by fuses USB_VID[15:0] (NXP vendor ID: 0x1FC9, if fuse not blown)
PID	Defined by fuses USB0_PID[15:0], 0x0126 if fuse not blown
String Descriptor1 (manufacturer)	NXP Semiconductor Inc.
String Descriptor2 (product)	S Blank SE Blank NS Blank
String Descriptor4	NXP Flash
String Descriptor5	NXP Flash

35.2.8.1.2 IOMUX Configuration for USB

The interface signals of the UTMI PHY are not configured in the IOMUX. The UTMI PHY interface uses dedicated contacts on the IC. See the chip data sheet for details.

35.2.8.2 Serial Download protocol

The 16 byte SDP command from host to device is sent using HID report 1.

The table below describes 16 byte SDP command data structure:

Table 35-35. 16 Byte SDP Command Data Structure

BYTE Offset	Size	Name	Description
0	2	COMMAND TYPE	The following commands are supported for ROM: • 0x0101 READ_REGISTER • 0x0202 WRITE_REGISTER • 0x0404 WRITE_FILE • 0x0505 ERROR_STATUS • 0x0A0A DCD_WRITE • 0x0B0B JUMP_ADDRESS • 0x0C0C SKIP_DCD_HEADER
2	4	ADDRESS	Only relevant for following commands: READ_REGISTER, WRITE_REGISTER, WRITE_FILE, DCD_WRITE, and JUMP_ADDRESS. For READ_REGISTER and WRITE_REGISTER commands, this field is address to a register. For WRITE_FILE and JUMP_ADDRESS commands, this field is an address to internal or external memory address.
6	1	FORMAT	Format of access, 0x8 for 8-bit access, 0x10 for 16-bit and 0x20 for 32-bit access. Only relevant for READ_REGISTER and WRITE_REGISTER commands.
7	4	DATA COUNT	Size of data to read or write. Only relevant for WRITE_FILE, READ_REGISTER, WRITE_REGISTER and DCD_WRITE commands. For WRITE_FILE and DCD_WRITE commands DATA COUNT is in byte units.
11	4	DATA	Value to write. Only relevant for WRITE_REGISTER command.
15	1	RESERVED	Reserved

35.2.8.2.1 SDP Command

SDP commands are decribed in the following sections.

35.2.8.2.1.1 READ REGISTER

The transaction for command READ_REGISTER consists of following reports: Report1 for command, Report3 for security configuration and Report4 for response or register value.

The register to read is specified in ADDRESS field of SDP command. First device sends Report3 with security configuration followed by Report4 with bytes read at given address. If count is greater than 64 then multiple reports with report id 4 are sent until entire data requested by host is sent. The STATUS is either 0x12343412 for closed parts and 0x56787856 for open or field return parts.

Report1, Command, Host to Device:

1	Valid values for READ_REGISTER COMMAND, ADDRESS, FORMAT, DATA_COUNT

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

Report4, Response, Device to Host: first response report

4	Register Value

ID 4 bytes of data containing register value. If number of bytes requested is less than 4 then remaining bytes should be ignored by host.

Multiple reports of report id 4 are sent until entire data requested is sent

Report4, Response, Device to Host: Last response report

4	Register Value

ID 64 bytes of data containing register value. If number of bytes requested is less than 64 then remaining bytes should be ignored by host.

35.2.8.2.1.2 WRITE REGISTER

The transaction for command WRITE_REGISTER consists of the following reports: Report1 for command, Report3 for security configuration and Report4 for write status.

Host sends Report1 with WRITE_REGISTER command. The register to write is specified in ADDRESS field of SDP command of Report1, with FORMAT field set to data type (number of bits to write 8, 16 or 32) and value to write in DATA field of SDP command. Device writes the DATA to register address and returns WRITE_COMPLETE code using Report4 and security configuration using Report3 to complete the transaction.

Report1, Command, Host to Device:

1	Valid values for WRITE_REGISTER COMMAND, ADDRESS, FORMAT, DATA_COUNT and DATA

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	WRITE_COMPLETE (0x128A8A12) status

ID 64 bytes data with first 4 bytes to indicate write is completed with code 0x128A8A12. On failure device will report HAB error status.

NOTE

ROMs is no longer in the SDP read white list.

35.2.8.2.1.3 WRITE_FILE

The transaction for command WRITE_FILE consists of following reports: Report1 for command-phase, Report2 for data-phase, Report3 for hab mode and Report4 to indicate data received in full.

The size of each Report2 is limited to 1024 bytes (limitation of USB HID protocol) hence multiple Report2 packets will be sent by host in data phase until entire data is transferred to device. Once entire data (DATA_COUNT bytes) is received then device sends report 3 with hab mode and report 4 with 0x88888888, indicating file download completed.

Report1, Host to Device:

1	Valid values for WRITE_FILE COMMAND, ADDRESS, DATA_COUNT

ID 16 byte SDP Command

==========Optional Begin==========

Host sends ERROR_STATUS command to query if HAB rejected the address

========== Optional End==========

Report2, Host to Device:

2	File data

ID Max 1024 bytes data per report

Report2, Host to Device:

2	File data

ID Max 1024 bytes data per report

Report3, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	COMPLETE (0x88888888) status

ID 64 bytes data with first 4 bytes to indicate file download has completed with code 0x88888888. On failure device will report HAB error status.

35.2.8.2.1.4 ERROR_STATUS

The transaction for SDP command ERROR_STATUS consists of three reports.

Report1 is used by host to send the command; device sends global error status in 4 bytes of Report4 after returning security configuration in Report3. When device receives ERROR_STATUS command it will return global error status that is updated for each command. This command is useful to find out if last command resulted in device error or succeeded.

Report1, Command, Host to Device:

1	ERROR_STATUS COMMAND

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	4 bytes Error status

ID first 4 bytes status in 64 bytes report 4

35.2.8.2.1.5 DCD WRITE

The SDP command DCD_WRITE is used by host to send multiple register writes in one shot. This command is provided to speed up the process of programming register writes such as to configure external RAM device.

The command goes with Report1 from host with COMMAND TYPE set to DCD_WRITE, ADDRESS which is used for temporary location of DCD data and DATA_COUNT to number of bytes sent in data out phase. In data phase host sends data for number of registers using Report2. Device completes the transaction with Report3 indicating security configuration and report 4 with WRITE_COMPLETE code 0x12828212.

Report1, Command, Host to Device:

1	DCD_WRITE COMMAND, ADDRESS, DATA_COUNT

ID 16 byte SDP Command

Report2, Data, Host to Device:

2	DCD binary data

ID Max 1024 bytes per report

Report3, Response, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	WRITE_COMPLETE (0x128A8A12) status

ID 64 bytes report with first 4 bytes to indicate write is completed with code 0x128A8A12. On failure device will report HAB error status.

See Device Configuration Data (DCD) for DCD format description.

35.2.8.2.1.6 SKIP_DCD_HEADER

The SDP command SKIP_DCD_HEADER is used by host to inform device skip the DCD configuration within download image.

Normally, if the download image should be run on DDR, the DCD configuration data should be built-in the image. In case the host has issued DCD_WRITE to push DCD configuration data to device for DDR initialization, the image with DCD information will cause ROM to initialize DDR twice and may cause initialization processing hung.

The SKIP_DCD_HEADER command informs the device to skip the DCD configuration with the download image, and avoid this issue.

This command should typically be sent after JUMP_ADDRESS. The command is sent by host in command-phase of transaction using Report1, there is no data phase for this command. Device completes the transaction with Report3 indicating security configuration and Report4 with OK_ACK code 0x900DD009.

Report1, Command, Host to Device:

1	SKIP_DCD_HEADER

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

Report4, Response, Device to Host:

4	OK_ACK (0x900DD009)

35.2.8.2.1.7 JUMP ADDRESS

The SDP command JUMP_ADDRESS will be the last command host can send to the device, after this command device will jump to the address specified in the ADDRESS field of SDP command and start executing.

This command should typically follow after WRITE_FILE command. The command is sent by host in command-phase of transaction using Report1, there is no data phase for this command but the device sends status Report3 to complete the transaction. And if HAB authentication fails, then it will also send Report4 with HAB error status.

Report1, Command, Host to Device:

1	JUMP_ADDRESS COMMAND, ADDRESS

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration

ID 4 bytes status

This report is sent by device only in case of an error jumping to the given address, device reports error in Report4, Response, Device to Host:

4	4 bytes HAB error status

ID 4 bytes status, 64 bytes report length

35.2.9 SD/MMC Manufacture Mode

When internal boot and recover boot (if enabled) failed, and the "Disable SDMMC Manufacture mode" fuse bit (DISABLE_SDMMC_MFG) isn't set, boot will go to SD/MMC manufacture mode before serial download mode. In manufacture mode, one bit bus width is used despite of the fuse setting. This mode will be skipped if Boot_Mode=01(Serial Download Mode), irrespective of the "Disable SDMMC Manufacture mode" fuse value.

In manufacture mode, SD or MMC card will be scanned on uSDHC0 and uSDHC1. If card is detected and valid boot image is found in card, then the boot image will be loaded and then executed. Pad of SD1_CD is used to detect whether card is inserted.

By default, SD/MMC manufacture mode is enabled. Blow the fuse of DISABLE_SDMMC_MFG to disable it.

NOTE

Secondary boot is not supported on SD/MMC manufacture mode.

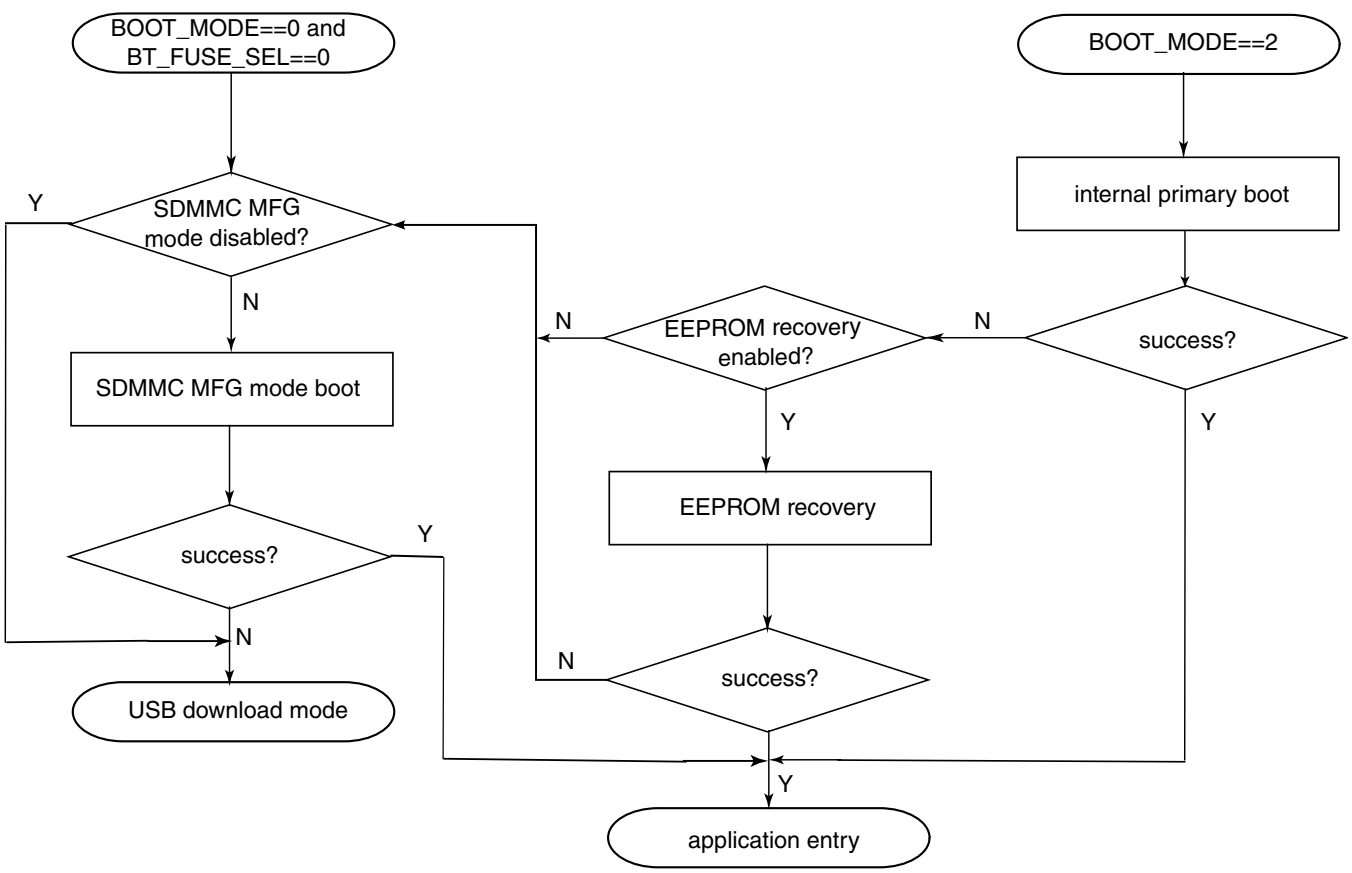


Figure 35-20. SD/MMC Manufacture boot flow

35.2.10 High Assurance Boot (HAB)

The High Assurance Boot (HAB) component of the ROM protects against the potential threat of attackers modifying areas of code or data in programmable memory to make it behave in an incorrect manner. The HAB also prevents attempts to gain access to features which should not be available.

The integration of the HAB feature with the ROM code ensures that the chip does not enter an operational state if the existing hardware security blocks have detected a condition that may be a security threat or areas of memory deemed to be important have been modified. The HAB uses RSA digital signatures to enforce these policies.

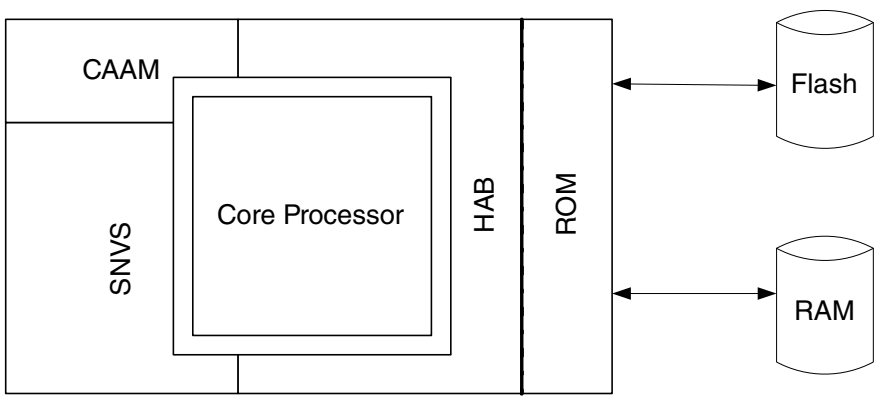


Figure 35-21. Secure Boot Components

NOTE

NXP provides a reference Code Signing Tool (CST) for key generation, certificate generation and code signing for use with the HAB library. The CST can be found by searching for "IMX_CST_TOOL" at http://www.nxp.com.

NOTE

For further details on making use of the secure boot feature using HAB, contact your local NXP representative.

35.2.10.1 HAB API Vector Table Addresses

For devices that perform a secure boot, the HAB library may be called by boot stages that execute after ROM code.

The RVT table contains the pointers to the HAB API functions and is located at 0x00000100 for A7 ROM and 0x1C000500 for M4 ROM.

NOTE

For additional information on secure boot including the HAB API, contact your local NXP representative.

35.2.11 Resume support in ROM

When resuming (exiting from VLLS mode), A7 core or M4 core will be re-powered and fetches instruction from the ROM code. ROM code then resumes the flow. The M4 ROM resume flow is as described as follows:

1. M4 ROM gets started and does the initialization job (related to HAB/security).
2. If SIM_DGO_GP1 is 0, go to step 4, otherwise clear SIM_DGO_GP1 register and then go to step 3.
3. If valid resuming entry be provisioned in SIM_DGO_GP1 (within 0x1FFD0000-0x20007FFF), M4 ROM jumps to this entry, otherwise issues WDOG reset.
4. Do the normal boot flow.

A7 ROM resume flow is described as follows:

1. A7 ROM gets started.
2. If SIM_DGO_GP3 is 0, go to step 4, otherwise clear SIM_DGO_GP3 register and then go to step 3.
3. If valid resuming entry be provisioned in SIM_DGO_GP3 (within 0x1FFD0000-0x20007FFF), or within OCRAM/QSPI/DDR, A7 ROM jumps to this entry, otherwise issues WDOG reset.
4. Do the normal boot flow.

NOTE

Even for resuming, M4 ROM will do some initialization (mainly related to HAB/security) and thus will take last 32 KB of TCMU.

NOTE

M4 resume entry must be in M4 ROM Free Area (0x1FFD0000-0x20007FFF). For A7 ROM, valid resume entry

is within M4 ROM Free Area, OCRAM, QSPI and DDR. Given that M4's involvement is necessary to wake-up A7 from VLLS, M4 image always has chance to check A7's resume entry in SIM_DGO_GP3 to make sure it is in the range as per expectation. For example, in some scenarios, it is preferred that the entry is in internal memory (M4 TCM or A7 OCRAM) for security concerns.

Chapter 36
Signal Multiplexing

36.1 Introduction

To optimize functionality in small packages, pins have several functions available via signal multiplexing. This chapter illustrates which of this device's signals are multiplexed on which external pin.

The Input/Output Multiplexing Controller (IOMUXC) controls which signal is present on the external pin. Please see the attached spreadsheet to know how the pins available on this device are configured. The available device packages and pinout diagrams are also provided in the attached spreadsheet.

36.2 Signal multiplexing constraints

1. A given peripheral function must be assigned to a maximum of one package pin. Do not program the same function to more than one pin.
2. To ensure the best signal timing for a given peripheral's interface, choose the pins in closest proximity to each other.

Chapter 37
Input/Output Multiplexing Controller (IOMUXC)

37.1 Chip-specific IOMUXC information

Table 37-1. Reference links to related information

Topic	Related module	Reference
Full description	IOMUXC	IOMUXC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

37.1.1 Input/Output Multiplexing Controller (IOMUXC)

The IOMUXC, working in conjunction with the IOMUX, enables the chip to share one pad for multiple signals from different peripheral interfaces. This pad sharing mechanism is done by multiplexing the pad's input and output signals. For each pad, there are up to 16 multiplexing options, which are called ALT modes. The IOMUX consists of combinatorial logic combined from several basic IOMUX cells. Each basic IOMUX cell handles one pad signal's multiplexing.

A peripheral's input signal may come from multiple pads. In order to avoid an input signal enabled on multiple locations the IOMUXC also control input multiplexing logic.

Every peripheral signal require a specific pad setting, such as pull-up or drive-strength. The IOMUXC also controls the pads setting parameters and digital filter functions of the pad.

i.MX 7ULP has three IOMUXC instances: IOMUXC0 for M4 ports, IOMUXC1 for A7 ports and IOMUXC DDR for DDR interface.

Table 37-2. IOMUXC configuration

Parameter	Description
Name	IOMUXC
Instances	3
Configurable features	IOMUXC0: Port A, B with Digital Filter enabled. IOMUXC1: Port C, D, E, F IOMUXC DDR: DDR & HSIC interfaces
Interface speed	NA
External I/O pins	NA

37.2 Overview

The IOMUX Controller (IOMUXC), together with the IOMUX, enables the device to share one pad to several functional blocks. The sharing is done by multiplexing the pad input/output signals. Every module requires a specific pad setting (such as pull up, keeper, and so on). The pad settings parameters are controlled by the IOMUXC. The IOMUX consists only of combinatorial logic combined from several basic iomux cells. Each basic iomux cell handles only one pad signal's muxing.

NOTE

DSE for DDR pads needs to be configured for highest speed (3'b111) for proper operation of DDR transactions.

37.3 Memory map and register definition

IOMUXC0 memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D000	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA0)	32	R/W	0000_0000h	37.3.1/1533
4103_D004	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA1)	32	R/W	0000_0000h	37.3.1/1533

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D008	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA2)	32	R/W	0000_0000h	37.3.1/1533
4103_D00C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA3)	32	R/W	0000_0000h	37.3.1/1533
4103_D010	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA4)	32	R/W	0000_0000h	37.3.1/1533
4103_D014	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA5)	32	R/W	0000_0000h	37.3.1/1533
4103_D018	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA6)	32	R/W	0000_0000h	37.3.1/1533
4103_D01C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA7)	32	R/W	0000_0000h	37.3.1/1533
4103_D020	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA8)	32	R/W	0000_0000h	37.3.1/1533
4103_D024	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA9)	32	R/W	0000_0000h	37.3.1/1533
4103_D028	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA10)	32	R/W	0000_0000h	37.3.1/1533
4103_D02C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA11)	32	R/W	0000_0000h	37.3.1/1533
4103_D030	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA12)	32	R/W	0000_0000h	37.3.1/1533
4103_D034	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA13)	32	R/W	0000_0000h	37.3.1/1533
4103_D038	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA14)	32	R/W	0000_0000h	37.3.1/1533
4103_D03C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA15)	32	R/W	0000_0000h	37.3.1/1533
4103_D040	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA16)	32	R/W	0000_0000h	37.3.1/1533
4103_D044	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA17)	32	R/W	0000_0000h	37.3.1/1533
4103_D048	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA18)	32	R/W	0000_0000h	37.3.1/1533
4103_D04C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA19)	32	R/W	0000_0000h	37.3.1/1533
4103_D050	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA20)	32	R/W	0000_0000h	37.3.1/1533
4103_D054	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA21)	32	R/W	0000_0000h	37.3.1/1533
4103_D058	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA22)	32	R/W	0000_0000h	37.3.1/1533
4103_D05C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA23)	32	R/W	0000_0000h	37.3.1/1533

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D060	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA24)	32	R/W	0000_0000h	37.3.1/1533
4103_D064	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA25)	32	R/W	0000_0000h	37.3.1/1533
4103_D068	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA26)	32	R/W	0000_0A03h	37.3.2/1536
4103_D06C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA27)	32	R/W	0000_0A00h	37.3.3/1539
4103_D070	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA28)	32	R/W	0000_0A03h	37.3.2/1536
4103_D074	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA29)	32	R/W	0000_0A02h	37.3.4/1542
4103_D078	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA30)	32	R/W	0000_0A03h	37.3.2/1536
4103_D07C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTA31)	32	R/W	0000_0000h	37.3.5/1545
4103_D080	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB0)	32	R/W	0000_0000h	37.3.5/1545
4103_D084	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB1)	32	R/W	0000_0000h	37.3.5/1545
4103_D088	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB2)	32	R/W	0000_0000h	37.3.5/1545
4103_D08C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB3)	32	R/W	0000_0000h	37.3.5/1545
4103_D090	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB4)	32	R/W	0000_0002h	37.3.6/1548
4103_D094	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB5)	32	R/W	0000_0002h	37.3.6/1548
4103_D098	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB6)	32	R/W	0000_0000h	37.3.7/1551
4103_D09C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB7)	32	R/W	0000_0000h	37.3.7/1551
4103_D0A0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB8)	32	R/W	0000_0000h	37.3.7/1551
4103_D0A4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB9)	32	R/W	0000_0000h	37.3.7/1551
4103_D0A8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB10)	32	R/W	0000_0000h	37.3.7/1551
4103_D0AC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB11)	32	R/W	0000_0000h	37.3.7/1551
4103_D0B0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB12)	32	R/W	0000_0000h	37.3.7/1551
4103_D0B4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB13)	32	R/W	0000_0000h	37.3.7/1551

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D0B8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB14)	32	R/W	0000_0000h	37.3.7/1551
4103_D0BC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB15)	32	R/W	0000_0000h	37.3.7/1551
4103_D0C0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB16)	32	R/W	0000_0000h	37.3.7/1551
4103_D0C4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB17)	32	R/W	0000_0000h	37.3.7/1551
4103_D0C8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB18)	32	R/W	0000_0000h	37.3.7/1551
4103_D0CC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_PTB19)	32	R/W	0000_0000h	37.3.7/1551
4103_D100	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI0_IPP_IND_LPSPI_PCS0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D104	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI0_IPP_IND_LPSPI_PCS1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D108	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI0_IPP_IND_LPSPI_PCS2_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D10C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI0_IPP_IND_LPSPI_PCS3_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D110	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI0_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D114	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI0_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D118	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI0_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D11C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI1_IPP_IND_LPSPI_PCS0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D120	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI1_IPP_IND_LPSPI_PCS1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D124	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI1_IPP_IND_LPSPI_PCS2_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D128	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI1_IPP_IND_LPSPI_PCS3_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D12C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI1_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D130	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI1_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D134	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPSPI1_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D138	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM0_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D13C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM0_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D140	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM0_IPP_IND_LPTPM_CH2_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D144	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM0_IPP_IND_LPTPM_CH3_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D148	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM0_IPP_IND_LPTPM_CH4_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D14C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM0_IPP_IND_LPTPM_CH5_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D150	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM1_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D154	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM1_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D158	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM2_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D15C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM2_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D160	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM3_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D164	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM3_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D168	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM3_IPP_IND_LPTPM_CH2_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D16C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM3_IPP_IND_LPTPM_CH3_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D170	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM3_IPP_IND_LPTPM_CH4_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D174	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM3_IPP_IND_LPTPM_CH5_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D178	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C0_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D17C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C0_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D180	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C0_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D184	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C1_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D188	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C1_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D18C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C1_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D190	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C2_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D194	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C2_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D198	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C2_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D19C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C3_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1A0	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C3_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D1A4	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPI2C3_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1A8	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM0_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1AC	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM1_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1B0	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM3_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1B4	N_SELECT_INPUT_DAISY_Register (IOMUXC0_PCC_AIPS0_IPP_IND_EXTCLK55_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1B8	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI0_IPP_IND_SAI_RXBCLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1BC	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI0_IPP_IND_SAI_RXSYNC_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1C0	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI0_IPP_IND_SAI_TXBCLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1C4	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI0_IPP_IND_SAI_TXSYNC_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1C8	N_SELECT_INPUT_DAISY_Register (IOMUXC0_PCC_AIPS1_IPP_IND_EXTCLK42_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1CC	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_RXBCLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1D0	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_RXSYNC_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1D4	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_TXBCLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1D8	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_TXSYNC_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1DC	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI0_IPP_IND_SAI_RXDATA0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D1E0	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI0_IPP_IND_SAI_RXDATA1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1E4	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_RXDATA0_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1E8	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_RXDATA1_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1EC	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_RXDATA2_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1F0	N_SELECT_INPUT_DAISY_Register (IOMUXC0_SAI1_IPP_IND_SAI_RXDATA3_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1F4	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPTPM2_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1F8	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART0_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D1FC	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART0_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D200	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART0_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D204	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART1_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D208	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART1_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D20C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART1_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D210	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART2_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D214	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART2_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D218	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART2_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554

Table continues on the next page...

IOMUXC0 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4103_D21C	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART3_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D220	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART3_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D224	N_SELECT_INPUT_DAISY_Register (IOMUXC0_LPUART3_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D228	N_SELECT_INPUT_DAISY_Register (IOMUXC0_D_IP_EWM_SYN_EWM_IN_SELECT_INPUT)	32	R/W	0000_0000h	37.3.8/1554
4103_D294	SW_MUX_CTL_PAD_RESET0_b SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_RESET0_b)	32	R/W	0000_0023h	37.3.9/1555

37.3.1 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. See the tab PCR fields under "Bit Name" column in the attached IOMUXC spreadsheet to verify if bit applies to a specific pad.

Address: 4103_D000h base + 0h offset + (4d × i), where i=0d to 25d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					DFD					DFCS	DFE	Reserved		OBE	IBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	Reserved			MUX_MODE				Reserved	DSE	ODE	Reserved		SRE	PE	PS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–22 DFD	Digital Filter Duration Field Select one out of next values for pad: PADn 00000 **DFD_0_Disabled** — Disabled 00001 **DFD_1_Count1** — Count1 00010 **DFD_2_Count2** — Count2 00011 **DFD_3_Count3** — Count3 00100 **DFD_4_Count4** — Count4 00101 **DFD_5_Count5** — Count5 00110 **DFD_6_Count6** — Count6

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	00111 **DFD_7_Count7** — Count7 01000 **DFD_8_Count8** — Count8 01001 **DFD_9_Count9** — Count9 01010 **DFD_10_Count10** — Count10 01011 **DFD_11_Count11** — Count11 01100 **DFD_12_Count12** — Count12 01101 **DFD_13_Count13** — Count13 01110 **DFD_14_Count14** — Count14 01111 **DFD_15_Count15** — Count15 10000 **DFD_16_Count16** — Count16 10001 **DFD_17_Count17** — Count17 10010 **DFD_18_Count18** — Count18 10011 **DFD_19_Count19** — Count19 10100 **DFD_20_Count20** — Count20 10101 **DFD_21_Count21** — Count21 10110 **DFD_22_Count22** — Count22 10111 **DFD_23_Count23** — Count23 11000 **DFD_24_Count24** — Count24 11001 **DFD_25_Count25** — Count25 11010 **DFD_26_Count26** — Count26 11011 **DFD_27_Count27** — Count27 11100 **DFD_28_Count28** — Count28 11101 **DFD_29_Count29** — Count29 11110 **DFD_30_Count30** — Count30 11111 **DFD_31_Count31** — Count31
21 DFCS	Digital Filter Clock Select Field Select one out of next values for pad: PADn 0 **DFCS_0_IPG_Clk** — IPG Clk 1 **DFCS_1_1Khz_CLK** — 1Khz CLK
20 DFE	Digital Filter Enable Field Select one out of next values for pad: PADn 0 **DFE_0_Disabled** — Disabled 1 **DFE_1_Enabled** — Enabled
19–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn 0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

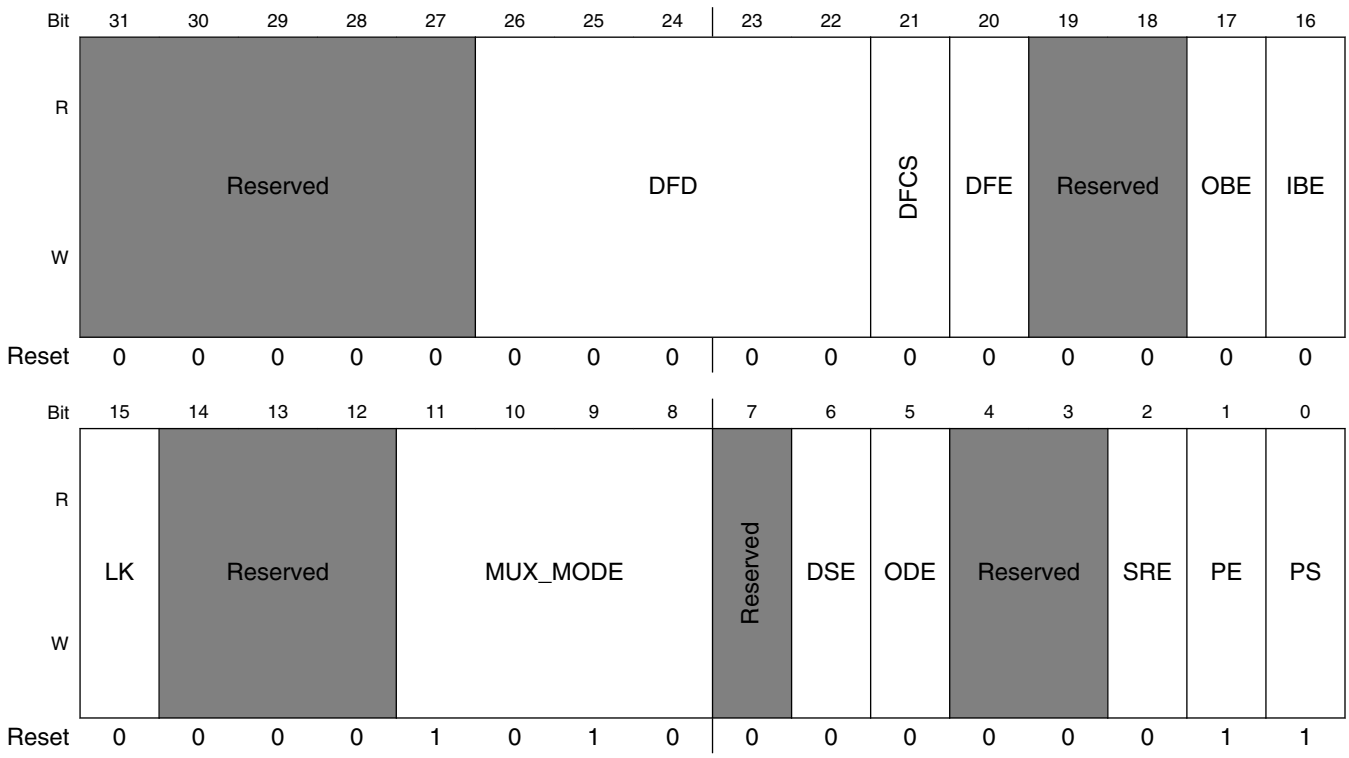
IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull Enable field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.3.2 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. Please refer to "IO Signal table" tab PCR fields under "Bit Name" column to verify if bit applies to a specific pad.

Address: 4103_D000h base + 68h offset + (8d × i), where i=0d to 2d



IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–22 DFD	Digital Filter Duration Field Select one out of next values for pad: PADn 00000 **DFD_0_Disabled** — Disabled 00001 **DFD_1_Count1** — Count1 00010 **DFD_2_Count2** — Count2 00011 **DFD_3_Count3** — Count3 00100 **DFD_4_Count4** — Count4 00101 **DFD_5_Count5** — Count5 00110 **DFD_6_Count6** — Count6

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	00111 **DFD_7_Count7** — Count7 01000 **DFD_8_Count8** — Count8 01001 **DFD_9_Count9** — Count9 01010 **DFD_10_Count10** — Count10 01011 **DFD_11_Count11** — Count11 01100 **DFD_12_Count12** — Count12 01101 **DFD_13_Count13** — Count13 01110 **DFD_14_Count14** — Count14 01111 **DFD_15_Count15** — Count15 10000 **DFD_16_Count16** — Count16 10001 **DFD_17_Count17** — Count17 10010 **DFD_18_Count18** — Count18 10011 **DFD_19_Count19** — Count19 10100 **DFD_20_Count20** — Count20 10101 **DFD_21_Count21** — Count21 10110 **DFD_22_Count22** — Count22 10111 **DFD_23_Count23** — Count23 11000 **DFD_24_Count24** — Count24 11001 **DFD_25_Count25** — Count25 11010 **DFD_26_Count26** — Count26 11011 **DFD_27_Count27** — Count27 11100 **DFD_28_Count28** — Count28 11101 **DFD_29_Count29** — Count29 11110 **DFD_30_Count30** — Count30 11111 **DFD_31_Count31** — Count31
21 DFCS	Digital Filter Clock Select Field Select one out of next values for pad: PADn 0 **DFCS_0_IPG_Clk** — IPG Clk 1 **DFCS_1_1Khz_CLK** — 1Khz CLK
20 DFE	Digital Filter Enable Field Select one out of next values for pad: PADn 0 **DFE_0_Disabled** — Disabled 1 **DFE_1_Enabled** — Enabled
19–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn 0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull Enable field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.3.3 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. See the "IO Signal Table" tab PCR fields under "Bit Name" column in the attached IOMUXC spreadsheet to verify if bit applies to a specific pad.

Address: 4103_D000h base + 6Ch offset + (0d × i), where i=0d to 0d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					DFD					DFCS	DFE	Reserved		OBE	IBE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LK	Reserved			MUX_MODE				Reserved	DSE	ODE	Reserved		SRE	PE	PS
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–22 DFD	Digital Filter Duration Field Select one out of next values for pad: PADn 00000 **DFD_0_Disabled** — Disabled 00001 **DFD_1_Count1** — Count1 00010 **DFD_2_Count2** — Count2 00011 **DFD_3_Count3** — Count3 00100 **DFD_4_Count4** — Count4 00101 **DFD_5_Count5** — Count5 00110 **DFD_6_Count6** — Count6

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	00111 **DFD_7_Count7** — Count7 01000 **DFD_8_Count8** — Count8 01001 **DFD_9_Count9** — Count9 01010 **DFD_10_Count10** — Count10 01011 **DFD_11_Count11** — Count11 01100 **DFD_12_Count12** — Count12 01101 **DFD_13_Count13** — Count13 01110 **DFD_14_Count14** — Count14 01111 **DFD_15_Count15** — Count15 10000 **DFD_16_Count16** — Count16 10001 **DFD_17_Count17** — Count17 10010 **DFD_18_Count18** — Count18 10011 **DFD_19_Count19** — Count19 10100 **DFD_20_Count20** — Count20 10101 **DFD_21_Count21** — Count21 10110 **DFD_22_Count22** — Count22 10111 **DFD_23_Count23** — Count23 11000 **DFD_24_Count24** — Count24 11001 **DFD_25_Count25** — Count25 11010 **DFD_26_Count26** — Count26 11011 **DFD_27_Count27** — Count27 11100 **DFD_28_Count28** — Count28 11101 **DFD_29_Count29** — Count29 11110 **DFD_30_Count30** — Count30 11111 **DFD_31_Count31** — Count31
21 DFCS	Digital Filter Clock Select Field Select one out of next values for pad: PADn 0 **DFCS_0_IPG_Clk** — IPG Clk 1 **DFCS_1_1Khz_CLK** — 1Khz CLK
20 DFE	Digital Filter Enable Field Select one out of next values for pad: PADn 0 **DFE_0_Disabled** — Disabled 1 **DFE_1_Enabled** — Enabled
19–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn 0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

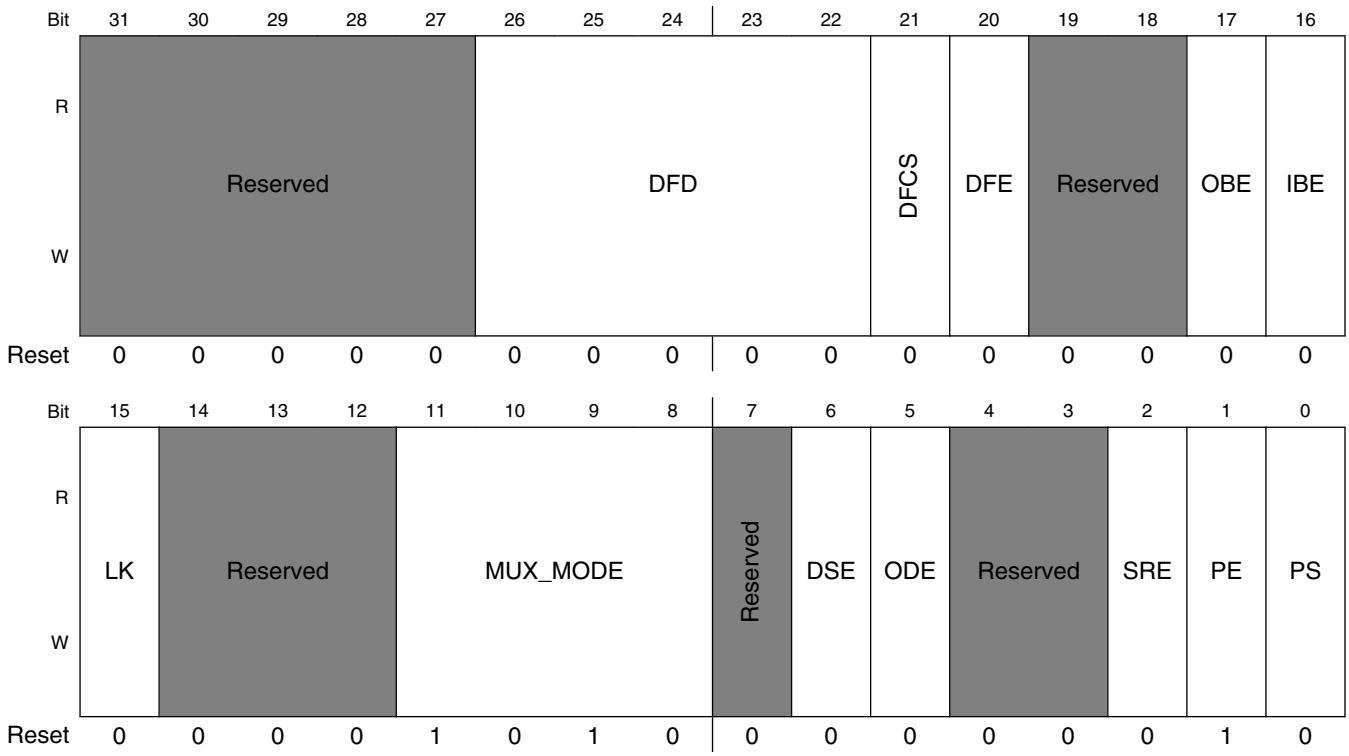
IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull Enable field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.3.4 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. Please refer to "IO Signal table" tab PCR fields under "Bit Name" column to verify if bit applies to a specific pad.

Address: 4103_D000h base + 74h offset + (0d × i), where i=0d to 0d



IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–22 DFD	Digital Filter Duration Field Select one out of next values for pad: PADn 00000 **DFD_0_Disabled** — Disabled 00001 **DFD_1_Count1** — Count1 00010 **DFD_2_Count2** — Count2 00011 **DFD_3_Count3** — Count3 00100 **DFD_4_Count4** — Count4 00101 **DFD_5_Count5** — Count5 00110 **DFD_6_Count6** — Count6

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	00111 **DFD_7_Count7** — Count7 01000 **DFD_8_Count8** — Count8 01001 **DFD_9_Count9** — Count9 01010 **DFD_10_Count10** — Count10 01011 **DFD_11_Count11** — Count11 01100 **DFD_12_Count12** — Count12 01101 **DFD_13_Count13** — Count13 01110 **DFD_14_Count14** — Count14 01111 **DFD_15_Count15** — Count15 10000 **DFD_16_Count16** — Count16 10001 **DFD_17_Count17** — Count17 10010 **DFD_18_Count18** — Count18 10011 **DFD_19_Count19** — Count19 10100 **DFD_20_Count20** — Count20 10101 **DFD_21_Count21** — Count21 10110 **DFD_22_Count22** — Count22 10111 **DFD_23_Count23** — Count23 11000 **DFD_24_Count24** — Count24 11001 **DFD_25_Count25** — Count25 11010 **DFD_26_Count26** — Count26 11011 **DFD_27_Count27** — Count27 11100 **DFD_28_Count28** — Count28 11101 **DFD_29_Count29** — Count29 11110 **DFD_30_Count30** — Count30 11111 **DFD_31_Count31** — Count31
21 DFCS	Digital Filter Clock Select Field Select one out of next values for pad: PADn 0 **DFCS_0_IPG_Clk** — IPG Clk 1 **DFCS_1_1Khz_CLK** — 1Khz CLK
20 DFE	Digital Filter Enable Field Select one out of next values for pad: PADn 0 **DFE_0_Disabled** — Disabled 1 **DFE_1_Enabled** — Enabled
19–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn 0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull Enable field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.3.5 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. Please refer to "IO Signal table" tab PCR fields under "Bit Name" column to verify if bit applies to a specific pad.

Address: 4103_D000h base + 7Ch offset + (4d × i), where i=0d to 4d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					DFD					DFCS	DFE	Reserved		OBE	IBE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LK	Reserved			MUX_MODE				Reserved	DSE	ODE	Reserved		SRE	PE	PS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–22 DFD	Digital Filter Duration Field Select one out of next values for pad: PADn 00000 **DFD_0_Disabled** — Disabled 00001 **DFD_1_Count1** — Count1 00010 **DFD_2_Count2** — Count2 00011 **DFD_3_Count3** — Count3 00100 **DFD_4_Count4** — Count4 00101 **DFD_5_Count5** — Count5 00110 **DFD_6_Count6** — Count6

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	00111 **DFD_7_Count7** — Count7 01000 **DFD_8_Count8** — Count8 01001 **DFD_9_Count9** — Count9 01010 **DFD_10_Count10** — Count10 01011 **DFD_11_Count11** — Count11 01100 **DFD_12_Count12** — Count12 01101 **DFD_13_Count13** — Count13 01110 **DFD_14_Count14** — Count14 01111 **DFD_15_Count15** — Count15 10000 **DFD_16_Count16** — Count16 10001 **DFD_17_Count17** — Count17 10010 **DFD_18_Count18** — Count18 10011 **DFD_19_Count19** — Count19 10100 **DFD_20_Count20** — Count20 10101 **DFD_21_Count21** — Count21 10110 **DFD_22_Count22** — Count22 10111 **DFD_23_Count23** — Count23 11000 **DFD_24_Count24** — Count24 11001 **DFD_25_Count25** — Count25 11010 **DFD_26_Count26** — Count26 11011 **DFD_27_Count27** — Count27 11100 **DFD_28_Count28** — Count28 11101 **DFD_29_Count29** — Count29 11110 **DFD_30_Count30** — Count30 11111 **DFD_31_Count31** — Count31
21 DFCS	Digital Filter Clock Select Field Select one out of next values for pad: PADn 0 **DFCS_0_IPG_Clk** — IPG Clk 1 **DFCS_1_1Khz_CLK** — 1Khz CLK
20 DFE	Digital Filter Enable Field Select one out of next values for pad: PADn 0 **DFE_0_Disabled** — Disabled 1 **DFE_1_Enabled** — Enabled
19–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn 0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

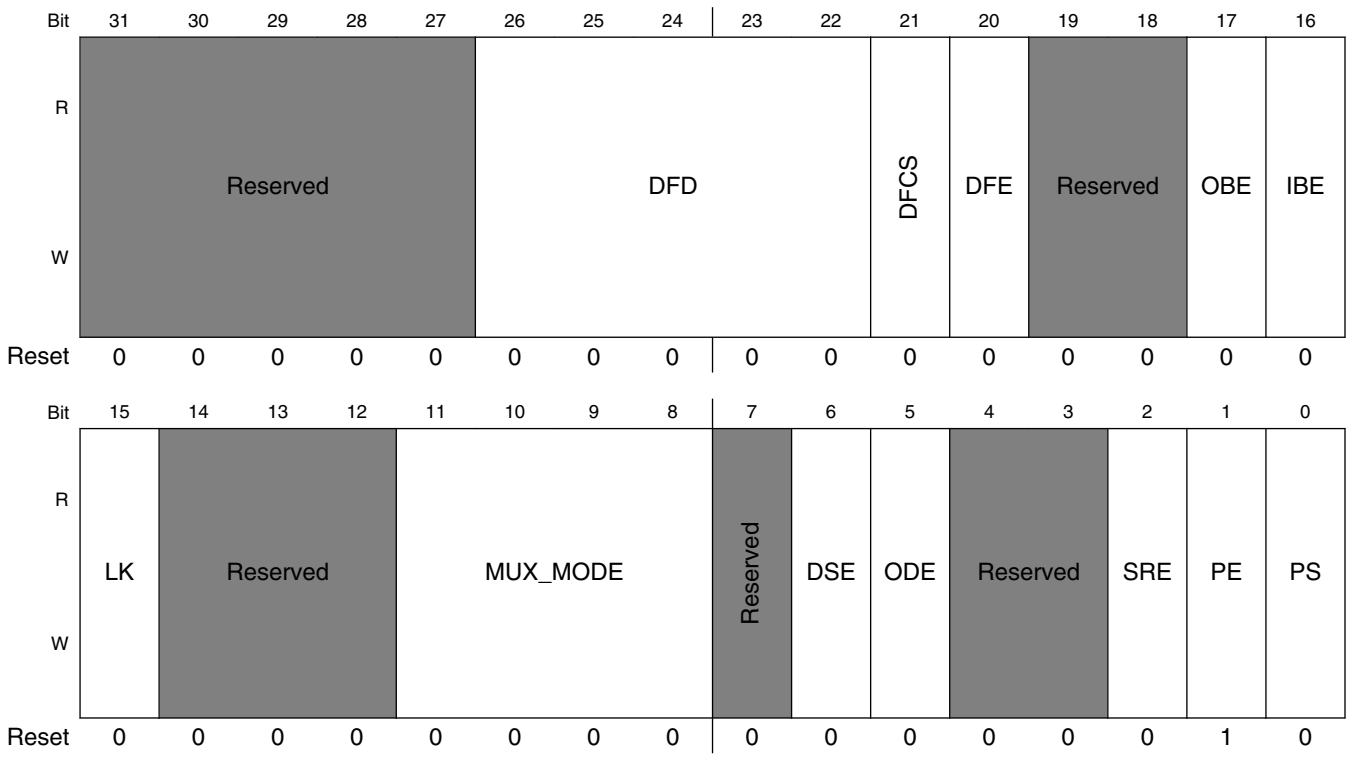
IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull Enable field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.3.6 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. Please refer to "IO Signal table" tab PCR fields under "Bit Name" column to verify if bit applies to a specific pad.

Address: 4103_D000h base + 90h offset + (4d × i), where i=0d to 1d



IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–22 DFD	Digital Filter Duration Field Select one out of next values for pad: PADn 00000 **DFD_0_Disabled** — Disabled 00001 **DFD_1_Count1** — Count1 00010 **DFD_2_Count2** — Count2 00011 **DFD_3_Count3** — Count3 00100 **DFD_4_Count4** — Count4 00101 **DFD_5_Count5** — Count5 00110 **DFD_6_Count6** — Count6

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	00111 **DFD_7_Count7** — Count7 01000 **DFD_8_Count8** — Count8 01001 **DFD_9_Count9** — Count9 01010 **DFD_10_Count10** — Count10 01011 **DFD_11_Count11** — Count11 01100 **DFD_12_Count12** — Count12 01101 **DFD_13_Count13** — Count13 01110 **DFD_14_Count14** — Count14 01111 **DFD_15_Count15** — Count15 10000 **DFD_16_Count16** — Count16 10001 **DFD_17_Count17** — Count17 10010 **DFD_18_Count18** — Count18 10011 **DFD_19_Count19** — Count19 10100 **DFD_20_Count20** — Count20 10101 **DFD_21_Count21** — Count21 10110 **DFD_22_Count22** — Count22 10111 **DFD_23_Count23** — Count23 11000 **DFD_24_Count24** — Count24 11001 **DFD_25_Count25** — Count25 11010 **DFD_26_Count26** — Count26 11011 **DFD_27_Count27** — Count27 11100 **DFD_28_Count28** — Count28 11101 **DFD_29_Count29** — Count29 11110 **DFD_30_Count30** — Count30 11111 **DFD_31_Count31** — Count31
21 DFCS	Digital Filter Clock Select Field Select one out of next values for pad: PADn 0 **DFCS_0_IPG_Clk** — IPG Clk 1 **DFCS_1_1Khz_CLK** — 1Khz CLK
20 DFE	Digital Filter Enable Field Select one out of next values for pad: PADn 0 **DFE_0_Disabled** — Disabled 1 **DFE_1_Enabled** — Enabled
19–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn 0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull Enable field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.3.7 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. Please refer to "IO Signal table" tab PCR fields under "Bit Name" column to verify if bit applies to a specific pad.

Address: 4103_D000h base + 98h offset + (4d × i), where i=0d to 13d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					DFD					DFCS	DFE	Reserved		OBE	IBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LK	Reserved			MUX_MODE				Reserved	DSE	ODE	Reserved		SRE	PE	PS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–22 DFD	Digital Filter Duration Field Select one out of next values for pad: PADn 00000 **DFD_0_Disabled** — Disabled 00001 **DFD_1_Count1** — Count1 00010 **DFD_2_Count2** — Count2 00011 **DFD_3_Count3** — Count3 00100 **DFD_4_Count4** — Count4 00101 **DFD_5_Count5** — Count5 00110 **DFD_6_Count6** — Count6

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	00111 **DFD_7_Count7** — Count7 01000 **DFD_8_Count8** — Count8 01001 **DFD_9_Count9** — Count9 01010 **DFD_10_Count10** — Count10 01011 **DFD_11_Count11** — Count11 01100 **DFD_12_Count12** — Count12 01101 **DFD_13_Count13** — Count13 01110 **DFD_14_Count14** — Count14 01111 **DFD_15_Count15** — Count15 10000 **DFD_16_Count16** — Count16 10001 **DFD_17_Count17** — Count17 10010 **DFD_18_Count18** — Count18 10011 **DFD_19_Count19** — Count19 10100 **DFD_20_Count20** — Count20 10101 **DFD_21_Count21** — Count21 10110 **DFD_22_Count22** — Count22 10111 **DFD_23_Count23** — Count23 11000 **DFD_24_Count24** — Count24 11001 **DFD_25_Count25** — Count25 11010 **DFD_26_Count26** — Count26 11011 **DFD_27_Count27** — Count27 11100 **DFD_28_Count28** — Count28 11101 **DFD_29_Count29** — Count29 11110 **DFD_30_Count30** — Count30 11111 **DFD_31_Count31** — Count31
21 DFCS	Digital Filter Clock Select Field Select one out of next values for pad: PADn 0 **DFCS_0_IPG_Clk** — IPG Clk 1 **DFCS_1_1Khz_CLK** — 1Khz CLK
20 DFE	Digital Filter Enable Field Select one out of next values for pad: PADn 0 **DFE_0_Disabled** — Disabled 1 **DFE_1_Enabled** — Enabled
19–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn 0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

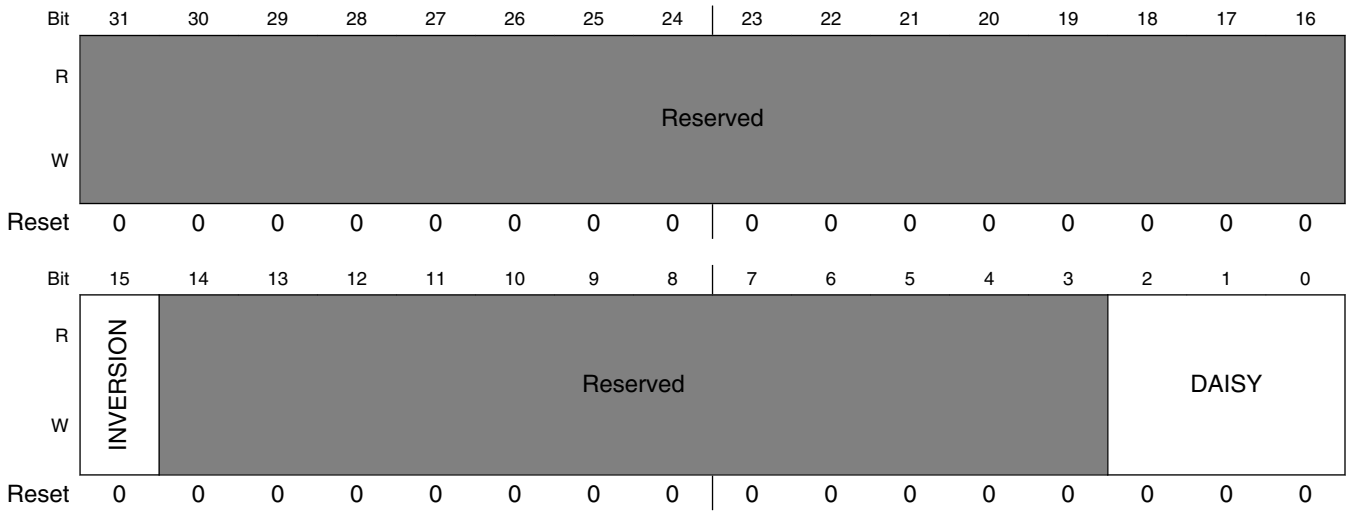
IOMUXC0_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull Enable field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.3.8 N_SELECT_INPUT_DAISY_Register (IOMUXC0_*n*_SELECT_INPUT)

This register is used to select a source pad for a specific module input function. Refer to "Input Mux" tab on IOMUX Sheet.

Address: 4103_D000h base + 100h offset + (4d × i), where i=0d to 74d



IOMUXC0_*n*_SELECT_INPUT field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 INVERSION	Controls the inversion of the pad->module input to instance 0 **DISABLE_INV** — Disable inversion. 1 **ENABLE_INV** — Enable inversion.
14–3 -	This field is reserved. Reserved
DAISY	Selects source pad for Module Input Function. Refer to "Input SSS" column from "Input Mux" tab on IOMUX Sheet.

37.3.9 SW_MUX_CTL_PAD_RESET0_b SW MUX Control Register (IOMUXC0_SW_MUX_CTL_PAD_RESET0_b)

SW_MUX_CTL Register

Address: 4103_D000h base + 294h offset = 4103_D294h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LK	Reserved								DSE	ODE	Reserved		SRE	PE	PS
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1

IOMUXC0_SW_MUX_CTL_PAD_RESET0_b field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: RESET0_b 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: RESET0_b 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: RESET0_b 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: RESET0_b 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow

Table continues on the next page...

IOMUXC0_SW_MUX_CTL_PAD_RESET0_b field descriptions (continued)

Field	Description
1 PE	Pull Enable field Select one out of next values for pad: RESET0_b 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: RESET0_b 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.4 Memory map and register definition

IOMUXC1 memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0000	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC0)	32	R/W	0000_0000h	37.4.1/1567
40AC_0004	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC1)	32	R/W	0000_0000h	37.4.1/1567
40AC_0008	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC2)	32	R/W	0000_0000h	37.4.1/1567
40AC_000C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC3)	32	R/W	0000_0000h	37.4.1/1567
40AC_0010	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC4)	32	R/W	0000_0000h	37.4.1/1567
40AC_0014	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC5)	32	R/W	0000_0000h	37.4.1/1567
40AC_0018	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC6)	32	R/W	0000_0000h	37.4.1/1567
40AC_001C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC7)	32	R/W	0000_0000h	37.4.1/1567
40AC_0020	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC8)	32	R/W	0000_0000h	37.4.1/1567
40AC_0024	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC9)	32	R/W	0000_0000h	37.4.1/1567
40AC_0028	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC10)	32	R/W	0000_0000h	37.4.1/1567
40AC_002C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC11)	32	R/W	0000_0000h	37.4.1/1567

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0030	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC12)	32	R/W	0000_0000h	37.4.1/1567
40AC_0034	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC13)	32	R/W	0000_0000h	37.4.1/1567
40AC_0038	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC14)	32	R/W	0000_0000h	37.4.1/1567
40AC_003C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC15)	32	R/W	0000_0000h	37.4.1/1567
40AC_0040	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC16)	32	R/W	0000_0000h	37.4.1/1567
40AC_0044	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC17)	32	R/W	0000_0000h	37.4.1/1567
40AC_0048	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC18)	32	R/W	0000_0000h	37.4.1/1567
40AC_004C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTC19)	32	R/W	0000_0000h	37.4.1/1567
40AC_0050	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED0)	32	R/W	0000_0000h	37.4.1/1567
40AC_0054	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED1)	32	R/W	0000_0000h	37.4.1/1567
40AC_0058	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED2)	32	R/W	0000_0000h	37.4.1/1567
40AC_005C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED3)	32	R/W	0000_0000h	37.4.1/1567
40AC_0060	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED4)	32	R/W	0000_0000h	37.4.1/1567
40AC_0064	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED5)	32	R/W	0000_0000h	37.4.1/1567
40AC_0068	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED6)	32	R/W	0000_0000h	37.4.1/1567
40AC_006C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED7)	32	R/W	0000_0000h	37.4.1/1567
40AC_0070	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED8)	32	R/W	0000_0000h	37.4.1/1567
40AC_0074	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED9)	32	R/W	0000_0000h	37.4.1/1567
40AC_0078	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED10)	32	R/W	0000_0000h	37.4.1/1567
40AC_007C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED11)	32	R/W	0000_0000h	37.4.1/1567
40AC_0080	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD0)	32	R/W	0000_0000h	37.4.1/1567
40AC_0084	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD1)	32	R/W	0000_0000h	37.4.1/1567

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0088	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD2)	32	R/W	0000_0000h	37.4.1/1567
40AC_008C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD3)	32	R/W	0000_0000h	37.4.1/1567
40AC_0090	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD4)	32	R/W	0000_0000h	37.4.1/1567
40AC_0094	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD5)	32	R/W	0000_0000h	37.4.1/1567
40AC_0098	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD6)	32	R/W	0000_0000h	37.4.1/1567
40AC_009C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD7)	32	R/W	0000_0000h	37.4.1/1567
40AC_00A0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD8)	32	R/W	0000_0000h	37.4.1/1567
40AC_00A4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD9)	32	R/W	0000_0000h	37.4.1/1567
40AC_00A8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD10)	32	R/W	0000_0000h	37.4.1/1567
40AC_00AC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTD11)	32	R/W	0000_0000h	37.4.1/1567
40AC_00B0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED12)	32	R/W	0000_0000h	37.4.1/1567
40AC_00B4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED13)	32	R/W	0000_0000h	37.4.1/1567
40AC_00B8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED14)	32	R/W	0000_0000h	37.4.1/1567
40AC_00BC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED15)	32	R/W	0000_0000h	37.4.1/1567
40AC_00C0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED16)	32	R/W	0000_0000h	37.4.1/1567
40AC_00C4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED17)	32	R/W	0000_0000h	37.4.1/1567
40AC_00C8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED18)	32	R/W	0000_0000h	37.4.1/1567
40AC_00CC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED19)	32	R/W	0000_0000h	37.4.1/1567
40AC_00D0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED20)	32	R/W	0000_0000h	37.4.1/1567
40AC_00D4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED21)	32	R/W	0000_0000h	37.4.1/1567
40AC_00D8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED22)	32	R/W	0000_0000h	37.4.1/1567
40AC_00DC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED23)	32	R/W	0000_0000h	37.4.1/1567

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_00E0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED24)	32	R/W	0000_0000h	37.4.1/1567
40AC_00E4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED25)	32	R/W	0000_0000h	37.4.1/1567
40AC_00E8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED26)	32	R/W	0000_0000h	37.4.1/1567
40AC_00EC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED27)	32	R/W	0000_0000h	37.4.1/1567
40AC_00F0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED28)	32	R/W	0000_0000h	37.4.1/1567
40AC_00F4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED29)	32	R/W	0000_0000h	37.4.1/1567
40AC_00F8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED30)	32	R/W	0000_0000h	37.4.1/1567
40AC_00FC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED31)	32	R/W	0000_0000h	37.4.1/1567
40AC_0100	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE0)	32	R/W	0000_0000h	37.4.1/1567
40AC_0104	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE1)	32	R/W	0000_0000h	37.4.1/1567
40AC_0108	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE2)	32	R/W	0000_0000h	37.4.1/1567
40AC_010C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE3)	32	R/W	0000_0000h	37.4.1/1567
40AC_0110	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE4)	32	R/W	0000_0000h	37.4.1/1567
40AC_0114	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE5)	32	R/W	0000_0000h	37.4.1/1567
40AC_0118	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE6)	32	R/W	0000_0000h	37.4.1/1567
40AC_011C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE7)	32	R/W	0000_0000h	37.4.1/1567
40AC_0120	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE8)	32	R/W	0000_0000h	37.4.1/1567
40AC_0124	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE9)	32	R/W	0000_0000h	37.4.1/1567
40AC_0128	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE10)	32	R/W	0000_0000h	37.4.1/1567
40AC_012C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE11)	32	R/W	0000_0000h	37.4.1/1567
40AC_0130	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE12)	32	R/W	0000_0000h	37.4.1/1567
40AC_0134	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE13)	32	R/W	0000_0000h	37.4.1/1567

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0138	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE14)	32	R/W	0000_0000h	37.4.1/1567
40AC_013C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTE15)	32	R/W	0000_0000h	37.4.1/1567
40AC_0140	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED32)	32	R/W	0000_0000h	37.4.1/1567
40AC_0144	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED33)	32	R/W	0000_0000h	37.4.1/1567
40AC_0148	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED34)	32	R/W	0000_0000h	37.4.1/1567
40AC_014C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED35)	32	R/W	0000_0000h	37.4.1/1567
40AC_0150	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED36)	32	R/W	0000_0000h	37.4.1/1567
40AC_0154	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED37)	32	R/W	0000_0000h	37.4.1/1567
40AC_0158	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED38)	32	R/W	0000_0000h	37.4.1/1567
40AC_015C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED39)	32	R/W	0000_0000h	37.4.1/1567
40AC_0160	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED40)	32	R/W	0000_0000h	37.4.1/1567
40AC_0164	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED41)	32	R/W	0000_0000h	37.4.1/1567
40AC_0168	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED42)	32	R/W	0000_0000h	37.4.1/1567
40AC_016C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED43)	32	R/W	0000_0000h	37.4.1/1567
40AC_0170	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED44)	32	R/W	0000_0000h	37.4.1/1567
40AC_0174	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED45)	32	R/W	0000_0000h	37.4.1/1567
40AC_0178	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED46)	32	R/W	0000_0000h	37.4.1/1567
40AC_017C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESERVED47)	32	R/W	0000_0000h	37.4.1/1567
40AC_0180	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF0)	32	R/W	0000_0000h	37.4.1/1567
40AC_0184	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF1)	32	R/W	0000_0000h	37.4.1/1567
40AC_0188	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF2)	32	R/W	0000_0000h	37.4.1/1567
40AC_018C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF3)	32	R/W	0000_0000h	37.4.1/1567

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0190	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF4)	32	R/W	0000_0000h	37.4.1/1567
40AC_0194	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF5)	32	R/W	0000_0000h	37.4.1/1567
40AC_0198	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF6)	32	R/W	0000_0000h	37.4.1/1567
40AC_019C	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF7)	32	R/W	0000_0000h	37.4.1/1567
40AC_01A0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF8)	32	R/W	0000_0000h	37.4.1/1567
40AC_01A4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF9)	32	R/W	0000_0000h	37.4.1/1567
40AC_01A8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF10)	32	R/W	0000_0000h	37.4.1/1567
40AC_01AC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF11)	32	R/W	0000_0000h	37.4.1/1567
40AC_01B0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF12)	32	R/W	0000_0000h	37.4.1/1567
40AC_01B4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF13)	32	R/W	0000_0000h	37.4.1/1567
40AC_01B8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF14)	32	R/W	0000_0000h	37.4.1/1567
40AC_01BC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF15)	32	R/W	0000_0000h	37.4.1/1567
40AC_01C0	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF16)	32	R/W	0000_0000h	37.4.1/1567
40AC_01C4	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF17)	32	R/W	0000_0000h	37.4.1/1567
40AC_01C8	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF18)	32	R/W	0000_0000h	37.4.1/1567
40AC_01CC	SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_PTF19)	32	R/W	0000_0000h	37.4.1/1567
40AC_0200	N_SELECT_INPUT DAISY Register (IOMUXC1_USDHC1_IPP_WP_ON_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0204	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO0_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0208	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO1_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_020C	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO2_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0210	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO3_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0214	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO4_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0218	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO5_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_021C	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO6_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0220	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO7_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0224	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO8_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0228	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO9_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_022C	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO10_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0230	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO11_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0234	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO12_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0238	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO13_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_023C	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO14_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0240	N_SELECT_INPUT DAISY Register (IOMUXC1_D_IP_FLEXIO_SYN1_IPP_IND_FLEXIO15_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0244	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART4_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0248	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART4_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_024C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART4_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0250	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART5_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0254	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART5_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0258	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART5_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_025C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART6_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0260	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART6_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0264	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART6_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0268	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART7_IPP_IND_LPUART_CTS_B_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_026C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART7_IPP_IND_LPUART_RXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0270	N_SELECT_INPUT DAISY Register (IOMUXC1_LPUART7_IPP_IND_LPUART_TXD_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0274	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C4_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0278	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C4_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_027C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C4_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0280	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM4_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0284	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM4_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0288	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM4_IPP_IND_LPTPM_CH2_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_028C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM4_IPP_IND_LPTPM_CH3_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0290	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM4_IPP_IND_LPTPM_CH4_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0294	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM4_IPP_IND_LPTPM_CH5_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0298	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM4_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_029C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI2_IPP_IND_LPSPI_PCS0_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02A0	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI2_IPP_IND_LPSPI_PCS1_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02A4	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI2_IPP_IND_LPSPI_PCS2_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02A8	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI2_IPP_IND_LPSPI_PCS3_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02AC	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI2_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02B0	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI2_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02B4	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI2_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02B8	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C5_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02BC	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C5_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02C0	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C5_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_02C4	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM5_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02C8	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM5_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02CC	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM5_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02D0	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM6_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02D4	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM6_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02D8	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM6_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02DC	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM7_IPP_IND_LPTPM_CH0_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02E0	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM7_IPP_IND_LPTPM_CH1_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02E4	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM7_IPP_IND_LPTPM_CH2_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02E8	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM7_IPP_IND_LPTPM_CH3_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02EC	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM7_IPP_IND_LPTPM_CH4_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02F0	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM7_IPP_IND_LPTPM_CH5_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02F4	N_SELECT_INPUT DAISY Register (IOMUXC1_LPTPM7_IPP_IND_LPTPM_CLK_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02F8	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C6_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_02FC	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C6_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569

Table continues on the next page...

IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_0300	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C6_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0304	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C7_IPP_IND_LPI2C_HREQ_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0308	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C7_IPP_IND_LPI2C_SCL_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_030C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPI2C7_IPP_IND_LPI2C_SDA_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0310	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI3_IPP_IND_LPSPI_PCS0_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0314	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI3_IPP_IND_LPSPI_PCS1_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0318	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI3_IPP_IND_LPSPI_PCS2_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_031C	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI3_IPP_IND_LPSPI_PCS3_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0320	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI3_IPP_IND_LPSPI_SCK_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0324	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI3_IPP_IND_LPSPI_SDI_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0328	N_SELECT_INPUT DAISY Register (IOMUXC1_LPSPI3_IPP_IND_LPSPI_SDO_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_032C	N_SELECT_INPUT DAISY Register (IOMUXC1_USDHC1_IPP_CARD_DET_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0330	N_SELECT_INPUT DAISY Register (IOMUXC1_USBO2_ULP1_IPP_IND_OTG_OC_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0334	N_SELECT_INPUT DAISY Register (IOMUXC1_USBO2_ULP1_IPP_IND_OTG2_OC_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_0338	N_SELECT_INPUT DAISY Register (IOMUXC1_DA_IP_HS_USB2PHY_28FDSOI_USB_ID_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569
40AC_033C	N_SELECT_INPUT DAISY Register (IOMUXC1_VIDEO_IN_IPP_IND_DE_SELECT_INPUT)	32	R/W	0000_0000h	37.4.2/1569

Table continues on the next page...

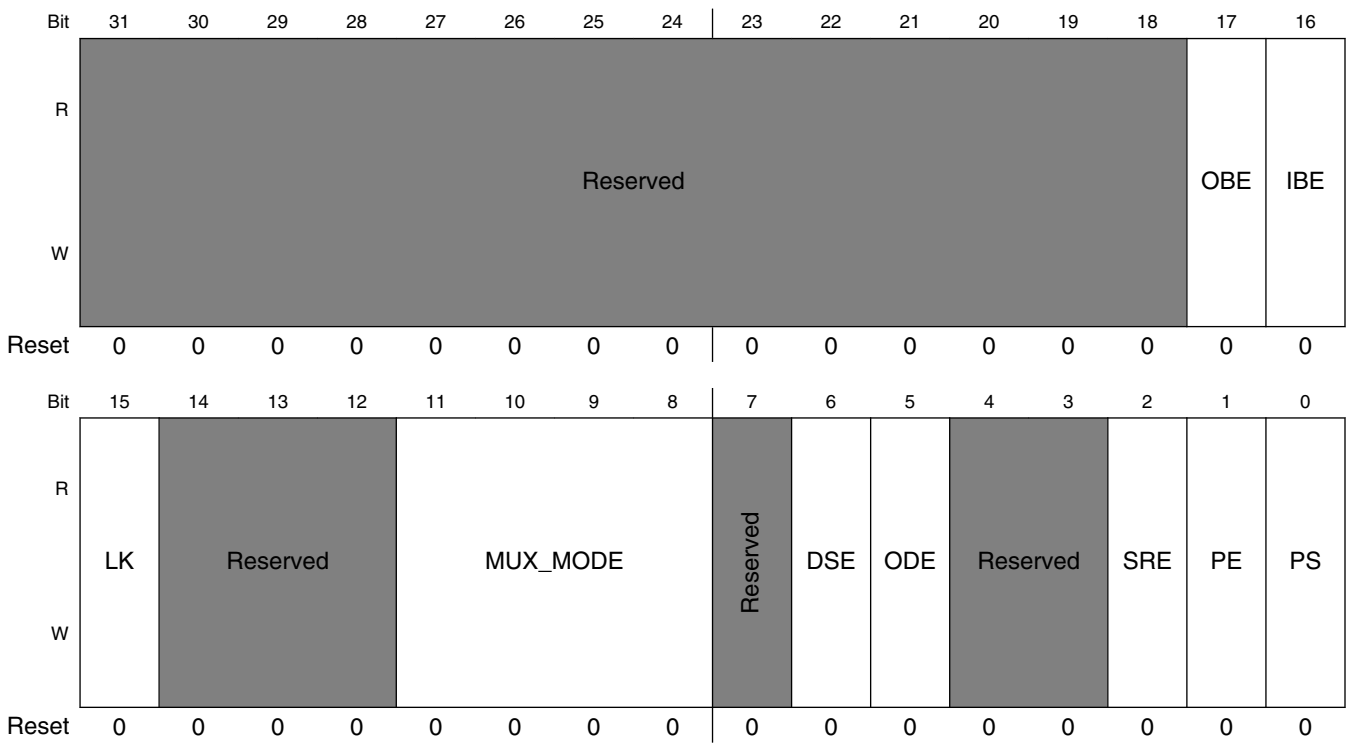
IOMUXC1 memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AC_039C	SW_MUX_CTL_PAD_RESET1_b SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESET1_b)	32	R/W	0000_0023h	37.4.3/1570

37.4.1 SW_MUX_CTL_PAD SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_*n*)

This register is used to control the settings for a PAD (bit-wise details given below) and select ALT modes using MUX_MODE bits. Not all bits are implemented in all pads. Please refer to "IO Signal table" tab PCR fields under "Bit Name" column in the attached IOMUXC spreadsheet to verify if bit applies to a specific pad.

Address: 40AC_0000h base + 0h offset + (4d × i), where i=0d to 115d



IOMUXC1_SW_MUX_CTL_PAD_*n* field descriptions

Field	Description
31–18 -	This field is reserved. Reserved
17 OBE	Output Buffer Enable Field Select one out of next values for pad: PADn

Table continues on the next page...

IOMUXC1_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
	0 **OBE_0_Disabled** — Disabled 1 **OBE_1_Enabled** — Enabled
16 IBE	Input Buffer Enable Field Select one out of next values for pad: PADn 0 **IBE_0_Disabled** — Disabled 1 **IBE_1_Enabled** — Enabled
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: PADn 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–12 -	This field is reserved. Reserved
11–8 MUX_MODE	MUX Mode Select Field. Selects one of the maximum sixteen modes available for any pad. (The field width would vary for different pads depending on different alt modes available on it). Refer to "PCR MUX_MODE" column from IOMUX sheet.
7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: PADn 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: PADn 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: PADn 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow
1 PE	Pull-up Enable Field Select one out of next values for pad: PADn 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled

Table continues on the next page...

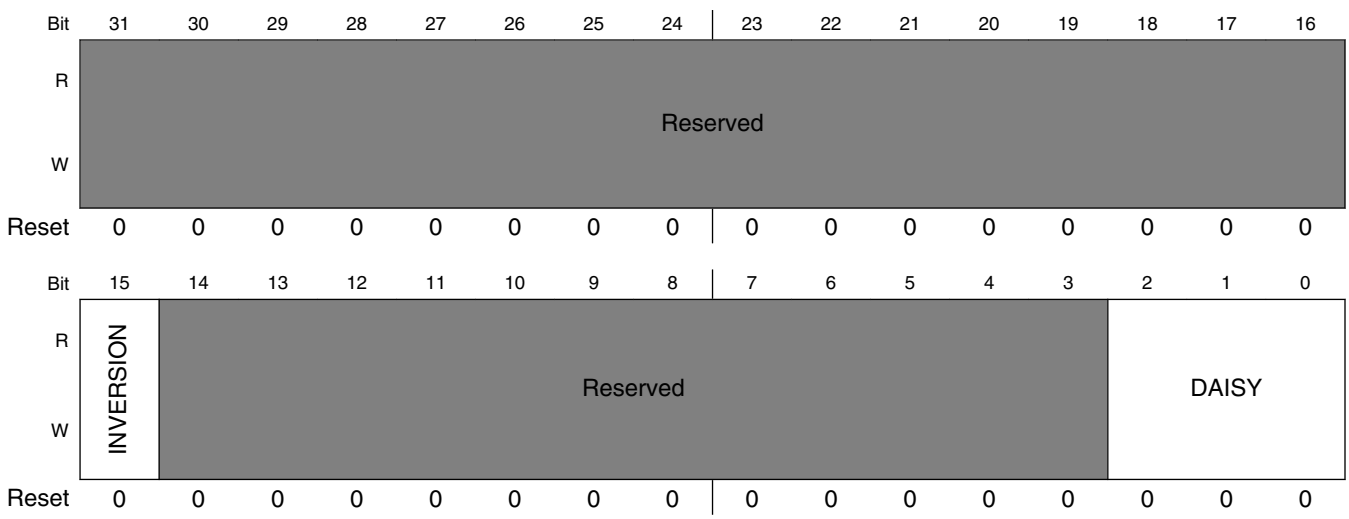
IOMUXC1_SW_MUX_CTL_PAD_*n* field descriptions (continued)

Field	Description
0 PS	Pull Select Field Select one out of next values for pad: PADn 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.4.2 N_SELECT_INPUT DAISY Register (IOMUXC1_*n*_SELECT_INPUT)

This register is used to select a source pad for a specific module input function. Refer to "Input Mux" tab on IOMUX Sheet.

Address: 40AC_0000h base + 200h offset + (4d × i), where i=0d to 79d



IOMUXC1_*n*_SELECT_INPUT field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 INVERSION	Control the inversion of the pad->module input 0 **DISABLE_INV** — Disable inversion. 1 **ENABLE_INV** — Enable inversion.
14–3 -	This field is reserved. Reserved
DAISY	Selects source pad for Module Input Function. Refer to "Input SSS" column from "Input Mux" tab on IOMUX Sheet.

37.4.3 SW_MUX_CTL_PAD_RESET1_b SW MUX Control Register (IOMUXC1_SW_MUX_CTL_PAD_RESET1_b)

SW_MUX_CTL Register

Address: 40AC_0000h base + 39Ch offset = 40AC_039Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LK	Reserved								DSE	ODE	Reserved		SRE	PE	PS
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1

IOMUXC1_SW_MUX_CTL_PAD_RESET1_b field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 LK	Lock Field When enabled, prevents writes to fields: PS, PE, SRE, ODE, DSE, MUX_MODE and LK. Select one out of next values for pad: RESET1_b 0 **LK_0_Disabled** — Disabled 1 **LK_1_Enabled** — Enabled
14–7 -	This field is reserved. Reserved
6 DSE	Drive Strength Enable Field Select one out of next values for pad: RESET1_b 0 **DSE_0_Standard** — Standard 1 **DSE_1_Hi_Drive** — Hi Drive
5 ODE	Open-drain Enable Field Select one out of next values for pad: RESET1_b 0 **ODE_0_Push_pull** — Push-pull 1 **ODE_1_Open_drain** — Open-drain
4–3 -	This field is reserved. Reserved
2 SRE	Slew Rate Enable Field Select one out of next values for pad: RESET1_b 0 **SRE_0_Standard** — Standard 1 **SRE_1_Slow** — Slow

Table continues on the next page...

IOMUXC1_SW_MUX_CTL_PAD_RESET1_b field descriptions (continued)

Field	Description
1 PE	Pull-up Enable Field Select one out of next values for pad: RESET1_b 0 **PE_0_pull_disabled** — pull disabled 1 **PE_1_pull_enabled** — pull enabled
0 PS	Pull Select Field Select one out of next values for pad: RESET1_b 0 **PS_0_pull_down** — pull-down 1 **PS_1_pull_up** — pull-up

37.5 Memory map and register definition

IOMUXC1_DDR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AD_0000	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ0)	32	R/W	0000_0000h	37.5.1/1575
40AD_0004	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ1)	32	R/W	0000_0000h	37.5.1/1575
40AD_0008	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ2)	32	R/W	0000_0000h	37.5.1/1575
40AD_000C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ3)	32	R/W	0000_0000h	37.5.1/1575
40AD_0010	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ4)	32	R/W	0000_0000h	37.5.1/1575
40AD_0014	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ5)	32	R/W	0000_0000h	37.5.1/1575
40AD_0018	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ6)	32	R/W	0000_0000h	37.5.1/1575
40AD_001C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ7)	32	R/W	0000_0000h	37.5.1/1575
40AD_0020	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ8)	32	R/W	0000_0000h	37.5.1/1575
40AD_0024	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ9)	32	R/W	0000_0000h	37.5.1/1575
40AD_0028	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ10)	32	R/W	0000_0000h	37.5.1/1575
40AD_002C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ11)	32	R/W	0000_0000h	37.5.1/1575

Table continues on the next page...

IOMUXC1_DDR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AD_0030	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ12)	32	R/W	0000_0000h	37.5.1/1575
40AD_0034	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ13)	32	R/W	0000_0000h	37.5.1/1575
40AD_0038	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ14)	32	R/W	0000_0000h	37.5.1/1575
40AD_003C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ15)	32	R/W	0000_0000h	37.5.1/1575
40AD_0040	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ16)	32	R/W	0000_0000h	37.5.1/1575
40AD_0044	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ17)	32	R/W	0000_0000h	37.5.1/1575
40AD_0048	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ18)	32	R/W	0000_0000h	37.5.1/1575
40AD_004C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ19)	32	R/W	0000_0000h	37.5.1/1575
40AD_0050	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ20)	32	R/W	0000_0000h	37.5.1/1575
40AD_0054	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ21)	32	R/W	0000_0000h	37.5.1/1575
40AD_0058	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ22)	32	R/W	0000_0000h	37.5.1/1575
40AD_005C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ23)	32	R/W	0000_0000h	37.5.1/1575
40AD_0060	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ24)	32	R/W	0000_0000h	37.5.1/1575
40AD_0064	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ25)	32	R/W	0000_0000h	37.5.1/1575
40AD_0068	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ26)	32	R/W	0000_0000h	37.5.1/1575
40AD_006C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ27)	32	R/W	0000_0000h	37.5.1/1575
40AD_0070	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ28)	32	R/W	0000_0000h	37.5.1/1575
40AD_0074	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ29)	32	R/W	0000_0000h	37.5.1/1575
40AD_0078	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ30)	32	R/W	0000_0000h	37.5.1/1575
40AD_007C	SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ31)	32	R/W	0000_0000h	37.5.1/1575
40AD_0080	SW_PAD_CTL_PAD_DDR_DQSn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQS0)	32	R/W	0000_0184h	37.5.2/1576

Table continues on the next page...

IOMUXC1_DDR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AD_0084	SW_PAD_CTL_PAD_DDR_DQSn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQS1)	32	R/W	0000_0184h	37.5.2/1576
40AD_0088	SW_PAD_CTL_PAD_DDR_DQSn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQS2)	32	R/W	0000_0184h	37.5.2/1576
40AD_008C	SW_PAD_CTL_PAD_DDR_DQSn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQS3)	32	R/W	0000_0184h	37.5.2/1576
40AD_0090	SW_PAD_CTL_PAD_DDR_DQMn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQM0)	32	R/W	0000_0180h	37.5.3/1578
40AD_0094	SW_PAD_CTL_PAD_DDR_DQMn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQM1)	32	R/W	0000_0180h	37.5.3/1578
40AD_0098	SW_PAD_CTL_PAD_DDR_DQMn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQM2)	32	R/W	0000_0180h	37.5.3/1578
40AD_009C	SW_PAD_CTL_PAD_DDR_DQMn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQM3)	32	R/W	0000_0180h	37.5.3/1578
40AD_00A0	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA0)	32	R/W	0000_0000h	37.5.4/1580
40AD_00A4	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA1)	32	R/W	0000_0000h	37.5.4/1580
40AD_00A8	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA2)	32	R/W	0000_0000h	37.5.4/1580
40AD_00AC	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA3)	32	R/W	0000_0000h	37.5.4/1580
40AD_00B0	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA4)	32	R/W	0000_0000h	37.5.4/1580
40AD_00B4	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA5)	32	R/W	0000_0000h	37.5.4/1580
40AD_00B8	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA6)	32	R/W	0000_0000h	37.5.4/1580
40AD_00BC	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA7)	32	R/W	0000_0000h	37.5.4/1580
40AD_00C0	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA8)	32	R/W	0000_0000h	37.5.4/1580
40AD_00C4	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CA9)	32	R/W	0000_0000h	37.5.4/1580
40AD_00C8	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CS0_B)	32	R/W	0000_0000h	37.5.4/1580
40AD_00CC	SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CS1_B)	32	R/W	0000_0000h	37.5.4/1580

Table continues on the next page...

IOMUXC1_DDR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AD_00D0	SW_PAD_CTL_PAD_DDR_CKEn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CKE0)	32	R/W	0000_000Ch	37.5.5/1582
40AD_00D4	SW_PAD_CTL_PAD_DDR_CKEn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CKE1)	32	R/W	0000_000Ch	37.5.5/1582
40AD_00D8	SW_PAD_CTL_PAD_DDR_CLK0 SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CLK0)	32	R/W	0000_018Ch	37.5.6/1584
40AD_00DC	SW_PAD_CTL_PAD_DDR_ODT SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ODT)	32	R/W	0000_018Ch	37.5.7/1586
40AD_00E0	SW_PAD_CTL_PAD_DDR_ZQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ZQ0)	32	R/W	0004_0000h	37.5.8/1588
40AD_00E4	SW_PAD_CTL_PAD_DDR_ZQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ZQ1)	32	R/W	0004_0000h	37.5.8/1588
40AD_00E8	SW_PAD_CTL_PAD_HSIC_DATA SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_DATA)	32	R/W	0004_0184h	37.5.9/1589
40AD_00EC	SW_PAD_CTL_PAD_HSIC_STROBE SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_STROBE)	32	R/W	0004_0184h	37.5.10/ 1592
40AD_00F0	SW_PAD_CTL_GRP_PUE SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PUE)	32	R/W	0000_0004h	37.5.11/ 1594
40AD_00F4	SW_PAD_CTL_GRP_PUE_DAT SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PUE_DAT)	32	R/W	0000_0004h	37.5.12/ 1595
40AD_00F8	SW_PAD_CTL_GRP_PKE SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PKE)	32	R/W	0000_0008h	37.5.13/ 1595
40AD_00FC	SW_PAD_CTL_GRP_PKE_DAT SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PKE_DAT)	32	R/W	0000_0008h	37.5.14/ 1596
40AD_0100	SW_PAD_CTL_GRP_PUS SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PUS)	32	R/W	0000_0000h	37.5.15/ 1597
40AD_0104	SW_PAD_CTL_GRP_DS_ADDR SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_ADDR)	32	R/W	0000_0180h	37.5.16/ 1597
40AD_0108	SW_PAD_CTL_GRP_DS_CTRL SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_CTRL)	32	R/W	0000_0180h	37.5.17/ 1598
40AD_010C	SW_PAD_CTL_GRP_DS_DAT0 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT0)	32	R/W	0000_0180h	37.5.18/ 1599
40AD_0110	SW_PAD_CTL_GRP_DS_DAT1 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT1)	32	R/W	0000_0180h	37.5.19/ 1600
40AD_0114	SW_PAD_CTL_GRP_DS_DAT2 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT2)	32	R/W	0000_0180h	37.5.20/ 1600
40AD_0118	SW_PAD_CTL_GRP_DS_DAT3 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT3)	32	R/W	0000_0180h	37.5.21/ 1601
40AD_011C	SW_PAD_CTL_GRP_HYS SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_HYS)	32	R/W	0000_0000h	37.5.22/ 1602
40AD_0120	SW_PAD_CTL_GRP_INSEL_DAT SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_INSEL_DAT)	32	R/W	0000_0000h	37.5.23/ 1603

Table continues on the next page...

IOMUXC1_DDR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AD_0124	SW_PAD_CTL_GRP_INSEL_DQS SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_INSEL_DQS)	32	R/W	0000_0000h	37.5.24/ 1604
40AD_0128	SW_PAD_CTL_GRP_DDRTYPE SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DDRTYPE)	32	R/W	0004_0000h	37.5.25/ 1605

37.5.1 SW_PAD_CTL_PAD_DDR_DQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ*n*)

This control register applies to DDR data pads : DDR_DQ0 ~ DDR_DQ31. MMDC module controls the "On Die Termination" for these pads. Additional pad controls are implemented on group control registers.

Address: 40AD_0000h base + 0h offset + (4d × i), where i=0d to 31d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DDR_TRIM		DCYCLE_ TRIM		Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ*n* field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15–14 DDR_TRIM	Output Driver Delay Trim Field Select one out of next values for pad: DDR_DQn 00 **DDR_TRIM_0_0pS** — 0pS 01 **DDR_TRIM_1_50pS** — 50pS 10 **DDR_TRIM_2_100pS** — 100pS 11 **DDR_TRIM_3_150pS** — 150pS
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: DDR_DQn 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQ*n* field descriptions (continued)

Field	Description
-	This field is reserved. Reserved

37.5.2 SW_PAD_CTL_PAD_DDR_DQSn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQS*n*)

This control register applies to DDR_DQS0 ~ DDR_DQS3 pads. MMDC module controls the "On Die Termination" for these pads.

Address: 40AD_0000h base + 80h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		DCYCLE_TRIM		CRPOINT_TRIM		Reserved		DSE		PUS		PKE	PUE	Reserved	
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQS*n* field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: DDR_DQSn 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQS*n* field descriptions (continued)

Field	Description
11–10 CRPOINT_TRIM	Crosspoint Adjustment Field Select one out of next values for pad: DDR_DQSn 00 **CRPOINT_TRIM_0_no_output_crosspoint_Vix_change** — no output crosspoint (Vix) change 01 **CRPOINT_TRIM_1_100mV_Vix_shift_down** — 100mV Vix shift down 10 **CRPOINT_TRIM_2_100mV_Vix_shift_up** — 100mV Vix shift up 11 **CRPOINT_TRIM_3_200mV_Vix_shift_up** — 200mV Vix shift up
9 -	This field is reserved. Reserved
8–6 DSE	Output Drive Strength Select Field Select one out of next values for pad: DDR_DQSn 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
5–4 PUS	Pull Up/Down Resistance Select Field Select one out of next values for pad: DDR_DQSn 00 **PUS_0_100_kOhm_Pull_Down** — 100 kOhm Pull Down 01 **PUS_1_47_Kohm_Pull_Up** — 47 Kohm Pull Up 10 **PUS_2_100_kOhm_Pull_Up** — 100 kOhm Pull Up 11 **PUS_3_22_kOhm_Pull_Up** — 22 kOhm Pull Up
3 PKE	Pull Up/Pull Down/Keeper Enable Field Select one out of next values for pad: DDR_DQSn 0 **PKE_0_Pull_Up_Down_and_Keeper_Disabled** — Pull Up/Down and Keeper Disabled 1 **PKE_1_Pull_Up_Down_or_Keeper_Enabled** — Pull Up/Down or Keeper Enabled
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for pad: DDR_DQSn 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.3 SW_PAD_CTL_PAD_DDR_DQMn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQM*n*)

This control register applies to DDR_DQM0 ~ DDR_DQM3 pads. Additional pad controls are implemented on group control registers.

Address: 40AD_0000h base + 90h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										DDR_ODT			Reserved		DDR_INPUT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DDR_TRIM		DCYCLE_ TRIM		Reserved		HYS	DSE			Reserved					
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQM*n* field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–19 DDR_ODT	On Die Termination Select Field Select one out of next values for pad: DDR_DQM0 000 **DDR_ODT_0_No_Termination** — No Termination 001 **DDR_ODT_1_120_Ohm** — 120 Ohm 010 **DDR_ODT_2_60_Ohm** — 60 Ohm 011 **DDR_ODT_3_40_Ohm** — 40 Ohm 100 **DDR_ODT_4_30_Ohm** — 30 Ohm 101 **DDR_ODT_5_24_Ohm** — 24 Ohm 110 **DDR_ODT_6_20_Ohm** — 20 Ohm 111 **DDR_ODT_7_17_Ohm** — 17 Ohm
18–17 -	This field is reserved. Reserved
16 DDR_INPUT	DDR/CMOS Input Select Field Select one out of next values for pad: DDR_DQM0 0 **DDR_INPUT_0_CMOS_input_type** — CMOS input type 1 **DDR_INPUT_1_Differential_input_mode** — Differential input mode

Table continues on the next page...

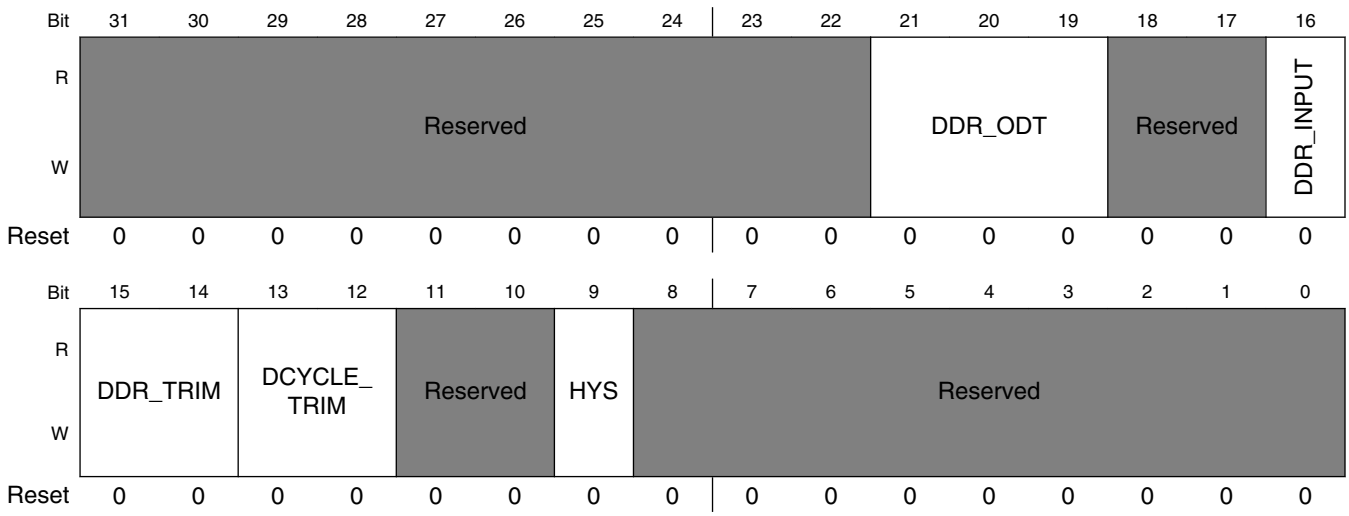
IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_DQM*n* field descriptions (continued)

Field	Description
15–14 DDR_TRIM	Output Driver Delay Trim Field Select one out of next values for pad: DDR_DQM0 00 **DDR_TRIM_0_0pS** — 0pS 01 **DDR_TRIM_1_50pS** — 50pS 10 **DDR_TRIM_2_100pS** — 100pS 11 **DDR_TRIM_3_150pS** — 150pS
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: DDR_DQM0 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change
11–10 -	This field is reserved. Reserved
9 HYS	Input Hysteresis Field Select one out of next values for pad: DDR_DQM0 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
8–6 DSE	Output Drive Strength Select Field Select one out of next values for pad: DDR_DQM0 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
-	This field is reserved. Reserved

37.5.4 SW_PAD_CTL_PAD_DDR_n SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR*n*)

This control register applies to DDR Address - DDR_CA0 ~ DDR_CA9 and DDR CS0_B, DDR_CS1_B pads. Additional pad controls are implemented on group control registers.

Address: 40AD_0000h base + A0h offset + (4d × i), where i=0d to 11d



IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR*n* field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–19 DDR_ODT	On Die Termination Select Field Select one out of next values for pad: DDR_CAn 000 **DDR_ODT_0_No_Termination** — No Termination 001 **DDR_ODT_1_120_Ohm** — 120 Ohm 010 **DDR_ODT_2_60_Ohm** — 60 Ohm 011 **DDR_ODT_3_40_Ohm** — 40 Ohm 100 **DDR_ODT_4_30_Ohm** — 30 Ohm 101 **DDR_ODT_5_24_Ohm** — 24 Ohm 110 **DDR_ODT_6_20_Ohm** — 20 Ohm 111 **DDR_ODT_7_17_Ohm** — 17 Ohm
18–17 -	This field is reserved. Reserved
16 DDR_INPUT	DDR/CMOS Input Select Field Select one out of next values for pad: DDR_CAn

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR*n* field descriptions (continued)

Field	Description
	0 **DDR_INPUT_0_CMOS_input_type** — CMOS input type 1 **DDR_INPUT_1_Differential_input_mode** — Differential input mode
15–14 DDR_TRIM	Output Driver Delay Trim Field Select one out of next values for pad: DDR_CAn 00 **DDR_TRIM_0_0pS** — 0pS 01 **DDR_TRIM_1_50pS** — 50pS 10 **DDR_TRIM_2_100pS** — 100pS 11 **DDR_TRIM_3_150pS** — 150pS
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: DDR_CAn 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change
11–10 -	This field is reserved. Reserved
9 HYS	Input Hysteresis Field Select one out of next values for pad: DDR_CAn 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
-	This field is reserved. Reserved

37.5.5 SW_PAD_CTL_PAD_DDR_CKEn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CKE*n*)

This control register applies to DDR_CKE0 and DDR_CKE1 pads. Additional pad controls are implemented on group control registers.

Address: 40AD_0000h base + D0h offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										DDR_ODT			Reserved		DDR_INPUT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DDR_TRIM		DCYCLE_TRIM		Reserved		HYS	Reserved			PUS		PKE	PUE	Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CKE*n* field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–19 DDR_ODT	On Die Termination Select Field Select one out of next values for pad: DDR_CKEn 000 **DDR_ODT_0_No_Termination** — No Termination 001 **DDR_ODT_1_120_Ohm** — 120 Ohm 010 **DDR_ODT_2_60_Ohm** — 60 Ohm 011 **DDR_ODT_3_40_Ohm** — 40 Ohm 100 **DDR_ODT_4_30_Ohm** — 30 Ohm 101 **DDR_ODT_5_24_Ohm** — 24 Ohm 110 **DDR_ODT_6_20_Ohm** — 20 Ohm 111 **DDR_ODT_7_17_Ohm** — 17 Ohm
18–17 -	This field is reserved. Reserved
16 DDR_INPUT	DDR/CMOS Input Select Field Select one out of next values for pad: DDR_CKEn 0 **DDR_INPUT_0_CMOS_input_type** — CMOS input type 1 **DDR_INPUT_1_Differential_input_mode** — Differential input mode

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CKE*n* field descriptions (continued)

Field	Description
15–14 DDR_TRIM	Output Driver Delay Trim Field Select one out of next values for pad: DDR_CKEn 00 **DDR_TRIM_0_0pS** — 0pS 01 **DDR_TRIM_1_50pS** — 50pS 10 **DDR_TRIM_2_100pS** — 100pS 11 **DDR_TRIM_3_150pS** — 150pS
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: DDR_CKEn 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change
11–10 -	This field is reserved. Reserved
9 HYS	Input Hysteresis Field Select one out of next values for pad: DDR_CKEn 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
8–6 -	This field is reserved. Reserved
5–4 PUS	Pull Up/Down Resistance Select Field Select one out of next values for pad: DDR_CKEn 00 **PUS_0_100_kOhm_Pull_Down** — 100 kOhm Pull Down 01 **PUS_1_47_Kohm_Pull_Up** — 47 Kohm Pull Up 10 **PUS_2_100_kOhm_Pull_Up** — 100 kOhm Pull Up 11 **PUS_3_22_kOhm_Pull_Up** — 22 kOhm Pull Up
3 PKE	Pull Up/Pull Down/Keeper Enable Field Select one out of next values for pad: DDR_CKEn 0 **PKE_0_Pull_Up_Down_and_Keeper_Disabled** — Pull Up/Down and Keeper Disabled 1 **PKE_1_Pull_Up_Down_or_Keeper_Enabled** — Pull Up/Down or Keeper Enabled
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for pad: DDR_CKEn 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.6 SW_PAD_CTL_PAD_DDR_CLK0 SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CLK0)

This control register applies to DDR_CLK0 pad. Additional pad controls are implemented on group control registers.

Address: 40AD_0000h base + D8h offset = 40AD_00D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										DDR_ODT			Reserved		DDR_INPUT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		DCYCLE_TRIM		CRPOINT_TRIM		HYS	DSE			PUS		PKE	PUE	Reserved	
Reset	0	0	0	0	0	0	0	1	1	0	0	0	1	1	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CLK0 field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–19 DDR_ODT	On Die Termination Select Field Select one out of next values for pad: DDR_CLK0 000 **DDR_ODT_0_No_Termination** — No Termination 001 **DDR_ODT_1_120_Ohm** — 120 Ohm 010 **DDR_ODT_2_60_Ohm** — 60 Ohm 011 **DDR_ODT_3_40_Ohm** — 40 Ohm 100 **DDR_ODT_4_30_Ohm** — 30 Ohm 101 **DDR_ODT_5_24_Ohm** — 24 Ohm 110 **DDR_ODT_6_20_Ohm** — 20 Ohm 111 **DDR_ODT_7_17_Ohm** — 17 Ohm
18–17 -	This field is reserved. Reserved
16 DDR_INPUT	DDR/CMOS Input Select Field Select one out of next values for pad: DDR_CLK0 0 **DDR_INPUT_0_CMOS_input_type** — CMOS input type 1 **DDR_INPUT_1_Differential_input_mode** — Differential input mode

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CLK0 field descriptions (continued)

Field	Description
15–14 -	This field is reserved. Reserved
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: DDR_CLK0 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change
11–10 CRPOINT_TRIM	Crosspoint Adjustment Field Select one out of next values for pad: DDR_CLK0 00 **CRPOINT_TRIM_0_no_output_crosspoint_Vix_change** — no output crosspoint (Vix) change 01 **CRPOINT_TRIM_1_100mV_Vix_shift_down** — 100mV Vix shift down 10 **CRPOINT_TRIM_2_100mV_Vix_shift_up** — 100mV Vix shift up 11 **CRPOINT_TRIM_3_200mV_Vix_shift_up** — 200mV Vix shift up
9 HYS	Input Hysteresis Field Select one out of next values for pad: DDR_CLK0 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
8–6 DSE	Output Drive Strength Select Field Select one out of next values for pad: DDR_CLK0 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
5–4 PUS	Pull Up/Down Resistance Select Field Select one out of next values for pad: DDR_CLK0 00 **PUS_0_100_kOhm_Pull_Down** — 100 kOhm Pull Down 01 **PUS_1_47_Kohm_Pull_Up** — 47 Kohm Pull Up 10 **PUS_2_100_kOhm_Pull_Up** — 100 kOhm Pull Up 11 **PUS_3_22_kOhm_Pull_Up** — 22 kOhm Pull Up
3 PKE	Pull Up/Pull Down/Keeper Enable Field Select one out of next values for pad: DDR_CLK0 0 **PKE_0_Pull_Up_Down_and_Keeper_Disabled** — Pull Up/Down and Keeper Disabled 1 **PKE_1_Pull_Up_Down_or_Keeper_Enabled** — Pull Up/Down or Keeper Enabled

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_CLK0 field descriptions (continued)

Field	Description
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for pad: DDR_CLK0 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.7 SW_PAD_CTL_PAD_DDR_ODT SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ODT)

This control register applies to DDR_ODT pad. Additional pad controls are implemented on group control registers.

Address: 40AD_0000h base + DCh offset = 40AD_00DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										DDR_ODT			Reserved		DDR_INPUT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DDR_TRIM		DCYCLE_ TRIM		Reserved		HYS	DSE			PUS		PKE	PUE	Reserved	
Reset	0	0	0	0	0	0	0	1	1	0	0	0	1	1	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ODT field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–19 DDR_ODT	On Die Termination Select Field Select one out of next values for pad: DDR_ODT 000 **DDR_ODT_0_No_Termination** — No Termination 001 **DDR_ODT_1_120_Ohm** — 120 Ohm 010 **DDR_ODT_2_60_Ohm** — 60 Ohm 011 **DDR_ODT_3_40_Ohm** — 40 Ohm

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ODT field descriptions (continued)

Field	Description
	100 **DDR_ODT_4_30_Ohm** — 30 Ohm 101 **DDR_ODT_5_24_Ohm** — 24 Ohm 110 **DDR_ODT_6_20_Ohm** — 20 Ohm 111 **DDR_ODT_7_17_Ohm** — 17 Ohm
18–17 -	This field is reserved. Reserved
16 DDR_INPUT	DDR/CMOS Input Select Field Select one out of next values for pad: DDR_ODT 0 **DDR_INPUT_0_CMOS_input_type** — CMOS input type 1 **DDR_INPUT_1_Differential_input_mode** — Differential input mode
15–14 DDR_TRIM	Output Driver Delay Trim Field Select one out of next values for pad: DDR_ODT 00 **DDR_TRIM_0_0pS** — 0pS 01 **DDR_TRIM_1_50pS** — 50pS 10 **DDR_TRIM_2_100pS** — 100pS 11 **DDR_TRIM_3_150pS** — 150pS
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: DDR_ODT 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change
11–10 -	This field is reserved. Reserved
9 HYS	Input Hysteresis Field Select one out of next values for pad: DDR_ODT 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
8–6 DSE	Output Drive Strength Select Field Select one out of next values for pad: DDR_ODT 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ODT field descriptions (continued)

Field	Description
5–4 PUS	Pull Up/Down Resistance Select Field Select one out of next values for pad: DDR_ODT 00 **PUS_0_100_kOhm_Pull_Down** — 100 kOhm Pull Down 01 **PUS_1_47_Kohm_Pull_Up** — 47 Kohm Pull Up 10 **PUS_2_100_kOhm_Pull_Up** — 100 kOhm Pull Up 11 **PUS_3_22_kOhm_Pull_Up** — 22 kOhm Pull Up
3 PKE	Pull Up/Pull Down/Keeper Enable Field Select one out of next values for pad: DDR_ODT 0 **PKE_0_Pull_Up_Down_and_Keeper_Disabled** — Pull Up/Down and Keeper Disabled 1 **PKE_1_Pull_Up_Down_or_Keeper_Enabled** — Pull Up/Down or Keeper Enabled
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for pad: DDR_ODT 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.8 SW_PAD_CTL_PAD_DDR_ZQn SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ZQ*n*)

This control register applies to DDR_ZQ0 and DDR_ZQ1 pads.

Address: 40AD_0000h base + E0h offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved													DDR_ SELECT	Reserv ed	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ZQ*n* field descriptions

Field	Description
31–19 -	This field is reserved. Reserved
18–17 DDR_SELECT	DDR Select Field Select one out of next values for pad: DDR_ZQn

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_DDR_ZQ*n* field descriptions (continued)

Field	Description
	00 **DDR_SELECT_0_DDR3_MODE** — DDR3 mode 01 **DDR_SELECT_1_reserved** — reserved 10 **DDR_SELECT_2_LPDDR2_LPDDR3** — LPDDR2/LPDDR3 modes 11 **DDR_SELECT_3_HSIC_USB mode** — HSIC-USB mode
-	This field is reserved. Reserved

37.5.9 SW_PAD_CTL_PAD_HSIC_DATA SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_DATA)

SW_PAD_CTL Register

Address: 40AD_0000h base + E8h offset = 40AD_00E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved										DDR_ODT			DDR_ SELECT		Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DDR_TRIM		DCYCLE_ TRIM		Reserved		HYS	DSE			PUS		Reserved	PUE	Reserved	
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0

IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_DATA field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–19 DDR_ODT	On Die Termination Select Field Select one out of next values for pad: HSIC_DATA

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_DATA field descriptions (continued)

Field	Description
	000 **DDR_ODT_0_No_Termination** — No Termination 001 **DDR_ODT_1_120_Ohm** — 120 Ohm 010 **DDR_ODT_2_60_Ohm** — 60 Ohm 011 **DDR_ODT_3_40_Ohm** — 40 Ohm 100 **DDR_ODT_4_30_Ohm** — 30 Ohm 101 **DDR_ODT_5_24_Ohm** — 24 Ohm 110 **DDR_ODT_6_20_Ohm** — 20 Ohm 111 **DDR_ODT_7_17_Ohm** — 17 Ohm
18–17 DDR_SELECT	DDR Select Field Select one out of next values for pad: HSIC_DATA 00 **DDR_SELECT_0_DDR3_MODE** — DDR3 mode 01 **DDR_SELECT_1_reserved** — reserved 10 **DDR_SELECT_2_LPDDR2_LPDDR3** — LPDDR2/LPDDR3 modes 11 **DDR_SELECT_3_HSIC_USB mode** — HSIC-USB mode
16 -	This field is reserved. Reserved
15–14 DDR_TRIM	Output Driver Delay Trim Field Select one out of next values for pad: HSIC_DATA 00 **DDR_TRIM_0_0pS** — 0pS 01 **DDR_TRIM_1_50pS** — 50pS 10 **DDR_TRIM_2_100pS** — 100pS 11 **DDR_TRIM_3_150pS** — 150pS
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: HSIC_DATA 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change
11–10 -	This field is reserved. Reserved
9 HYS	Input Hysteresis Field Select one out of next values for pad: HSIC_DATA 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
8–6 DSE	Output Drive Strength Select Field Select one out of next values for pad: HSIC_DATA 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm

Table continues on the next page...

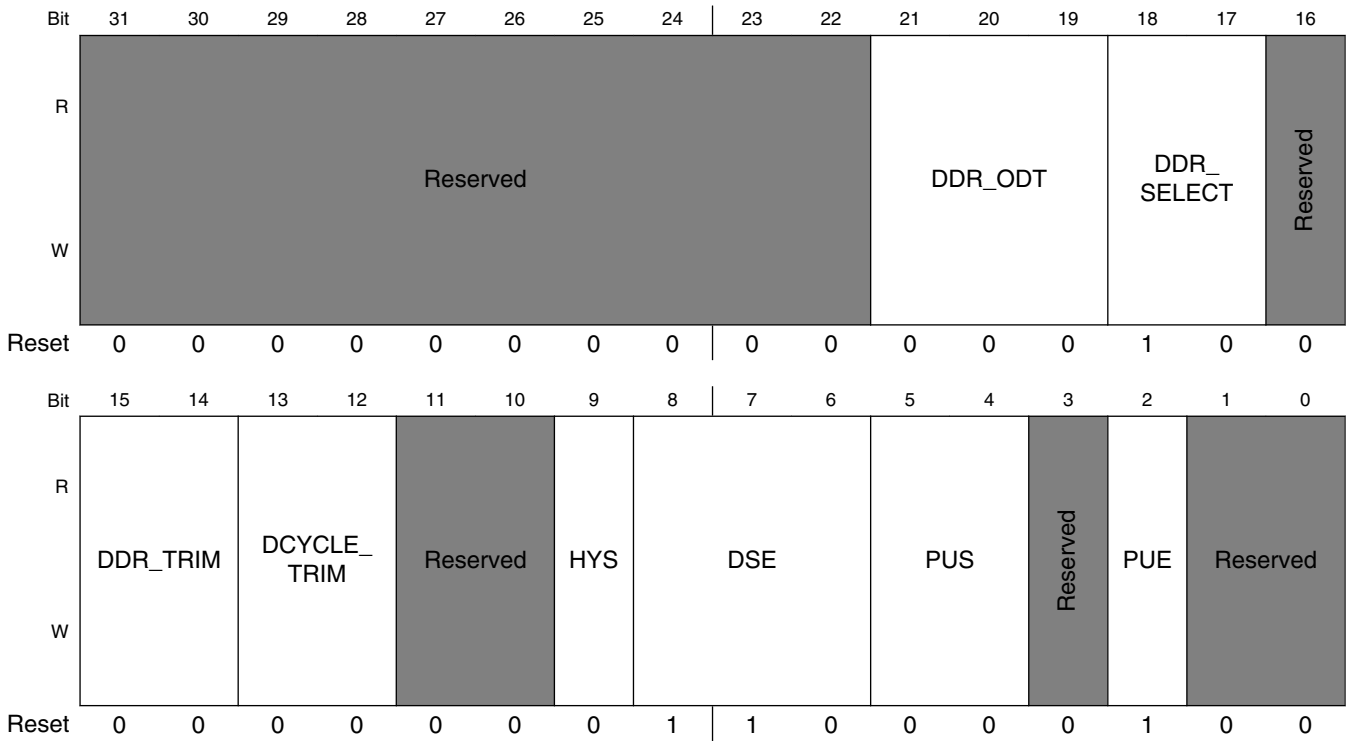
IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_DATA field descriptions (continued)

Field	Description
	100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
5–4 PUS	Pull Up/Down Resistance Select Field Select one out of next values for pad: HSIC_DATA 00 **PUS_0_100_kOhm_Pull_Down** — 100 kOhm Pull Down 01 **PUS_1_47_Kohm_Pull_Up** — 47 Kohm Pull Up 10 **PUS_2_100_kOhm_Pull_Up** — 100 kOhm Pull Up 11 **PUS_3_22_kOhm_Pull_Up** — 22 kOhm Pull Up
3 -	This field is reserved. Reserved
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for pad: HSIC_DATA 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.10 SW_PAD_CTL_PAD_HSIC_STROBE SW PAD Control Register (IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_STROBE)

SW_PAD_CTL Register

Address: 40AD_0000h base + ECh offset = 40AD_00ECh



IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_STROBE field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–19 DDR_ODT	On Die Termination Select Field Select one out of next values for pad: HSIC_STROBE 000 **DDR_ODT_0_No_Termination** — No Termination 001 **DDR_ODT_1_120_Ohm** — 120 Ohm 010 **DDR_ODT_2_60_Ohm** — 60 Ohm 011 **DDR_ODT_3_40_Ohm** — 40 Ohm 100 **DDR_ODT_4_30_Ohm** — 30 Ohm 101 **DDR_ODT_5_24_Ohm** — 24 Ohm 110 **DDR_ODT_6_20_Ohm** — 20 Ohm 111 **DDR_ODT_7_17_Ohm** — 17 Ohm

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_STROBE field descriptions (continued)

Field	Description
18–17 DDR_SELECT	DDR Select Field Select one out of next values for pad: HSIC_STROBE 00 **DDR_SELECT_0_DDR3_MODE** — DDR3 mode 01 **DDR_SELECT_1_reserved** — reserved 10 **DDR_SELECT_2_LPDDR2_LPDDR3** — LPDDR2/LPDDR3 modes 11 **DDR_SELECT_3_HSIC_USB mode** — HSIC-USB mode
16 -	This field is reserved. Reserved
15–14 DDR_TRIM	Output Driver Delay Trim Field Select one out of next values for pad: HSIC_STROBE 00 **DDR_TRIM_0_0pS** — 0pS 01 **DDR_TRIM_1_50pS** — 50pS 10 **DDR_TRIM_2_100pS** — 100pS 11 **DDR_TRIM_3_150pS** — 150pS
13–12 DCYCLE_TRIM	Duty Cycle Control Field Select one out of next values for pad: HSIC_STROBE 00 **DCYCLE_TRIM_0_no_duty_cycle_change** — no duty cycle change 01 **DCYCLE_TRIM_1_duty_cycle_increased** — duty cycle increased ~3.7% 10 **DCYCLE_TRIM_2_duty_cycle_decreased** — duty cycle decreased ~3.7% 11 **DCYCLE_TRIM_3_no_duty_cycle_change** — no duty cycle change
11–10 -	This field is reserved. Reserved
9 HYS	Input Hysteresis Field Select one out of next values for pad: HSIC_STROBE 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
8–6 DSE	Output Drive Strength Select Field Select one out of next values for pad: HSIC_STROBE 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
5–4 PUS	Pull Up/Down Resistance Select Field Select one out of next values for pad: HSIC_STROBE

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_PAD_HSIC_STROBE field descriptions (continued)

Field	Description
	00 **PUS_0_100_kOhm_Pull_Down** — 100 kOhm Pull Down 01 **PUS_1_47_Kohm_Pull_Up** — 47 Kohm Pull Up 10 **PUS_2_100_kOhm_Pull_Up** — 100 kOhm Pull Up 11 **PUS_3_22_kOhm_Pull_Up** — 22 kOhm Pull Up
3 -	This field is reserved. Reserved
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for pad: HSIC_STROBE 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.11 SW_PAD_CTL_GRP_PUE SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PUE)

SW_GRP Register

Address: 40AD_0000h base + F0h offset = 40AD_00F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													PUE	Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_PUE field descriptions

Field	Description
31–3 -	This field is reserved. Reserved
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for group: PUE (Pads: DDR_CA0 DDR_CA1 DDR_CA2 DDR_CA3 DDR_CA4 DDR_CA5 DDR_CA6 DDR_CA7 DDR_CA8 DDR_CA9 DDR_CS0_B DDR_CS1_B DDR_DQM0 DDR_DQM1 DDR_DQM2 DDR_DQM3) 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.12 SW_PAD_CTL_GRP_PUE_DAT SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PUE_DAT)

SW_GRP Register

Address: 40AD_0000h base + F4h offset = 40AD_00F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													PUE	Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_PUE_DAT field descriptions

Field	Description
31–3 -	This field is reserved. Reserved
2 PUE	Pull Up/Down or Keeper Selection Field Select one out of next values for group: PUE_DAT (Pads: DDR_DQ0 DDR_DQ1 DDR_DQ10 DDR_DQ11 DDR_DQ12 DDR_DQ13 DDR_DQ14 DDR_DQ15 DDR_DQ16 DDR_DQ17 DDR_DQ18 DDR_DQ19 DDR_DQ2 DDR_DQ20 DDR_DQ21 DDR_DQ22 DDR_DQ23 DDR_DQ24 DDR_DQ25 DDR_DQ26 DDR_DQ27 DDR_DQ28 DDR_DQ29 DDR_DQ3 DDR_DQ30 DDR_DQ31 DDR_DQ4 DDR_DQ5 DDR_DQ6 DDR_DQ7 DDR_DQ8 DDR_DQ9) 0 **PUE_0_Keeper_Selected** — Keeper Selected 1 **PUE_1_Pull_Up_Down_Selected** — Pull Up/Down Selected
-	This field is reserved. Reserved

37.5.13 SW_PAD_CTL_GRP_PKE SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PKE)

SW_GRP Register

Address: 40AD_0000h base + F8h offset = 40AD_00F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												PKE	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_PKE field descriptions

Field	Description
31–4 -	This field is reserved. Reserved
3 PKE	Pull Up/Pull Down/Keeper Enable Field Select one out of next values for group: PKE (Pads: DDR_CA0 DDR_CA1 DDR_CA2 DDR_CA3 DDR_CA4 DDR_CA5 DDR_CA6 DDR_CA7 DDR_CA8 DDR_CA9 DDR_CS0_B DDR_CS1_B DDR_DQM0 DDR_DQM1 DDR_DQM2 DDR_DQM3) 0 **PKE_0_Pull_Up_Down_and_Keeper_Disabled** — Pull Up/Down and Keeper Disabled 1 **PKE_1_Pull_Up_Down_or_Keeper_Enabled** — Pull Up/Down or Keeper Enabled
-	This field is reserved. Reserved

37.5.14 SW_PAD_CTL_GRP_PKE_DAT SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PKE_DAT)

SW_GRP Register

Address: 40AD_0000h base + FCh offset = 40AD_00FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												PKE	Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_PKE_DAT field descriptions

Field	Description
31–4 -	This field is reserved. Reserved
3 PKE	Pull Up/Pull Down/Keeper Enable Field Select one out of next values for group: PKE_DAT (Pads: DDR_DQ0 DDR_DQ1 DDR_DQ10 DDR_DQ11 DDR_DQ12 DDR_DQ13 DDR_DQ14 DDR_DQ15 DDR_DQ16 DDR_DQ17 DDR_DQ18 DDR_DQ19 DDR_DQ2 DDR_DQ20 DDR_DQ21 DDR_DQ22 DDR_DQ23 DDR_DQ24 DDR_DQ25 DDR_DQ26 DDR_DQ27 DDR_DQ28 DDR_DQ29 DDR_DQ3 DDR_DQ30 DDR_DQ31 DDR_DQ4 DDR_DQ5 DDR_DQ6 DDR_DQ7 DDR_DQ8 DDR_DQ9) 0 **PKE_0_Pull_Up_Down_and_Keeper_Disabled** — Pull Up/Down and Keeper Disabled 1 **PKE_1_Pull_Up_Down_or_Keeper_Enabled** — Pull Up/Down or Keeper Enabled
-	This field is reserved. Reserved

37.5.15 SW_PAD_CTL_GRP_PUS SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_PUS)

SW_GRP Register

Address: 40AD_0000h base + 100h offset = 40AD_0100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																										PUS		Reserved			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_PUS field descriptions

Field	Description
31–6 -	This field is reserved. Reserved
5–4 PUS	Pull Up/Down Resistance Select Field Select one out of next values for group: PUS (Pads: DDR_CA0 DDR_CA1 DDR_CA2 DDR_CA3 DDR_CA4 DDR_CA5 DDR_CA6 DDR_CA7 DDR_CA8 DDR_CA9 DDR_CS0_B DDR_CS1_B DDR_DQ0 DDR_DQ1 DDR_DQ10 DDR_DQ11 DDR_DQ12 DDR_DQ13 DDR_DQ14 DDR_DQ15 DDR_DQ16 DDR_DQ17 DDR_DQ18 DDR_DQ19 DDR_DQ2 DDR_DQ20 DDR_DQ21 DDR_DQ22 DDR_DQ23 DDR_DQ24 DDR_DQ25 DDR_DQ26 DDR_DQ27 DDR_DQ28 DDR_DQ29 DDR_DQ3 DDR_DQ30 DDR_DQ31 DDR_DQ4 DDR_DQ5 DDR_DQ6 DDR_DQ7 DDR_DQ8 DDR_DQ9 DDR_DQM0 DDR_DQM1 DDR_DQM2 DDR_DQM3) 00 **PUS_0_100_kOhm_Pull_Down** — 100 kOhm Pull Down 01 **PUS_1_47_Kohm_Pull_Up** — 47 Kohm Pull Up 10 **PUS_2_100_kOhm_Pull_Up** — 100 kOhm Pull Up 11 **PUS_3_22_kOhm_Pull_Up** — 22 kOhm Pull Up
-	This field is reserved. Reserved

37.5.16 SW_PAD_CTL_GRP_DS_ADDR SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_ADDR)

SW_GRP Register

Address: 40AD_0000h base + 104h offset = 40AD_0104h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																							DSE			Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

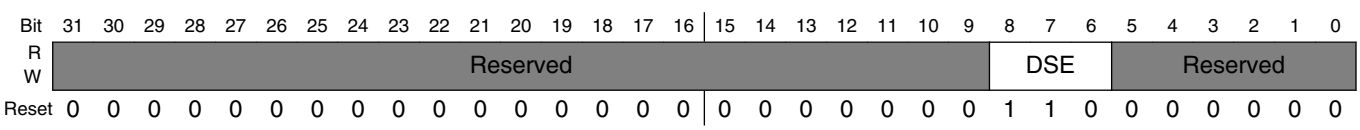
IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_ADDR field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8–6 DSE	Output Drive Strength Select Field Select one out of next values for group: DS_ADDR (Pads: DDR_CA0 DDR_CA1 DDR_CA2 DDR_CA3 DDR_CA4 DDR_CA5 DDR_CA6 DDR_CA7 DDR_CA8 DDR_CA9) 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
-	This field is reserved. Reserved

37.5.17 SW_PAD_CTL_GRP_DS_CTRL SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_CTRL)

SW_GRP Register

Address: 40AD_0000h base + 108h offset = 40AD_0108h



IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_CTRL field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8–6 DSE	Output Drive Strength Select Field Select one out of next values for group: DS_CTRL (Pads: DDR_CKE0 DDR_CKE1 DDR_CS0_B DDR_CS1_B) 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_CTRL field descriptions (continued)

Field	Description
	110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
-	This field is reserved. Reserved

37.5.18 SW_PAD_CTL_GRP_DS_DAT0 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT0)

SW_GRP Register

Address: 40AD_0000h base + 10Ch offset = 40AD_010Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																							DSE			Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT0 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8–6 DSE	Output Drive Strength Select Field Select one out of next values for group: DS_DAT0 (Pads: DDR_DQ0 DDR_DQ1 DDR_DQ2 DDR_DQ3 DDR_DQ4 DDR_DQ5 DDR_DQ6 DDR_DQ7) 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
-	This field is reserved. Reserved

37.5.19 SW_PAD_CTL_GRP_DS_DAT1 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT1)

SW_GRP Register

Address: 40AD_0000h base + 110h offset = 40AD_0110h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																							DSE			Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8–6 DSE	Output Drive Strength Select Field Select one out of next values for group: DS_DAT1 (Pads: DDR_DQ10 DDR_DQ11 DDR_DQ12 DDR_DQ13 DDR_DQ14 DDR_DQ15 DDR_DQ8 DDR_DQ9) 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
-	This field is reserved. Reserved

37.5.20 SW_PAD_CTL_GRP_DS_DAT2 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT2)

SW_GRP Register

Address: 40AD_0000h base + 114h offset = 40AD_0114h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																							DSE			Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8–6 DSE	Output Drive Strength Select Field Select one out of next values for group: DS_DAT2 (Pads: DDR_DQ16 DDR_DQ17 DDR_DQ18 DDR_DQ19 DDR_DQ20 DDR_DQ21 DDR_DQ22 DDR_DQ23) 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm 110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
-	This field is reserved. Reserved

37.5.21 SW_PAD_CTL_GRP_DS_DAT3 SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT3)

SW_GRP Register

Address: 40AD_0000h base + 118h offset = 40AD_0118h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																							DSE			Reserved					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT3 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8–6 DSE	Output Drive Strength Select Field Select one out of next values for group: DS_DAT3 (Pads: DDR_DQ24 DDR_DQ25 DDR_DQ26 DDR_DQ27 DDR_DQ28 DDR_DQ29 DDR_DQ30 DDR_DQ31) 000 **DSE_0_Driver_Disabled** — Driver Disabled 001 **DSE_1_240_Ohm** — 240 Ohm 010 **DSE_2_240_Ohm_2___120_Ohm** — 240/2=120 Ohm 011 **DSE_3_240_3__80_Ohm** — 240/3=80 Ohm 100 **DSE_4_240_4___60_Ohm** — 240/4=60 Ohm 101 **DSE_5_240_5___48_Ohm** — 240/5=48 Ohm

Table continues on the next page...

IOMUXC1_DDR_SW_PAD_CTL_GRP_DS_DAT3 field descriptions (continued)

Field	Description
	110 **DSE_6_240_6___40_Ohm** — 240/6=40 Ohm 111 **DSE_7_240_7___34_Ohm** — 240/7=34 Ohm
-	This field is reserved. Reserved

37.5.22 SW_PAD_CTL_GRP_HYS SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_HYS)

SW_GRP Register

Address: 40AD_0000h base + 11Ch offset = 40AD_011Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						HYS	Reserved								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

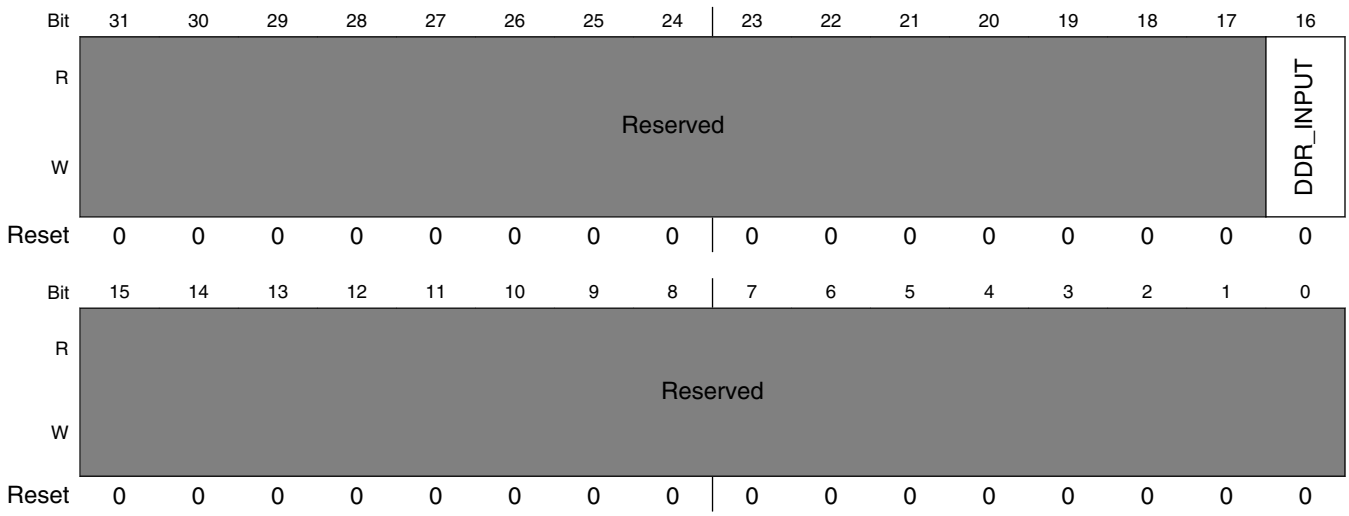
IOMUXC1_DDR_SW_PAD_CTL_GRP_HYS field descriptions

Field	Description
31–10 -	This field is reserved. Reserved
9 HYS	Input Hysteresis Field Select one out of next values for group: HYS (Pads: DDR_DQ0 DDR_DQ1 DDR_DQ10 DDR_DQ11 DDR_DQ12 DDR_DQ13 DDR_DQ14 DDR_DQ15 DDR_DQ16 DDR_DQ17 DDR_DQ18 DDR_DQ19 DDR_DQ2 DDR_DQ20 DDR_DQ21 DDR_DQ22 DDR_DQ23 DDR_DQ24 DDR_DQ25 DDR_DQ26 DDR_DQ27 DDR_DQ28 DDR_DQ29 DDR_DQ3 DDR_DQ30 DDR_DQ31 DDR_DQ4 DDR_DQ5 DDR_DQ6 DDR_DQ7 DDR_DQ8 DDR_DQ9 DDR_DQS0 DDR_DQS1 DDR_DQS2 DDR_DQS3) 0 **HYS_0_CMOS_input** — CMOS input 1 **HYS_1_Schmitt_trigger_input** — Schmitt trigger input
-	This field is reserved. Reserved

37.5.23 SW_PAD_CTL_GRP_INSEL_DAT SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_INSEL_DAT)

SW_GRP Register

Address: 40AD_0000h base + 120h offset = 40AD_0120h



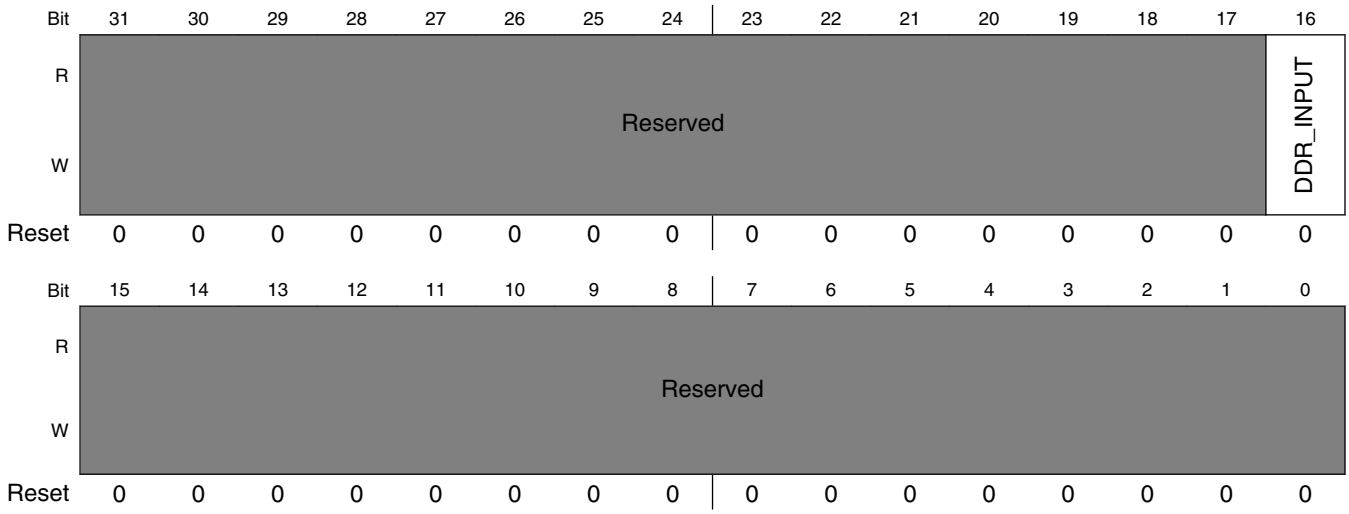
IOMUXC1_DDR_SW_PAD_CTL_GRP_INSEL_DAT field descriptions

Field	Description
31–17 -	This field is reserved. Reserved
16 DDR_INPUT	DDR/CMOS Input Select Field Select one out of next values for group: INSEL_DAT (Pads: DDR_DQ0 DDR_DQ1 DDR_DQ10 DDR_DQ11 DDR_DQ12 DDR_DQ13 DDR_DQ14 DDR_DQ15 DDR_DQ16 DDR_DQ17 DDR_DQ18 DDR_DQ19 DDR_DQ2 DDR_DQ20 DDR_DQ21 DDR_DQ22 DDR_DQ23 DDR_DQ24 DDR_DQ25 DDR_DQ26 DDR_DQ27 DDR_DQ28 DDR_DQ29 DDR_DQ3 DDR_DQ30 DDR_DQ31 DDR_DQ4 DDR_DQ5 DDR_DQ6 DDR_DQ7 DDR_DQ8 DDR_DQ9) 0 **DDR_INPUT_0_CMOS_input_type** — CMOS input type 1 **DDR_INPUT_1_Differential_input_mode** — Differential input mode
-	This field is reserved. Reserved

37.5.24 SW_PAD_CTL_GRP_INSEL_DQS SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_INSEL_DQS)

SW_GRP Register

Address: 40AD_0000h base + 124h offset = 40AD_0124h



IOMUXC1_DDR_SW_PAD_CTL_GRP_INSEL_DQS field descriptions

Field	Description
31–17 -	This field is reserved. Reserved
16 DDR_INPUT	DDR/CMOS Input Select Field Select one out of next values for group: INSEL_DQS (Pads: DDR_DQS0 DDR_DQS1 DDR_DQS2 DDR_DQS3) 0 **DDR_INPUT_0_CMOS_input_type** — CMOS input type 1 **DDR_INPUT_1_Differential_input_mode** — Differential input mode
-	This field is reserved. Reserved

37.5.25 SW_PAD_CTL_GRP_DDRTYPE SW GRP Register (IOMUXC1_DDR_SW_PAD_CTL_GRP_DDRTYPE)

SW_GRP Register

Address: 40AD_0000h base + 128h offset = 40AD_0128h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved													DDR_ SELECT	Reserv ed	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC1_DDR_SW_PAD_CTL_GRP_DDRTYPE field descriptions

Field	Description
31–19 -	This field is reserved. Reserved
18–17 DDR_SELECT	DDR Select Field Select one out of next values for group: DDRTYPE (Pads: DDR_CA0 DDR_CA1 DDR_CA2 DDR_CA3 DDR_CA4 DDR_CA5 DDR_CA6 DDR_CA7 DDR_CA8 DDR_CA9 DDR_CKE0 DDR_CKE1 DDR_CLK0 DDR_CS0_B DDR_CS1_B DDR_DQ0 DDR_DQ1 DDR_DQ10 DDR_DQ11 DDR_DQ12 DDR_DQ13 DDR_DQ14 DDR_DQ15 DDR_DQ16 DDR_DQ17 DDR_DQ18 DDR_DQ19 DDR_DQ2 DDR_DQ20 DDR_DQ21 DDR_DQ22 DDR_DQ23 DDR_DQ24 DDR_DQ25 DDR_DQ26 DDR_DQ27 DDR_DQ28 DDR_DQ29 DDR_DQ3 DDR_DQ30 DDR_DQ31 DDR_DQ4 DDR_DQ5 DDR_DQ6 DDR_DQ7 DDR_DQ8 DDR_DQ9 DDR_DQM0 DDR_DQM1 DDR_DQM2 DDR_DQM3 DDR_DQS0 DDR_DQS1 DDR_DQS2 DDR_DQS3 DDR_ODT) 00 **DDR_SELECT_0_DDR3_MODE** — DDR3 mode 01 **DDR_SELECT_1_reserved** — reserved 10 **DDR_SELECT_2_LPDDR2_LPDDR3** — LPDDR2/LPDDR3 modes 11 **DDR_SELECT_3_HSIC_USB mode** — HSIC-USB mode
-	This field is reserved. Reserved

37.6 Functional description

37.6.1 GPIO pads

The figure below shows the GPIO Pad .

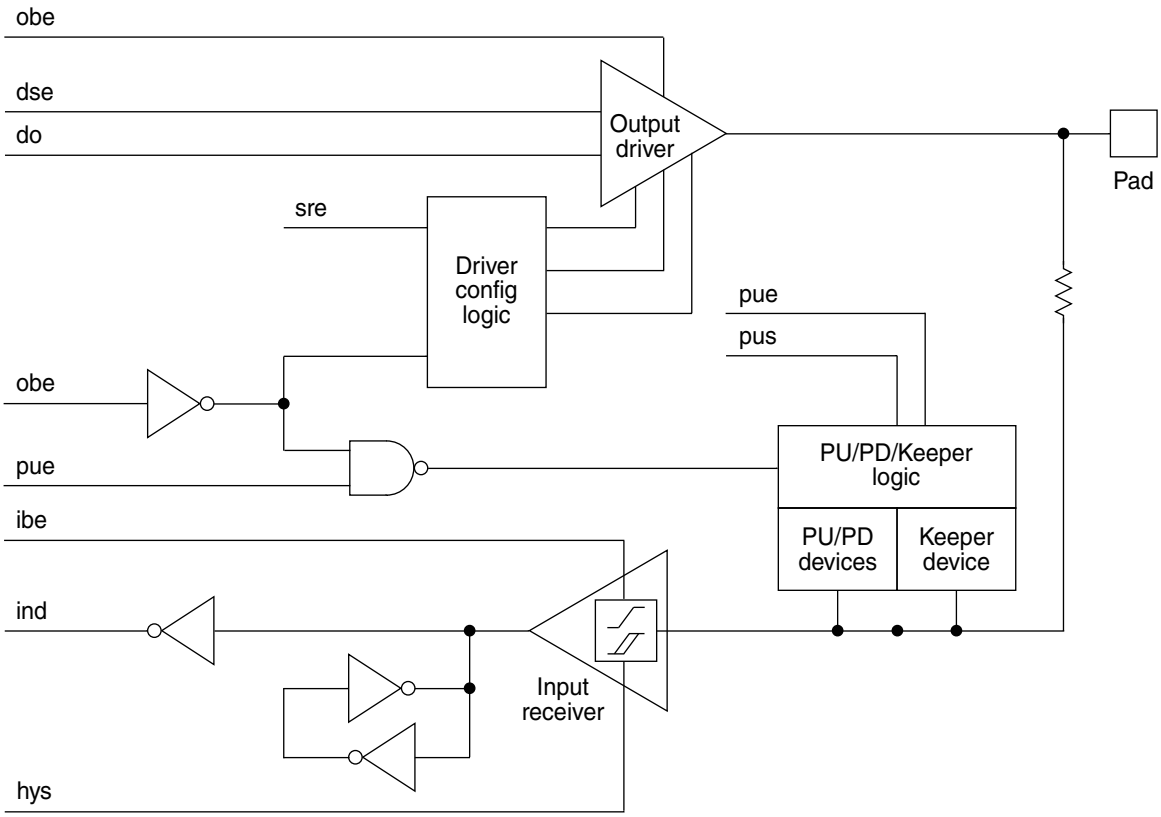


Figure 37-1. GPIO PAD

Table 37-3. Signal Description

Signal Name	Description	Description
Pad	Input/Output	I/O to external world
do	Input	Data coming from the core into the pad
obe	Input	Output enable
ode	Input	Select open drain or CMOS output
dse	Input	Drive Strength select
sre	Input	Slew rate control
pue	Input	Enable pull-up/down or keeper
pus	Input	Pull-up/down select
ibe	Input	Input enable
ind	Output	Data coming out of the pad into the core

NOTE

For DDR pads, there are additional control signals as explained in IOMUXC register map to control DDR functionality.

Table 37-4. Truth Table - Core to Pad

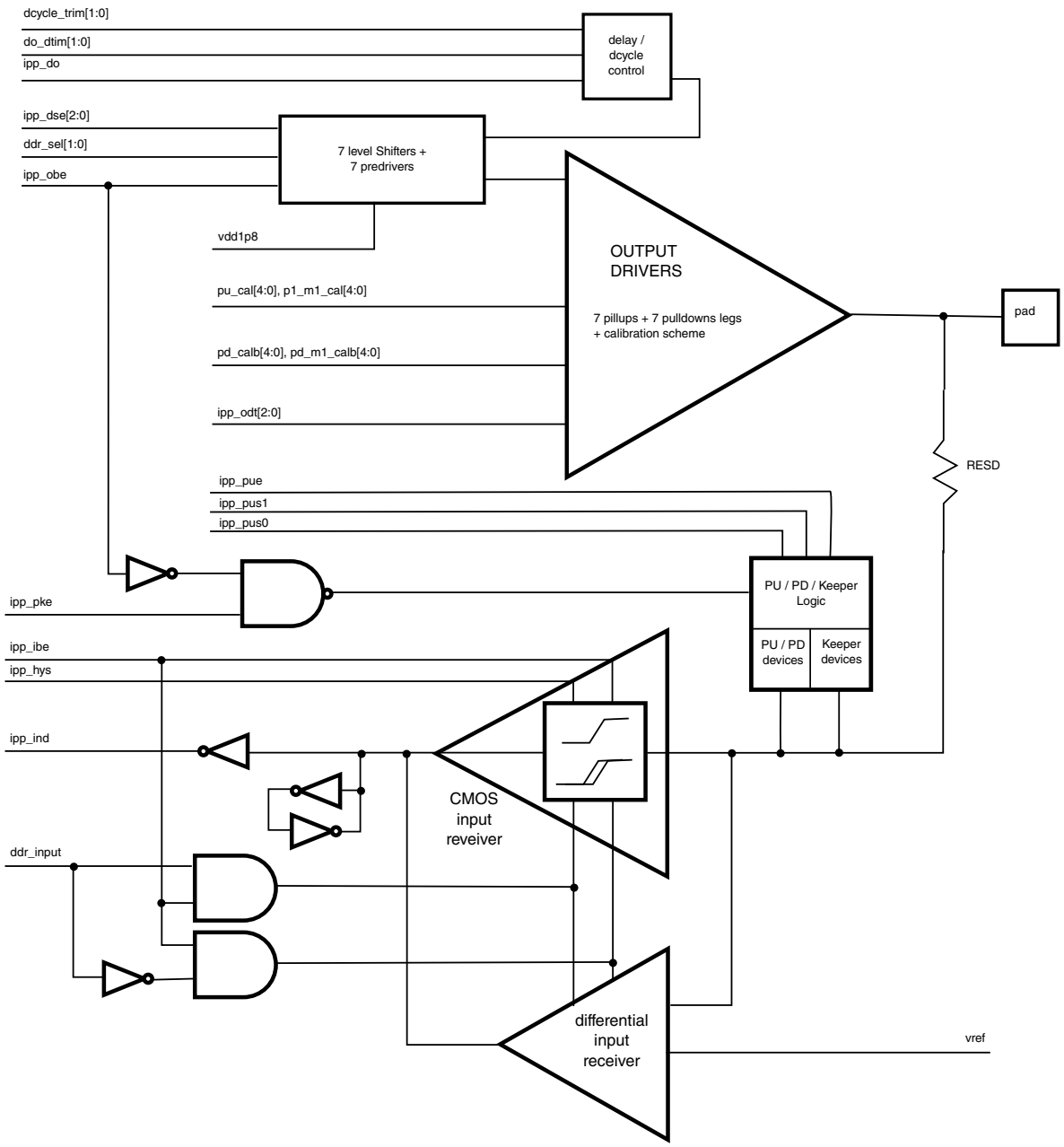
ode	obe	pue	pus	do	dse	pad
Output Buffer Mode (functional)						
0	1	X	X	do	X	do
0	0	X	X	X	X	Z
Output Buffer Mode + pull-up/pull-down resistors						
0	0	1	0	X	X	weak pull-down
0	0	1	1	X	X	weak pull-up
Open Drain Mode						
1	1	X	X	do	X	do
1	0	X	X	X	X	Z
Open Drain Mode + Pull-up						
1	1	1	1	0	X	0 (open drain + pull-up
1	0	X	X	X	X	Z

Table 37-5. Truth Table - Pad to Core

ibe	pad	ind
0	X	Safe value
1	pad	pad

IOMUX module is used to configure the Pad settings through the IOMUX registers and multiplex multiple modules on to a single PAD and drive input to a module from multiple PADs. Muxing is done only on obe, ibe, ind and do and rest of the pad settings are through IOMUX registers. DDR pads are dedicated pads for DDR and there is no muxing on them.

37.6.2 DDR pads


dcycle_trim[1:0]
do_dtim[1:0]
ipp_do
delay / dcycle control
ipp_dse[2:0]
ddr_sel[1:0]
ipp_obe
7 level Shifters + 7 predrivers
vdd1p8
OUTPUT DRIVERS
7 pillups + 7 pulldowns legs + calibration scheme
pad
pu_cal[4:0], p1_m1_cal[4:0]
pd_calb[4:0], pd_m1_calb[4:0]
ipp_odt[2:0]
RESD
ipp_pue
ipp_pus1
ipp_pus0
PU / PD / Keeper Logic
PU / PD devices
Keeper devices
ipp_pke
ipp_ibe
ipp_hys
ipp_ind
CMOS input reveiver
ddr_input
differential input receiver
vref

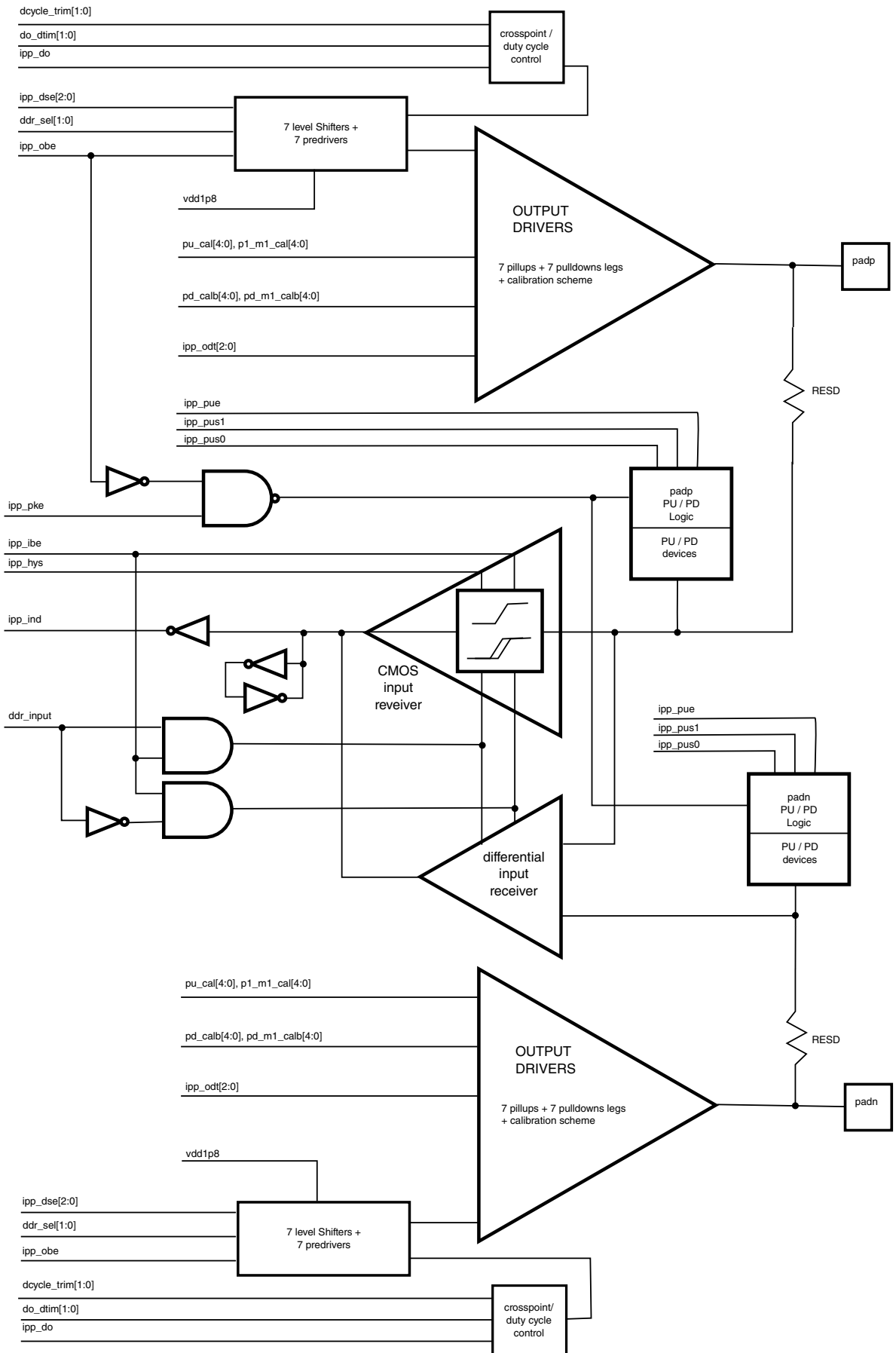

dcycle_trim[1:0]
do_dtim[1:0]
ipp_do
crosspoint / duty cycle control
ipp_dse[2:0]
ddr_sel[1:0]
ipp_obe
7 level Shifters + 7 predrivers
vdd1p8
OUTPUT DRIVERS
7 pillups + 7 pulldowns legs + calibration scheme
pu_cal[4:0], p1_m1_cal[4:0]
pd_calb[4:0], pd_m1_calb[4:0]
ipp_odt[2:0]
padp
RESD
ipp_pue
ipp_pus1
ipp_pus0
padp PU / PD Logic
PU / PD devices
ipp_pke
ipp_ibe
ipp_hys
ipp_ind
CMOS input reveiver
ddr_input
ipp_pue
ipp_pus1
ipp_pus0
padn PU / PD Logic
PU / PD devices
differential input receiver
pu_cal[4:0], p1_m1_cal[4:0]
pd_calb[4:0], pd_m1_calb[4:0]
ipp_odt[2:0]
OUTPUT DRIVERS
7 pillups + 7 pulldowns legs + calibration scheme
RESD
padn
vdd1p8
ipp_dse[2:0]
ddr_sel[1:0]
ipp_obe
7 level Shifters + 7 predrivers
dcycle_trim[1:0]
do_dtim[1:0]
ipp_do
crosspoint/ duty cycle control

Table 37-6. Signal Description

Name	Direction	Description
vdd1p8	input	1.8V PD predriver and level shifters supply voltage
pad	inout	I/O to the external world (single-ended and calibration cells)
padp, padn	inout	Positive and negative output pads to the external world (DDR CLK cell)
ipp_do	input	Data coming from the core into the pad
ipp_obe	input	Control signal for active high output enable (0=disable, 1=enable)
ddr_input	input	Control signal to select differential or CMOS input receiver mode: '0': CMOS input receiver mode (ipp_ibe=1). '1': differential DDR input receiver mode (ipp_ibe=1).
ipp_pke	input	Control signal to enable the pullup, pulldown resistors or pad keeper (in pbigjtv12_clamp_ddr, pbigjtv12_trigger_ddr cells only) capability: 0: capability disabled 1: capability enabled When ipp_pke=0, ipp_pue, ipp_pus0, ipp_pus1 have no functionality. Pad keeper is located only in pbigjtv12_ddr_single cell. There is no keeper in pbigjtv12_ddr_clk and calibration_ddr cells.
ipp_pue	input	Control signal to select pullup/ down resistors or pad keeper 0: pad keeper enable (pbigjtv12_ddr_single cell only; no functionality for pbigjtv12_ddr_clk cell) 1: pullup/down resistors enabled
ipp_pus0, ipp_pus1	input	Control signals to select pullup or pulldown resistors strength (in DDR CLK cell values are applicable for padp, for padn pu values mean pd and vice versa) **LPDDR2/LPDDR3/DDR3 modes:** 00: 100K Ohm PD res 01: 47K Ohm PU res 10: 100K Ohm PU res 11: 22K Ohm PU res **HSIC-USB mode:** 00: 30K Ohm PD res 01: 3K Ohm PU res 10: 100K Ohm PU res 11: 30K Ohm PU res
ipp_ind	output	Data coming out of the pad into the core

Table continues on the next page...

Table 37-6. Signal Description (continued)

Name	Direction	Description
ipp_ibe	input	Control signal for active high input enable (0=disable, 1=enable)
ipp_hys	input	Control signal to select Schmitt trigger or CMOS input (when CMOS input receiver selected) 0: CMOS input (for DDR CLK cell this is single-ended input mode and 'padp-]ipp_ind' path) 1: Schmitt trigger input
ddr_sel[1:0]	input	2-bit control signal to select DDR operational modes: 00 - DDR3 mode; 10 - LPDDR2/LPDDR3 modes; 01 - reserved 11 - HSIC-USB mode
ipp_dse[2:0]	input	3-bit control signal to select output drive strength (by enabling pullup/ pulldown driver units) 000 - output driver disabled; 001 - 240 Ohm 010 - 240/2 = 120 Ohm 011 - 240/3 = 80 Ohm 100 - 240/4 = 60 Ohm 101 - 240/5 = 48 Ohm 110 - 240/6 = 40 Ohm 111 - 240/7 = 34 Ohm (max drive strength)
ipp_odt[2:0]	input	3-bit control signal to select ODT strength in DDR3 mode: 001 - 120 Ohm ODT 010 - 60 Ohm ODT 011 - 40 Ohm ODT 100 - 30 Ohm ODT 101 - 24 Ohm ODT 110 - 20 Ohm ODT 111 - 17 Ohm ODT
do_trim[1:0] **(in ddr_single cell only)**	input	2-bit control signal for ipp_do-]pad delay in DDR single cell with ~50ps step (independent from PVT case): 00 : min delay; 01 : + ~50 ps delay; 10 : + ~100 ps delay; 11 : + ~150 ps delay;
pu_cal[4:0]	input	5-bit control signal for PU output driver sections ##4, 5, 6, 7 calibration
pu_m1_cal[4:0]	input	5-bit control signal for PU output driver sections ##1, 2, 3 calibration
pd_calb[4:0]	input	5-bit control signal for PD output driver sections ##4, 5, 6, 7 calibration
pd_m1_calb[4:0]	input	5-bit control signal for PD output driver sections ##1, 2, 3 calibration

Table continues on the next page...

Table 37-6. Signal Description (continued)

Name	Direction	Description
vref	input	DDR padring reference voltage (from cell: 'an_ddr_vref'), that is used for single-ended cell only. vref=ovdd/2

Table 37-7. DDR Cells Truth table - core to pad

ipp_obe	ipp_pke	ipp_pue	ipp_pus1	ipp_pus0	ipp_do	ipp_dse[2:0]	pad (single ended cell) / padp (CLK cell)	padn (CLK cell)
Output Buffer Mode (functional)								
1	0	X	X	X	ipp_do	001, 010, 011, 100, 101, 110, 111	ipp_do	! ipp_do
1	0	X	X	X	ipp_do	000	Z	Z
0	0	X	X	X	X	X	Z	Z
Output Buffer Mode + keeper (DDR single-ended cell only)								
0	1	0	X	X	X		pad (keep previous state)	-
Output Buffer Mode + pull-up/pull-down resistors								
0	1	1	0	0	X		weak0 (pull-down 100 KOhm)	weak1 (pull-up 100 KOhm)
0	1	1	0	1	X		weak1 (pull-up 47 KOhm)	weak0 (pull-down 47 KOhm)
0	1	1	1	0	X		weak1 (pull-up 100 KOhm)	weak0 (pull-down 100 KOhm)
0	1	1	1	1	X		weak1 (pull-up 22 KOhm)	weak0 (pull-down 22 KOhm)

Table 37-8. Truth table - pad to core (CMOS input)

ipp_ibe	pad (single ended cell) / padp (CLK cell)	ipp_ind
0	X	ipp_ind (keep previous state)
1	pad / padp	pad / padp

37.6.3 IOMUX Sheet

The following is a snippet from the IOMUX excel sheet attached with this document, where Port PTA0 has GPIO and the LPSPI0 Peripheral Chip Select 1 muxed onto it. Either of them can be selected by configuring MUX_MODE[2:0] bits of IOMUXC register for PTA0. Use the spreadsheet to set the SSS bits for input muxing and the SW_MUX bits to select the output muxing.

Port	CR	CR#	SW_MUX	Addr	Function	Module	Description	Direction
PTA0	PCR0_PTA0	0	0000_0000	0x4103_d000	CMP0_IN1_3V	CMP0	CMP0 Analog Voltage Input 1 (3v Range)	I
			0000_0001		PTA0	PTA	General Purpose I/O	I/O
			0000_0011		LPSPI0_PCS1	LPSPI0	LPSPI0 Peripheral Chip Select 1 or Host Request	I/O
			0000_0100		LPUART0_CTS_b	LPUART0	LPUART0 Clear to Send	I
			0000_0101		LPI2C0_SCL	LPI2C0	LPI2C0 Serial Clock Line	I/O
			0000_0110		TPM0_CLKIN	TPM0	TPM0 External Clock	I
			0000_0111		I2S0_RX_BCLK	I2S0	I2S0 Receive Bit Clock	I/O
			0000_1101		LLWU0_P0	LLWU0	LLWU Wakeup Input 0	I
			-		BT0_CFG0	CMC0	CM4 Boot Configuration 0	I

Figure 37-4. IOMUX Sheet Snippet

37.7 Special Pad Settings

This section describes the special pad setting requirements for several modules in this device. DDR Pins are dedicated pins and do not have any alternate functionality on them. The control for DDR pins are described in Miscellaneous Pins Configuration (Chapter 19). GPIO Pin settings (sre, ode, hys, dse, pus, pke,pue) are controlled by IOMUX registers, however, special care needs to be taken for ibe/obe. The ibe/obe bits on IOMUX registers are used to optionally enable pin's ibe/obe. Modules are responsible for controlling ibe/obe.

37.7.1 I2C

I2C pins, SCL and SDA are both Open-drain and bi-directional. Hence, ODE bit should be set in IOMUX registers.

37.7.2 SCI

Pull-ups should be enabled to ensure there are no spurious transitions.

37.7.3 Reset Pins Configuration

By default, a pull up is enabled on RESET0_B and RESET1_B pins. Additionally dse is set for those pins. Ibe and obe are controlled by reset controller.

Chapter 38
Trigger MUX (TRGMUX)

38.1 Chip-specific TRIGMUX information

Table 38-1. Reference links to related information

Topic	Related module	Reference
Full description	TRIGMUX	TRIGMUX
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

38.1.1 Trigger MUX (TRGMUX)

The Trigger MUX module (TRGMUX) allows software to configure the trigger inputs for various peripherals. Trigger inputs and outputs are defined according to the i.MX 7ULP Trigger MUX table.

Table 38-2. TRGMUX0 outputs

Register	Offset	Select	Output	Pretrigger	TRGMUX0 Outputs	Functions
TRGMUX_DMAMUX0	0x00	0	0	No	DMAMUX0 channel 0	Trigger
		1	1	No	DMAMUX0 channel 1	Trigger
		2	2	No	DMAMUX0 channel 2	Trigger
		3	3	No	DMAMUX0 channel 3	Trigger
	0x04	0	4	No	DMAMUX0 channel 4	Trigger
		1	5	No	DMAMUX0 channel 5	Trigger
		2	6	No	DMAMUX0 channel 6	Trigger
		3	7	No	DMAMUX0 channel 7	Trigger
TRGMUX_LPIT0	0x08	0	8	No	LPIT0 trigger 0	Trigger
		1	9	No	LPIT0 trigger 1	Trigger

Table continues on the next page...

Table 38-2. TRGMUX0 outputs (continued)

Register	Offset	Select	Output	Pretrigger	TRGMUX0 Outputs	Functions
		2	10	No	LPIT0 trigger 2	Trigger
		3	11	No	LPIT0 trigger 3	Trigger
		N/A	12 - 15	N/A	N/A	N/A
TRGMUX_TPM0	0x10	0	16	No	TPM0 channel 0	Input capture, trigger, output modulation
		1	17	No	TPM0 channel 1	Input capture, trigger
		2	18	No	TPM0 counter	Trigger
		N/A	19	N/A	N/A	N/A
TRGMUX_TPM1	0x14	0	20	No	TPM1 channel 0	Input capture, trigger, output modulation
		1	21	No	TPM1 channel 1	Input capture, trigger
		2	22	No	TPM1 counter	Trigger
		N/A	23	N/A	N/A	N/A
TRGMUX_TPM2	0x18	0	24	No	TPM2 channel 0	Input capture, trigger, output modulation
		1	25	No	TPM2 channel 1	Input capture, trigger
		2	26	No	TPM2 counter	Trigger
		N/A	27	N/A	N/A	N/A
TRGMUX_TPM3	0x1C	0	28	No	TPM3 channel 0	Input capture, trigger, output modulation
		1	29	No	TPM3 channel 1	Input capture, trigger
		2	30	No	TPM3 counter	Trigger
		N/A	31	N/A	N/A	N/A
TRGMUX_ADC0	0x20	0	32	No	ADC0	Trigger A
		1	33	No	ADC0	Trigger B
		N/A	34 - 35	N/A	N/A	N/A
TRGMUX_ADC1	0x24	0	36	No	ADC1	Trigger A
		1	37	No	ADC1	Trigger B
		N/A	38 - 39	N/A	N/A	N/A
TRGMUX_CMP0	0x28	0	40	No	CMP0	Window, trigger
		N/A	41 - 43	N/A	N/A	N/A
TRGMUX_CMP1	0x2C	0	44	No	CMP1	Window, trigger
		N/A	45 - 47	N/A	N/A	N/A
TRGMUX_DAC0	0x30	0	48	Yes	DAC0	Trigger
		N/A	49 - 51	N/A	N/A	N/A

Table continues on the next page...

Table 38-2. TRGMUX0 outputs (continued)

Register	Offset	Select	Output	Pretrigger	TRGMUX0 Outputs	Functions
TRGMUX_DAC1	0x34	0	52	Yes	DAC1	Trigger
		N/A	53 - 55	N/A	N/A	N/A
TRGMUX_LPUART0	0x38	0	56	No	LPUART0	RX input, CTS input, RX modulation
		N/A	57 - 59	N/A	N/A	N/A
TRGMUX_LPUART1	0x3C	0	60	No	LPUART1	RX input, CTS input, RX modulation
		N/A	61 - 63	N/A	N/A	N/A
TRGMUX_LPUART2	0x40	0	64	No	LPUART2	RX input, CTS input, RX modulation
		N/A	65 - 67	N/A	N/A	N/A
TRGMUX_LPUART3	0x44	0	68	No	LPUART3	RX input, CTS input, RX modulation
		N/A	69 - 71	N/A	N/A	N/A
TRGMUX_LPI2C0	0x48	0	72	No	LPI2C0	Host request
		N/A	73 - 75	N/A	N/A	N/A
TRGMUX_LPI2C1	0x4C	0	76	No	LPI2C1	Host request
		N/A	77 - 79	N/A	N/A	N/A
TRGMUX_LPI2C2	0x50	0	80	No	LPI2C2	Host request
		N/A	81 - 83	N/A	N/A	N/A
TRGMUX_LPI2C3	0x54	0	84	No	LPI2C3	Host request
		N/A	85 - 87	N/A	N/A	N/A
TRGMUX_LPSPI0	0x58	0	88	No	LPSPI0	Host request
		N/A	89 - 91	N/A	N/A	N/A
TRGMUX_LPSPI1	0x5C	0	92	No	LPSPI1	Host request
		N/A	93 - 95	N/A	N/A	N/A
TRGMUX_FLEXIO0	0x60	0	96	No	FlexIO0 trigger 0	Trigger
		1	97	No	FlexIO0 trigger 1	Trigger
		2	98	No	FlexIO0 trigger 2	Trigger
		3	99	No	FlexIO0 trigger 3	Trigger

Table 38-3. TRGMUX0 inputs

Mux Select	Source Module	Source Function	Pretrigger	Source Location
0 - 1	Reserved	Reserved	N/A	N/A
2	FlexIO0	Timer 0	No	SoC
3	FlexIO0	Timer 1	No	SoC
4	FlexIO0	Timer 2	No	SoC

Table continues on the next page...

Table 38-3. TRGMUX0 inputs (continued)

Mux Select	Source Module	Source Function	Pretrigger	Source Location
5	FlexIO0	Timer 3	No	SoC
6	FlexIO0	Timer 4	No	SoC
7	FlexIO0	Timer 5	No	SoC
8	FlexIO0	Timer 6	No	SoC
9	FlexIO0	Timer 7	No	SoC
10	TPM0	Overflow	No	SoC
11	TPM0	Channel 0	Yes	SoC
12	TPM0	Channel 1	Yes	SoC
13	TPM1	Overflow	No	SoC
14	TPM1	Channel 0	Yes	SoC
15	TPM1	Channel 1	Yes	SoC
16	TPM2	Overflow	No	SoC
17	TPM2	Channel 0	Yes	SoC
18	TPM2	Channel 1	Yes	SoC
19	TPM3	Overflow	No	SoC
20	TPM3	Channel 0	Yes	SoC
21	TPM3	Channel 1	Yes	SoC
22	LPIT0	Channel 0	Yes	SoC
23	LPIT0	Channel 1	Yes	SoC
24	LPIT0	Channel 2	Yes	SoC
25	LPIT0	Channel 3	Yes	SoC
26 - 29	Reserved	Reserved	N/A	N/A
30	LPUART0	RX data	No	SoC
31	LPUART0	TX data	No	SoC
32	LPUART0	RX idle	No	SoC
33	LPUART1	RX data	No	SoC
34	LPUART1	TX data	No	SoC
35	LPUART1	RX Idle	No	SoC
36	LPUART2	RX data	No	SoC
37	LPUART2	TX data	No	SoC
38	LPUART2	RX idle	No	SoC
39	LPUART3	RX data	No	SoC
40	LPUART3	TX data	No	SoC
41	LPUART3	RX idle	No	SoC
42	LPI2C0	Master STOP	No	SoC
43	LPI2C0	Slave STOP	No	SoC
44	LPI2C1	Master STOP	No	SoC
45	LPI2C1	Slave STOP	No	SoC
46	LPI2C2	Master STOP	No	SoC

Table continues on the next page...

Table 38-3. TRGMUX0 inputs (continued)

Mux Select	Source Module	Source Function	Pretrigger	Source Location
47	LPI2C2	Slave STOP	No	SoC
48	LPI2C3	Master STOP	No	SoC
49	LPI2C3	Slave STOP	No	SoC
50	LPSPI0	Frame	No	SoC
51	LPSPI0	RX data	No	SoC
52	LPSPI1	Frame	No	SoC
53	LPSPI1	RX data	No	SoC
54 - 55	Reserved	Reserved	N/A	N/A
56	LPTMR0	Trigger	Yes	SoC
57	LPTMR1	Trigger	Yes	SoC
58	CMP0	Output	No	SoC
59	CMP1	Output	No	SoC
60 - 63	Reserved	Reserved	N/A	N/A
64	PORT A	Pin Trigger	No	SoC
65	PORT B	Pin Trigger	No	SoC
66	I2S0	TX Frame Sync	No	SoC
67	I2S0	RX Frame Sync	No	SoC
68	I2S1	TX Frame Sync	No	SoC
69	I2S1	RX Frame Sync	No	SoC

Table 38-4. TRGMUX1 outputs

Register	Offset	Select	Output	Pretrigger	TRGMUX0 Outputs	Functions
TRGMUX_DMAMUX1	0x00	0	0	No	DMAMUX1 channel 0	Trigger
		1	1	No	DMAMUX1 channel 1	Trigger
		2	2	No	DMAMUX1 channel 2	Trigger
		3	3	No	DMAMUX1 channel 3	Trigger
	0x04	0	4	No	DMAMUX1 channel 4	Trigger
		1	5	No	DMAMUX1 channel 5	Trigger
		2	6	No	DMAMUX1 channel 6	Trigger
		3	7	No	DMAMUX1 channel 7	Trigger
TRGMUX_TPM4	0x08	0	8	No	TPM4 channel 0	Input capture, trigger, output modulation
		1	9	No	TPM4 channel 1	Input capture, trigger
		2	10	No	TPM4 counter	Trigger
		N/A	11	N/A	N/A	N/A
TRGMUX_TPM5	0x0C	0	12	No	TPM5 channel 0	Input capture, trigger, output modulation

Table continues on the next page...

Table 38-4. TRGMUX1 outputs (continued)

Register	Offset	Select	Output	Pretrigger	TRGMUX0 Outputs	Functions
		1	13	No	TPM5 channel 1	Input capture, trigger
		2	14	No	TPM5 counter	Trigger
		N/A	15	N/A	N/A	N/A
TRGMUX_TPM6	0x10	0	16	No	TPM6 channel 0	Input capture, trigger, output modulation
		1	17	No	TPM6 channel 1	Input capture, trigger
		2	18	No	TPM6 counter	Trigger
		N/A	19	N/A	N/A	N/A
TRGMUX_TPM7	0x14	0	20	No	TPM7 channel 0	Input capture, trigger, output modulation
		1	21	No	TPM7 channel 1	Input capture, trigger
		2	22	No	TPM7 counter	Trigger
		N/A	23	N/A	N/A	N/A
TRGMUX_LPUART4	0x18	0	24	No	LPUART4	RX input, CTS input, RX modulation
		N/A	25 - 27	N/A	N/A	N/A
TRGMUX_LPUART5	0x1C	0	28	No	LPUART5	RX input, CTS input, RX modulation
		N/A	29 - 31	N/A	N/A	N/A
TRGMUX_LPUART6	0x20	0	32	No	LPUART6	RX input, CTS input, RX modulation
		N/A	33 - 35	N/A	N/A	N/A
TRGMUX_LPUART7	0x24	0	36	No	LPUART7	RX input, CTS input, RX modulation
		N/A	37 - 39	N/A	N/A	N/A
TRGMUX_LPI2C4	0x28	0	40	No	LPI2C4	Host request
		N/A	41 - 43	N/A	N/A	N/A
TRGMUX_LPI2C5	0x2C	0	44	No	LPI2C5	Host request
		N/A	45 - 47	N/A	N/A	N/A
TRGMUX_LPI2C6	0x30	0	48	No	LPI2C6	Host request
		N/A	49 - 51	N/A	N/A	N/A
TRGMUX_LPI2C7	0x34	0	52	No	LPI2C7	Host request

Table continues on the next page...

Table 38-4. TRGMUX1 outputs (continued)

Register	Offset	Select	Output	Pretrigger	TRGMUX0 Outputs	Functions
		N/A	53 - 55	N/A	N/A	N/A
TRGMUX_LPSPI2	0x38	0	56	No	LPSPI2	Host request
		N/A	57 - 59	N/A	N/A	N/A
TRGMUX_LPSPI3	0x3C	0	60	No	LPSPI3	Host request
		N/A	61 - 63	N/A	N/A	N/A
TRGMUX_FLEXIO1	0x40	0	64	No	FlexIO1 trigger 0	Trigger
		1	65	No	FlexIO1 trigger 1	Trigger
		2	66	No	FlexIO1 trigger 2	Trigger
		3	67	No	FlexIO1 trigger 3	Trigger
TRGMUX_LPIT1	0x44	0	68	No	LPIT1 trigger 0	Trigger
		1	69	No	LPIT1 trigger 1	Trigger
		2	70	No	LPIT1 trigger 2	Trigger
		3	71	No	LPIT1 trigger 3	Trigger

Table 38-5. TRGMUX1 inputs

Mux Select	Source Module	Source Function	Pretrigger	Source Location
0	Reserved	Reserved	N/A	N/A
1	FlexIO1	Timer 0	No	SoC
2	FlexIO1	Timer 1	No	SoC
3	FlexIO1	Timer 2	No	SoC
4	FlexIO1	Timer 3	No	SoC
5	FlexIO1	Timer 4	No	SoC
6	FlexIO1	Timer 5	No	SoC
7	FlexIO1	Timer 6	No	SoC
8	FlexIO1	Timer 7	No	SoC
9	TPM4	Overflow	No	SoC
10	TPM4	Channel 0	Yes	SoC
11	TPM4	Channel 1	Yes	SoC
12	TPM5	Overflow	No	SoC
13	TPM5	Channel 0	Yes	SoC
14	TPM5	Channel 1	Yes	SoC
15	TPM6	Overflow	No	SoC
16	TPM6	Channel 0	Yes	SoC
17	TPM6	Channel 1	Yes	SoC
18	TPM7	Overflow	No	SoC
19	TPM7	Channel 0	Yes	SoC
20	TPM7	Channel 1	Yes	SoC
21	LPUART4	RX data	No	SoC

Table continues on the next page...

Table 38-5. TRGMUX1 inputs (continued)

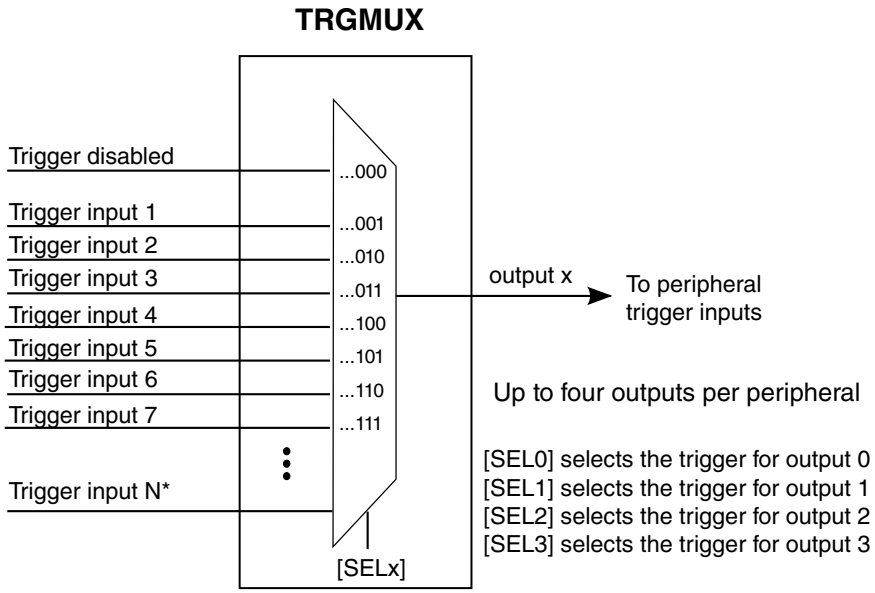
Mux Select	Source Module	Source Function	Pretrigger	Source Location
22	LPUART4	TX data	No	SoC
23	LPUART4	RX idle	No	SoC
24	LPUART5	RX data	No	SoC
25	LPUART5	TX data	No	SoC
26	LPUART5	RX Idle	No	SoC
27	LPUART6	RX data	No	SoC
28	LPUART6	TX data	No	SoC
29	LPUART6	RX idle	No	SoC
30	LPUART7	RX data	No	SoC
31	LPUART7	TX data	No	SoC
32	LPUART7	RX idle	No	SoC
33	LPI2C4	Master STOP	No	SoC
34	LPI2C4	Slave STOP	No	SoC
35	LPI2C5	Master STOP	No	SoC
36	LPI2C5	Slave STOP	No	SoC
37	LPI2C6	Master STOP	No	SoC
38	LPI2C6	Slave STOP	No	SoC
39	LPI2C7	Master STOP	No	SoC
40	LPI2C7	Slave STOP	No	SoC
41	LPSPI2	Frame	No	SoC
42	LPSPI2	RX data	No	SoC
43	LPSPI3	Frame	No	SoC
44	LPSPI3	RX data	No	SoC
45	PORT C	Pin Trigger	No	SoC
46	PORT D	Pin Trigger	No	SoC
47	PORT E	Pin Trigger	No	SoC
48	PORT F	Pin Trigger	No	SoC
49	USB0	Start of Frame	No	SoC
50	USB1	Start of Frame	No	SoC
51	LPIT1	Channel 0	Yes	SoC
52	LPIT1	Channel 1	Yes	SoC
53	LPIT1	Channel 2	Yes	SoC
54	LPIT1	Channel 3	Yes	SoC

38.2 Introduction

The trigger multiplexer (TRGMUX) module allows software to configure the trigger inputs for various peripherals.

38.3 Features

The TRGMUX module allows software to select the trigger source for peripherals. The block diagram below shows the trigger selection logic of the TRGMUX module.



* Up to 255 trigger inputs may be available for SEL0, SEL1, and SEL2. For SEL3, up to 127 trigger inputs may be available. When the number of trigger inputs is 255, SEL3 is not available and becomes reserved. See the chip-specific TRGMUX information for the maximum number of trigger inputs supported on this device.

Figure 38-1. TRGMUX block diagram

Each peripheral has its own dedicated TRGMUX register. See each peripheral's TRGMUX register for details.

38.4 Memory map and register definition

The TRGMUX registers contain fields for selecting trigger sources for peripheral modules.

TRGMUX registers can be written only in supervisor mode.

38.4.1 TRGMUX register descriptions

38.4.1.1 TRGMUX Memory map

Table 38-6. Select Bit Fields

Field	Description
SELx	This read/write field is used to configure the MUX select for the peripheral trigger inputs. 0000_0000 - (0x00) - 0000_0001 - (0x01) Reserved 0000_0010 - (0x02) FlexIO0 0000_0011 - (0x03) FlexIO0 0000_0100 - (0x04) FlexIO0 0000_0101 - (0x05) FlexIO0 0000_0110 - (0x06) FlexIO0 0000_0111 - (0x07) FlexIO0 0000_1000 - (0x08) FlexIO0 0000_1001 - (0x09) FlexIO0 0000_1010 - (0x0A) TPM0 0000_1011 - (0x0B) TPM0 0000_1100 - (0x0C) TPM0 0000_1101 - (0x0D) TPM1 0000_1110 - (0x0E) TPM1 0000_1111 - (0x0F) TPM1 0001_0000 - (0x10) TPM2 0001_0001 - (0x11) TPM2 0001_0010 - (0x12) TPM2 0001_0011 - (0x13) TPM3 0001_0100 - (0x14) TPM3 0001_0101 - (0x15) TPM3 0001_0110 - (0x16) LPIT0 0001_0111 - (0x17) LPIT0 0001_1000 - (0x18) LPIT0 0001_1001 - (0x19) LPIT0 0001_1010 - (0x1A) Reserved 0001_1011 - (0x1B) Reserved 0001_1100 - (0x1C) Reserved 0001_1101 - (0x1D) Reserved 0001_1110 - (0x1E) LPUART0 0001_1111 - (0x1F) LPUART0 0010_0000 - (0x20) LPUART0 0010_0001 - (0x21) LPUART1

Table 38-6. Select Bit Fields

Field	Description
	0010_0010 - (0x22) LPUART1
	0010_0011 - (0x23) LPUART1
	0010_0100 - (0x24) LPUART2
	0010_0101 - (0x25) LPUART2
	0010_0110 - (0x26) LPUART2
	0010_0111 - (0x27) LPUART3
	0010_1000 - (0x28) LPUART3
	0010_1001 - (0x29) LPUART3
	0010_1010 - (0x2A) LPI2C0
	0010_1011 - (0x2B) LPI2C0
	0010_1100 - (0x2C) LPI2C1
	0010_1101 - (0x2D) LPI2C1
	0010_1110 - (0x2E) LPI2C2
	0010_1111 - (0x2F) LPI2C2
	0011_0000 - (0x30) LPI2C3
	0011_0001 - (0x31) LPI2C3
	0011_0010 - (0x32) LPSPI0
	0011_0011 - (0x33) LPSPI0
	0011_0100 - (0x34) LPSPI1
	0011_0101 - (0x35) LPSPI1
	0011_0110 - (0x36) Reserved
	0011_0111 - (0x37) Reserved
	0011_1000 - (0x38) LPTMR0
	0011_1001 - (0x39) LPTMR1
	0011_1010 - (0x3A) CMP0
	0011_1011 - (0x3B) CMP1
	0011_1100 - (0x3C) Reserved
	0011_1101 - (0x3D) Reserved
	0011_1110 - (0x3E) Reserved
	0011_1111 - (0x3F) Reserved
	0100_0000 - (0x40) Port A
	0100_0001 - (0x41) Port B
	0100_0010 - (0x42) I2S0
	0100_0011 - (0x43) I2S0
	0100_0100 - (0x44) I2S1
	0100_0101 - (0x45) I2S1
	0100_0110 - (0x46) Reserved
	0100_0111 - (0x47) Reserved

Table 38-6. Select Bit Fields

Field	Description
	0100_1000 - (0x48) Reserved
	0100_1001 - (0x49) Reserved
	0100_1010 - (0x4A) Reserved
	0100_1011 - (0x4B) Reserved
	0100_1100 - (0x4C) Reserved
	0100_1101 - (0x4D) Reserved
	0100_1110 - (0x4E) Reserved
	0100_1111 - (0x4F) Reserved
	0101_0000 - (0x50) Reserved
	0101_0001 - (0x51) Reserved
	0101_0010 - (0x52) Reserved
	0101_0011 - (0x53) Reserved
	0101_0100 - (0x54) Reserved
	0101_0101 - (0x55) Reserved
	0101_0110 - (0x56) Reserved
	0101_0111 - (0x57) Reserved
	0101_1000 - (0x58) Reserved
	0101_1001 - (0x59) Reserved
	0101_1010 - (0x5A) Reserved
	0101_1011 - (0x5B) Reserved
	0101_1100 - (0x5C) Reserved
	0101_1101 - (0x5D) Reserved
	0101_1110 - (0x5E) Reserved
	0101_1111 - (0x5F) Reserved
	0110_0000 - (0x60) Reserved
	0110_0001 - (0x61) Reserved
	0110_0010 - (0x62) Reserved
	0110_0011 - (0x63) Reserved
	0110_0100 - (0x64) Reserved
	0110_0101 - (0x65) Reserved
	0110_0110 - (0x66) Reserved
	0110_0111 - (0x67) Reserved
	0110_1000 - (0x68) Reserved
	0110_1001 - (0x69) Reserved
	0110_1010 - (0x6A) Reserved
	0110_1011 - (0x6B) Reserved
	0110_1100 - (0x6C) Reserved
	0110_1101 - (0x6D) Reserved

Table 38-6. Select Bit Fields

Field	Description
	0110_1110 - (0x6E) Reserved 0110_1111 - (0x6F) Reserved 0111_0000 - (0x70) Reserved 0111_0001 - (0x71) Reserved 0111_0010 - (0x72) Reserved 0111_0011 - (0x73) Reserved 0111_0100 - (0x74) Reserved 0111_0101 - (0x75) Reserved 0111_0110 - (0x76) Reserved 0111_0111 - (0x77) Reserved 0111_1000 - (0x78) Reserved 0111_1001 - (0x79) Reserved 0111_1010 - (0x7A) Reserved 0111_1011 - (0x7B) Reserved 0111_1100 - (0x7C) Reserved 0111_1101 - (0x7D) Reserved 0111_1110 - (0x7E) Reserved 0111_1111 - (0x7F) Reserved

TRGMUX0 base address: 4102_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRGMUX DMAMUX0_A Register (DMAMUX0_A)	32	RW	0000_0000h
4h	TRGMUX DMAMUX0_B Register (DMAMUX0_B)	32	RW	0000_0000h
8h	TRGMUX LPIT0 Register (LPIT0)	32	RW	0000_0000h
10h	TRGMUX TPM0 Register (TPM0)	32	RW	0000_0000h
14h	TRGMUX TPM1 Register (TPM1)	32	RW	0000_0000h
18h	TRGMUX TPM2 Register (TPM2)	32	RW	0000_0000h
1Ch	TRGMUX TPM3 Register (TPM3)	32	RW	0000_0000h
20h	TRGMUX ADC0 Register (ADC0)	32	RW	0000_0000h
24h	TRGMUX ADC1 Register (ADC1)	32	RW	0000_0000h
28h	TRGMUX CMP0 Register (CMP0)	32	RW	0000_0000h
2Ch	TRGMUX CMP1 Register (CMP1)	32	RW	0000_0000h
30h	TRGMUX DAC0 Register (DAC0)	32	RW	0000_0000h
34h	TRGMUX DAC1 Register (DAC1)	32	RW	0000_0000h
38h	TRGMUX LPUART0 Register (LPUART0)	32	RW	0000_0000h
3Ch	TRGMUX LPUART1 Register (LPUART1)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
40h	TRGMUX LPUART2 Register (LPUART2)	32	RW	0000_0000h
44h	TRGMUX LPUART3 Register (LPUART3)	32	RW	0000_0000h
48h	TRGMUX LPI2C0 Register (LPI2C0)	32	RW	0000_0000h
4Ch	TRGMUX LPI2C1 Register (LPI2C1)	32	RW	0000_0000h
50h	TRGMUX LPI2C2 Register (LPI2C2)	32	RW	0000_0000h
54h	TRGMUX LPI2C3 Register (LPI2C3)	32	RW	0000_0000h
58h	TRGMUX LPSPI0 Register (LPSPI0)	32	RW	0000_0000h
5Ch	TRGMUX LPSPI1 Register (LPSPI1)	32	RW	0000_0000h
60h	TRGMUX FLEXIO0 Register (FLEXIO0)	32	RW	0000_0000h

38.4.1.2 TRGMUX DMAMUX0_A Register (DMAMUX0_A)

38.4.1.2.1 Offset

Register	Offset
DMAMUX0_A	0h

38.4.1.2.2 Function

This register is for the DMAMUX0_A module.

38.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.2.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.3 TRGMUX DMAMUX0_B Register (DMAMUX0_B)

38.4.1.3.1 Offset

Register	Offset
DMAMUX0_B	4h

38.4.1.3.2 Function

This register is for the DMAMUX0_B module.

38.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.3.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.4 TRGMUX LPIT0 Register (LPIT0)

38.4.1.4.1 Offset

Register	Offset
LPIT0	8h

38.4.1.4.2 Function

This register is for the LPIT0 module.

38.4.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.4.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.

Table continues on the next page...

Field	Function
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.5 TRGMUX TPM0 Register (TPM0)

38.4.1.5.1 Offset

Register	Offset
TPM0	10h

38.4.1.5.2 Function

This register is for the TPM0 module.

38.4.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.5.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.6 TRGMUX TPM1 Register (TPM1)

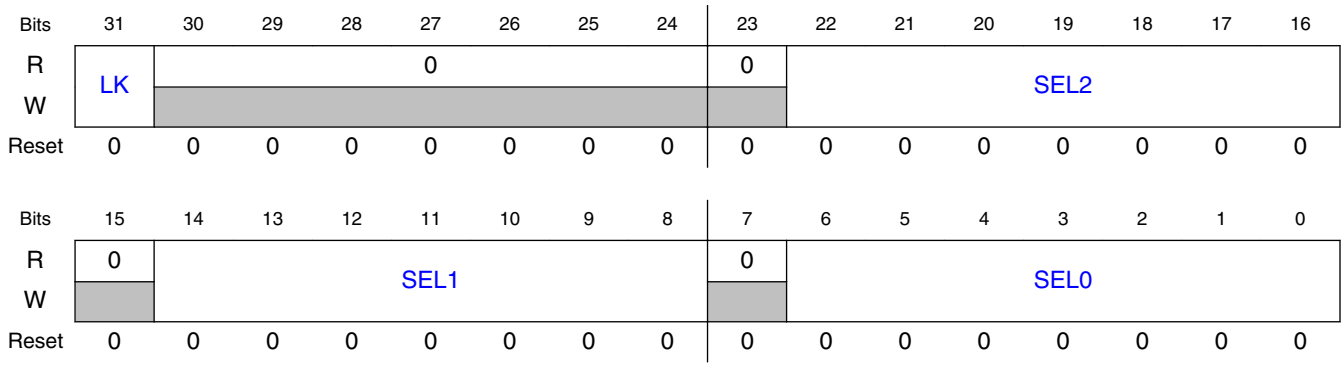
38.4.1.6.1 Offset

Register	Offset
TPM1	14h

38.4.1.6.2 Function

This register is for the TPM1 module.

38.4.1.6.3 Diagram



38.4.1.6.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.7 TRGMUX TPM2 Register (TPM2)

38.4.1.7.1 Offset

Register	Offset
TPM2	18h

38.4.1.7.2 Function

This register is for the TPM2 module.

38.4.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.7.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.8 TRGMUX TPM3 Register (TPM3)

38.4.1.8.1 Offset

Register	Offset
TPM3	1Ch

38.4.1.8.2 Function

This register is for the TPM3 module.

38.4.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.8.4 Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.9 TRGMUX ADC0 Register (ADC0)

38.4.1.9.1 Offset

Register	Offset
ADC0	20h

38.4.1.9.2 Function

This register is for the ADC0 module.

38.4.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.9.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.10 TRGMUX ADC1 Register (ADC1)

38.4.1.10.1 Offset

Register	Offset
ADC1	24h

38.4.1.10.2 Function

This register is for the ADC1 module.

38.4.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.10.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.

Table continues on the next page...

Field	Function
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.11 TRGMUX CMP0 Register (CMP0)

38.4.1.11.1 Offset

Register	Offset
CMP0	28h

38.4.1.11.2 Function

This register is for the CMP0 module.

38.4.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.11.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written.

Table continues on the next page...

Field	Function
	1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.12 TRGMUX CMP1 Register (CMP1)

38.4.1.12.1 Offset

Register	Offset
CMP1	2Ch

38.4.1.12.2 Function

This register is for the CMP1 module.

38.4.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.12.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.13 TRGMUX DAC0 Register (DAC0)

38.4.1.13.1 Offset

Register	Offset
DAC0	30h

38.4.1.13.2 Function

This register is for the DAC0 module.

38.4.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.13.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.14 TRGMUX DAC1 Register (DAC1)

38.4.1.14.1 Offset

Register	Offset
DAC1	34h

38.4.1.14.2 Function

This register is for the DAC1 module.

38.4.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.14.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written.

Table continues on the next page...

Field	Function
	1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.15 TRGMUX LPUART0 Register (LPUART0)

38.4.1.15.1 Offset

Register	Offset
LPUART0	38h

38.4.1.15.2 Function

This register is for the LPUART0 module.

38.4.1.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.15.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.16 TRGMUX LPUART1 Register (LPUART1)

38.4.1.16.1 Offset

Register	Offset
LPUART1	3Ch

38.4.1.16.2 Function

This register is for the LPUART1 module.

38.4.1.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.16.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.17 TRGMUX LPUART2 Register (LPUART2)

38.4.1.17.1 Offset

Register	Offset
LPUART2	40h

38.4.1.17.2 Function

This register is for the LPUART2 module.

38.4.1.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.17.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written.

Table continues on the next page...

Field	Function
	1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.18 TRGMUX LPUART3 Register (LPUART3)

38.4.1.18.1 Offset

Register	Offset
LPUART3	44h

38.4.1.18.2 Function

This register is for the LPUART3 module.

38.4.1.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.18.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.19 TRGMUX LPI2C0 Register (LPI2C0)

38.4.1.19.1 Offset

Register	Offset
LPI2C0	48h

38.4.1.19.2 Function

This register is for the LPI2C0 module.

38.4.1.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.19.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.20 TRGMUX LPI2C1 Register (LPI2C1)

38.4.1.20.1 Offset

Register	Offset
LPI2C1	4Ch

38.4.1.20.2 Function

This register is for the LPI2C1 module.

38.4.1.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.20.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written.

Table continues on the next page...

Field	Function
	1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.21 TRGMUX LPI2C2 Register (LPI2C2)

38.4.1.21.1 Offset

Register	Offset
LPI2C2	50h

38.4.1.21.2 Function

This register is for the LPI2C2 module.

38.4.1.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.21.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.22 TRGMUX LPI2C3 Register (LPI2C3)

38.4.1.22.1 Offset

Register	Offset
LPI2C3	54h

38.4.1.22.2 Function

This register is for the LPI2C3 module.

38.4.1.22.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.22.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.23 TRGMUX LPSPI0 Register (LPSPI0)

38.4.1.23.1 Offset

Register	Offset
LPSPI0	58h

38.4.1.23.2 Function

This register is for the LPSPI0 module.

38.4.1.23.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.23.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written.

Table continues on the next page...

Field	Function
	1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.24 TRGMUX LPSPI1 Register (LPSPI1)

38.4.1.24.1 Offset

Register	Offset
LPSPI1	5Ch

38.4.1.24.2 Function

This register is for the LPSPI1 module.

38.4.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.24.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 —	This read-only bit field is reserved and always has the value 0.
15 —	This read-only bit field is reserved and always has the value 0.
14-8 —	This read-only bit field is reserved and always has the value 0.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.1.25 TRGMUX FLEXIO0 Register (FLEXIO0)

38.4.1.25.1 Offset

Register	Offset
FLEXIO0	60h

38.4.1.25.2 Function

This register is for the FLEXIO0 module.

38.4.1.25.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	SEL3							0	SEL2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL1							0	SEL0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.1.25.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23 —	This read-only bit field is reserved and always has the value 0.
22-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15 —	This read-only bit field is reserved and always has the value 0.
14-8	Trigger MUX Input 1 Source Select

Table continues on the next page...

Field	Function
SEL1	This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7 —	This read-only bit field is reserved and always has the value 0.
6-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2 TRGMUX register descriptions

38.4.2.1 TRGMUX Memory map

Table 38-7. Select Bit Fields

Field	Description
SELx	This read/write field is used to configure the MUX select for the peripheral trigger inputs. 0000_0000 - (0x00) - 0000_0001 - (0x01) FlexIO1 0000_0010 - (0x02) FlexIO1 0000_0011 - (0x03) FlexIO1 0000_0100 - (0x04) FlexIO1 0000_0101 - (0x05) FlexIO1 0000_0110 - (0x06) FlexIO1 0000_0111 - (0x07) FlexIO1 0000_1000 - (0x08) FlexIO1 0000_1001 - (0x09) TPM4 0000_1010 - (0x0A) TPM4 0000_1011 - (0x0B) TPM4 0000_1100 - (0x0C) TPM5 0000_1101 - (0x0D) TPM5 0000_1110 - (0x0E) TPM5 0000_1111 - (0x0F) TPM6 0001_0000 - (0x10) TPM6 0001_0001 - (0x11) TPM6 0001_0010 - (0x12) TPM7 0001_0011 - (0x13) TPM7 0001_0100 - (0x14) TPM7

Table 38-7. Select Bit Fields

Field	Description
	0001_0101 - (0x15) LPUART4 Reserved
	0001_0110 - (0x16) LPUART4 Reserved
	0001_0111 - (0x17) LPUART4 Reserved
	0001_1000 - (0x18) LPUART5 Reserved
	0001_1001 - (0x19) LPUART5 Reserved
	0001_1010 - (0x1A) LPUART5 Reserved
	0001_1011 - (0x1B) LPUART6 Reserved
	0001_1100 - (0x1C) LPUART6 Reserved
	0001_1101 - (0x1D) LPUART6 Reserved
	0001_1110 - (0x1E) LPUART7 Reserved
	0001_1111 - (0x1F) LPUART7 Reserved
	0010_0000 - (0x20) LPUART7 Reserved
	0010_0001 - (0x21) LPI2C4
	0010_0010 - (0x22) LPI2C4
	0010_0011 - (0x23) LPI2C5
	0010_0100 - (0x24) LPI2C5
	0010_0101 - (0x25) LPI2C6
	0010_0110 - (0x26) LPI2C6
	0010_0111 - (0x27) LPI2C7
	0010_1000 - (0x28) LPI2C7
	0010_1001 - (0x29) LPSPI2
	0010_1010 - (0x2A) LPSPI2
	0010_1011 - (0x2B) LPSPI3
	0010_1100 - (0x2C) LPSPI3
	0010_1101 - (0x2D) PORT C
	0010_1110 - (0x2E) PORT D
	0010_1111 - (0x2F) PORT E
	0011_0000 - (0x30) PORT F
	0011_0001 - (0x31) USB0
	0011_0010 - (0x32) USB1
	0011_0011 - (0x33) LPIT1
	0011_0100 - (0x34) LPIT1
	0011_0101 - (0x35) LPIT1
	0011_0110 - (0x36) LPIT1
	0011_0111 - (0x37) Reserved
	0011_1000 - (0x38) Reserved
	0011_1001 - (0x39) Reserved
	0011_1010 - (0x3A) Reserved

Table 38-7. Select Bit Fields

Field	Description
	0011_1011 - (0x3B) Reserved 0011_1100 - (0x3C) Reserved 0011_1101 - (0x3D) Reserved 0011_1110 - (0x3E) Reserved 0011_1111 - (0x3F) Reserved

TRGMUX1 base address: 403A_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TRGMUX DMAMUX1_A Register (DMAMUX1_A)	32	RW	0000_0000h
4h	TRGMUX DMAMUX1_B Register (DMAMUX1_B)	32	RW	0000_0000h
8h	TRGMUX TPM4 Register (TPM4)	32	RW	0000_0000h
Ch	TRGMUX TPM5 Register (TPM5)	32	RW	0000_0000h
10h	TRGMUX TPM6 Register (TPM6)	32	RW	0000_0000h
14h	TRGMUX TPM7 Register (TPM7)	32	RW	0000_0000h
18h	TRGMUX LPUART4 Register (LPUART4)	32	RW	0000_0000h
1Ch	TRGMUX LPUART5 Register (LPUART5)	32	RW	0000_0000h
20h	TRGMUX LPUART6 Register (LPUART6)	32	RW	0000_0000h
24h	TRGMUX LPUART7 Register (LPUART7)	32	RW	0000_0000h
28h	TRGMUX LPI2C4 Register (LPI2C4)	32	RW	0000_0000h
2Ch	TRGMUX LPI2C5 Register (LPI2C5)	32	RW	0000_0000h
30h	TRGMUX LPI2C6 Register (LPI2C6)	32	RW	0000_0000h
34h	TRGMUX LPI2C7 Register (LPI2C7)	32	RW	0000_0000h
38h	TRGMUX LPSPI2 Register (LPSPI2)	32	RW	0000_0000h
3Ch	TRGMUX LPSPI3 Register (LPSPI3)	32	RW	0000_0000h
40h	TRGMUX FLEXIO1 Register (FLEXIO1)	32	RW	0000_0000h
44h	TRGMUX LPIT1 Register (LPIT1)	32	RW	0000_0000h

38.4.2.2 TRGMUX DMAMUX1_A Register (DMAMUX1_A)

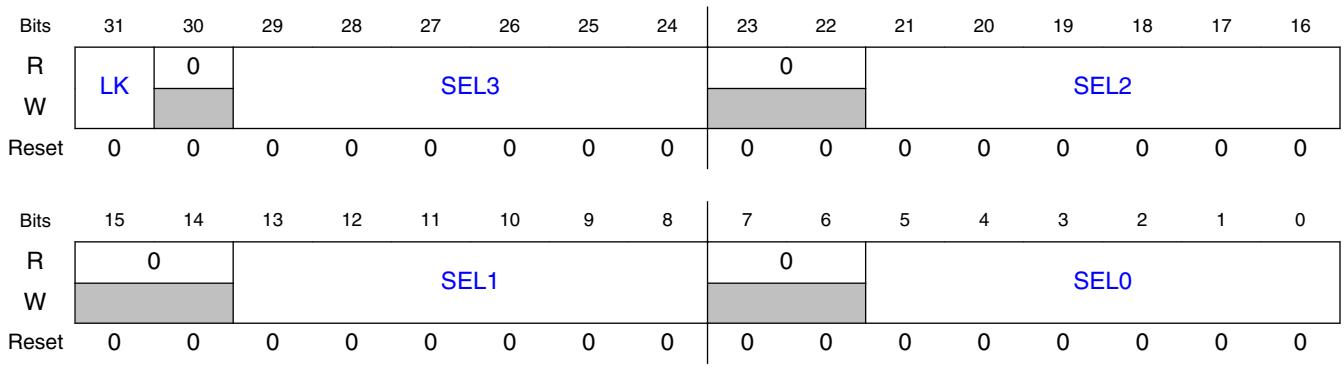
38.4.2.2.1 Offset

Register	Offset
DMAMUX1_A	0h

38.4.2.2.2 Function

This register is for the DMAMUX1_A module.

38.4.2.2.3 Diagram



38.4.2.2.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.3 TRGMUX DMAMUX1_B Register (DMAMUX1_B)

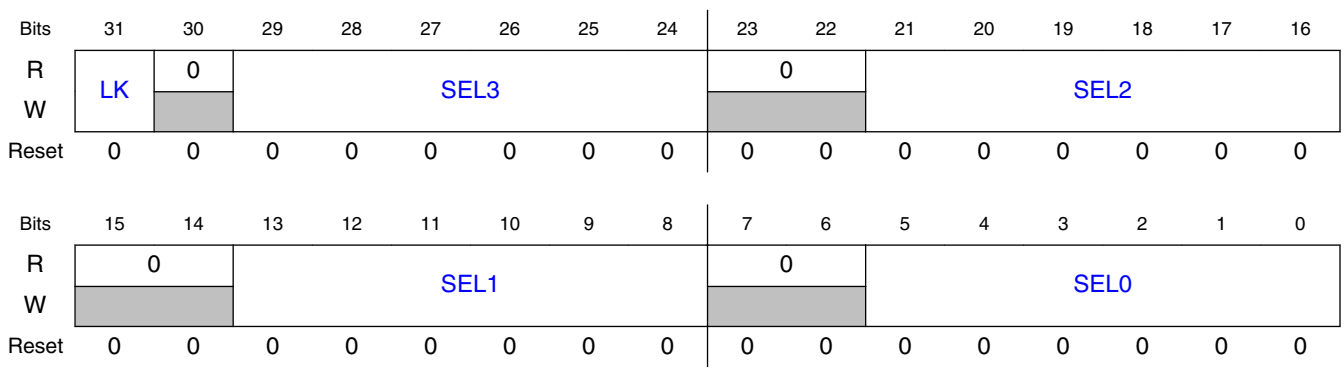
38.4.2.3.1 Offset

Register	Offset
DMAMUX1_B	4h

38.4.2.3.2 Function

This register is for the DMAMUX1_B module.

38.4.2.3.3 Diagram



38.4.2.3.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
29-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.4 TRGMUX TPM4 Register (TPM4)

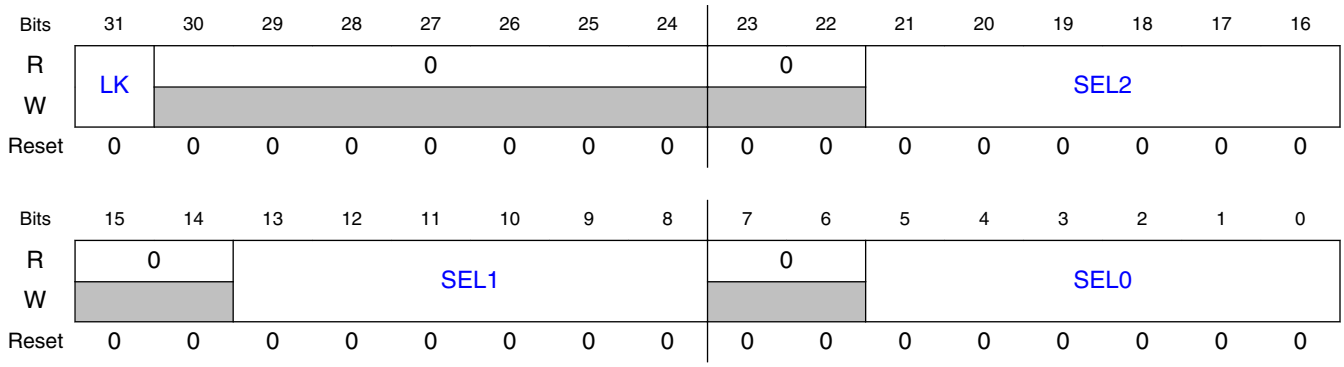
38.4.2.4.1 Offset

Register	Offset
TPM4	8h

38.4.2.4.2 Function

This register is for the TPM4 module.

38.4.2.4.3 Diagram



38.4.2.4.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.5 TRGMUX TPM5 Register (TPM5)

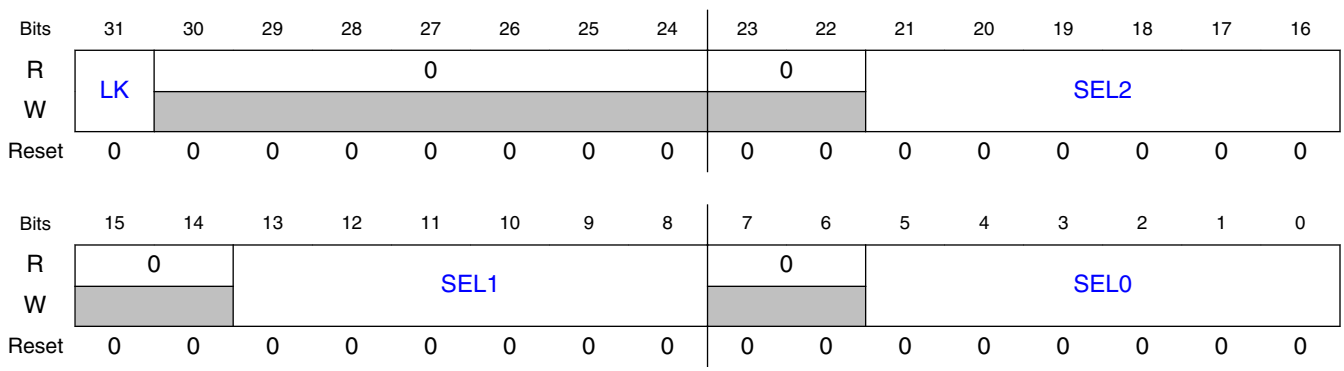
38.4.2.5.1 Offset

Register	Offset
TPM5	Ch

38.4.2.5.2 Function

This register is for the TPM5 module.

38.4.2.5.3 Diagram



38.4.2.5.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.6 TRGMUX TPM6 Register (TPM6)

38.4.2.6.1 Offset

Register	Offset
TPM6	10h

38.4.2.6.2 Function

This register is for the TPM6 module.

38.4.2.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0		SEL2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		SEL1						0		SEL0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.2.6.4 Fields

Field	Function
31	TRGMUX register lock.

Table continues on the next page...

Field	Function
LK	This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.7 TRGMUX TPM7 Register (TPM7)

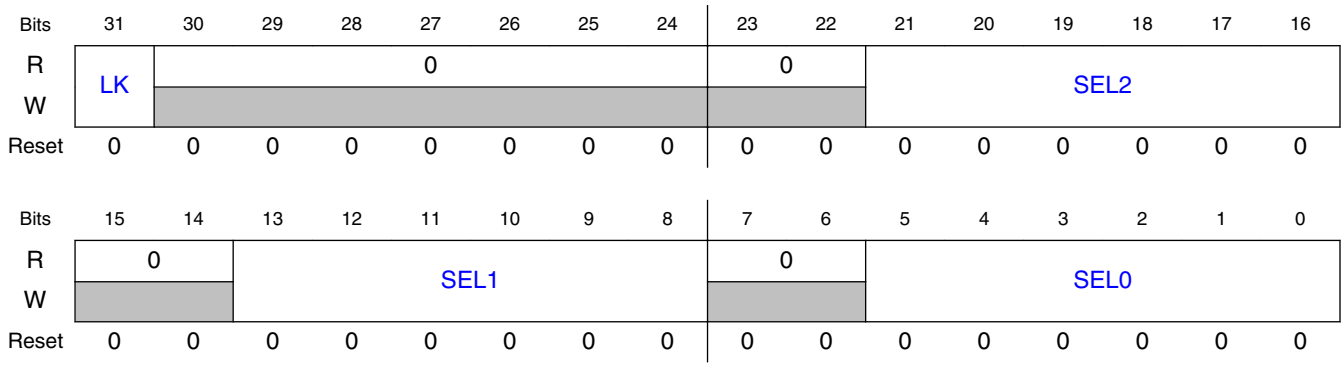
38.4.2.7.1 Offset

Register	Offset
TPM7	14h

38.4.2.7.2 Function

This register is for the TPM7 module.

38.4.2.7.3 Diagram



38.4.2.7.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.8 TRGMUX LPUART4 Register (LPUART4)

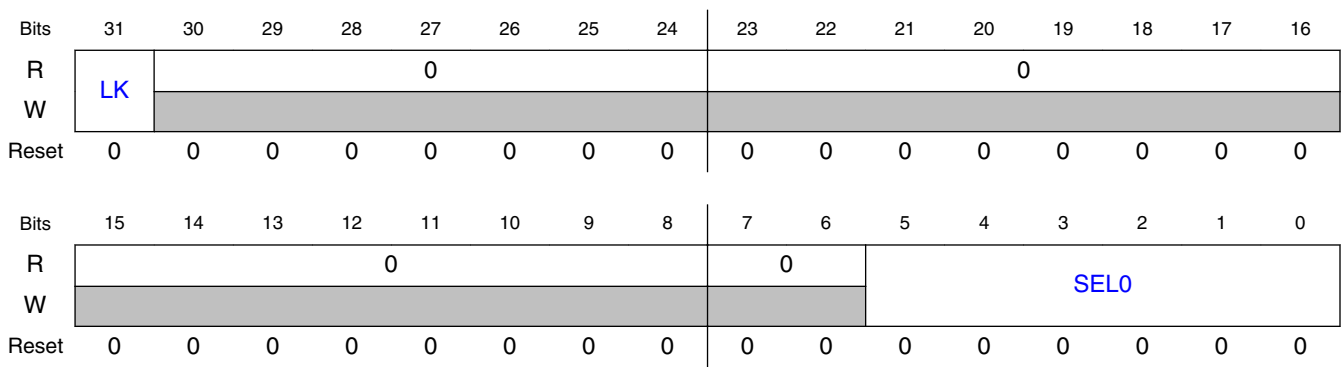
38.4.2.8.1 Offset

Register	Offset
LPUART4	18h

38.4.2.8.2 Function

This register is for the LPUART4 module.

38.4.2.8.3 Diagram



38.4.2.8.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.9 TRGMUX LPUART5 Register (LPUART5)

38.4.2.9.1 Offset

Register	Offset
LPUART5	1Ch

38.4.2.9.2 Function

This register is for the LPUART5 module.

38.4.2.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0		SEL0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.2.9.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.10 TRGMUX LPUART6 Register (LPUART6)

38.4.2.10.1 Offset

Register	Offset
LPUART6	20h

38.4.2.10.2 Function

This register is for the LPUART6 module.

38.4.2.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0		SEL0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.2.10.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.11 TRGMUX LPUART7 Register (LPUART7)

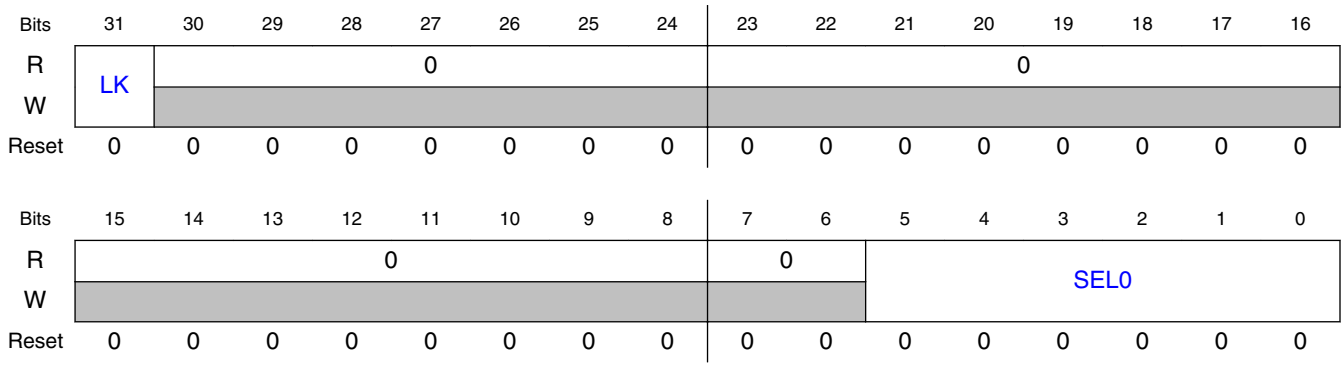
38.4.2.11.1 Offset

Register	Offset
LPUART7	24h

38.4.2.11.2 Function

This register is for the LPUART7 module.

38.4.2.11.3 Diagram



38.4.2.11.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.12 TRGMUX LPI2C4 Register (LPI2C4)

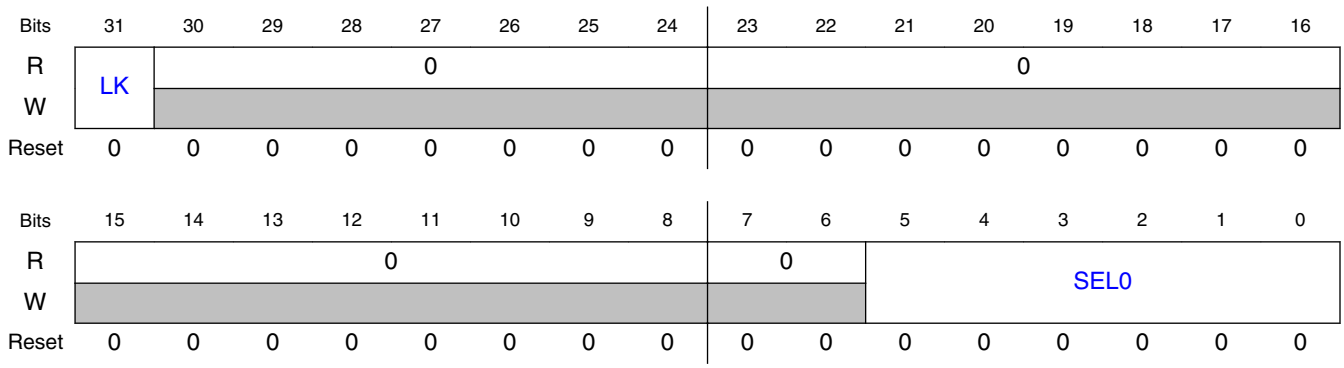
38.4.2.12.1 Offset

Register	Offset
LPI2C4	28h

38.4.2.12.2 Function

This register is for the LPI2C4 module.

38.4.2.12.3 Diagram



38.4.2.12.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.13 TRGMUX LPI2C5 Register (LPI2C5)

38.4.2.13.1 Offset

Register	Offset
LPI2C5	2Ch

38.4.2.13.2 Function

This register is for the LPI2C5 module.

38.4.2.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0		SEL0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.2.13.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.14 TRGMUX LPI2C6 Register (LPI2C6)

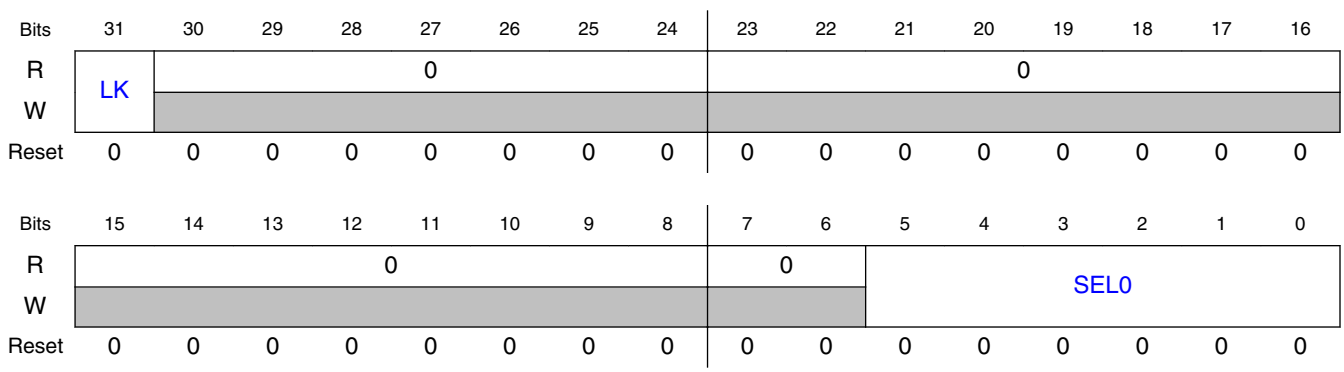
38.4.2.14.1 Offset

Register	Offset
LPI2C6	30h

38.4.2.14.2 Function

This register is for the LPI2C6 module.

38.4.2.14.3 Diagram



38.4.2.14.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.15 TRGMUX LPI2C7 Register (LPI2C7)

38.4.2.15.1 Offset

Register	Offset
LPI2C7	34h

38.4.2.15.2 Function

This register is for the LPI2C7 module.

38.4.2.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0							0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0		SEL0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.2.15.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written.

Table continues on the next page...

Field	Function
	1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.16 TRGMUX LPSPI2 Register (LPSPI2)

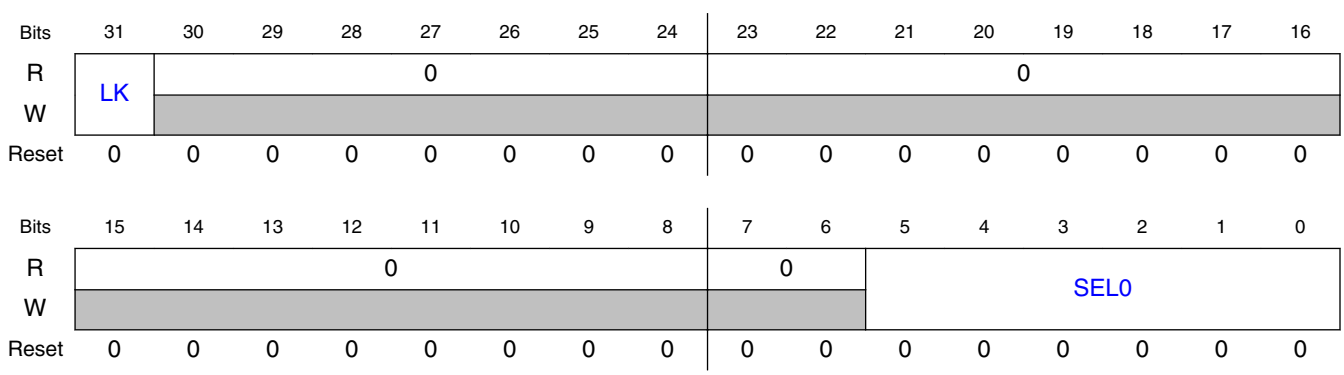
38.4.2.16.1 Offset

Register	Offset
LPSPI2	38h

38.4.2.16.2 Function

This register is for the LPSPI2 module.

38.4.2.16.3 Diagram



38.4.2.16.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.17 TRGMUX LPSPI3 Register (LPSPI3)

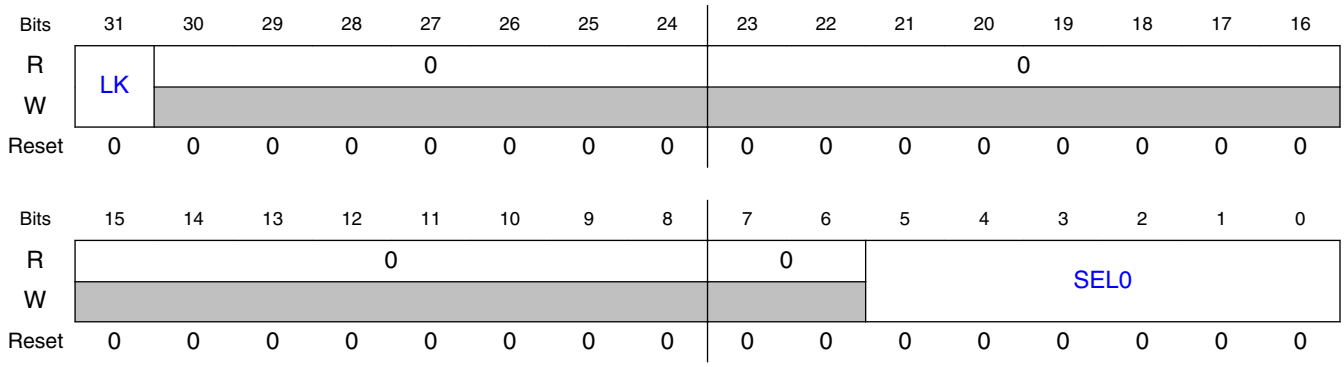
38.4.2.17.1 Offset

Register	Offset
LPSPI3	3Ch

38.4.2.17.2 Function

This register is for the LPSPI3 module.

38.4.2.17.3 Diagram



38.4.2.17.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30-24 —	This read-only bit field is reserved and always has the value 0.
23-16 —	This read-only bit field is reserved and always has the value 0.
15-8 —	This read-only bit field is reserved and always has the value 0.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.18 TRGMUX FLEXIO1 Register (FLEXIO1)

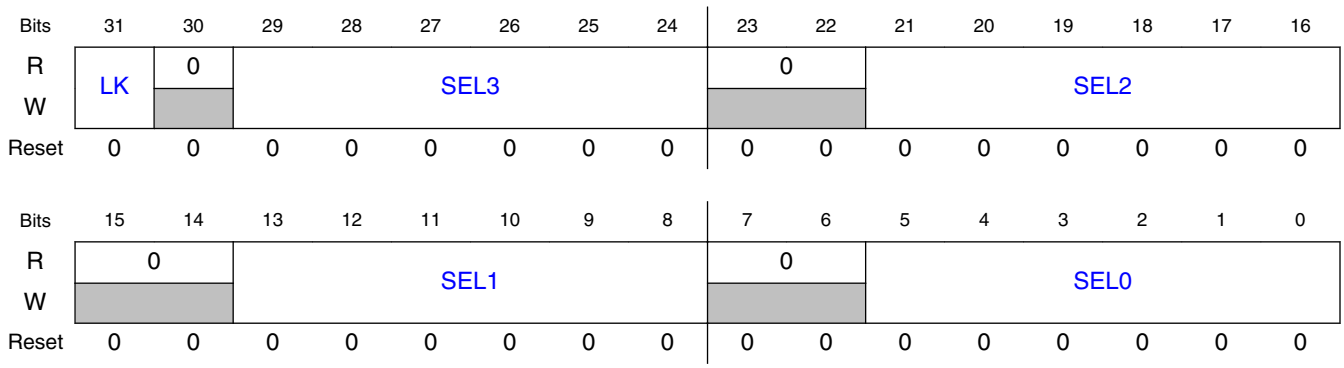
38.4.2.18.1 Offset

Register	Offset
FLEXIO1	40h

38.4.2.18.2 Function

This register is for the FLEXIO1 module.

38.4.2.18.3 Diagram



38.4.2.18.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30 —	This read-only bit field is reserved and always has the value 0.
29-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

38.4.2.19 TRGMUX LPIT1 Register (LPIT1)

38.4.2.19.1 Offset

Register	Offset
LPIT1	44h

38.4.2.19.2 Function

This register is for the LPIT1 module.

38.4.2.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LK	0	SEL3						0		SEL2					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		SEL1						0		SEL0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

38.4.2.19.4 Fields

Field	Function
31 LK	TRGMUX register lock. This bit shows whether the register can be written or not. The LK bit can only be written once after any system reset. Once LK is set, the SELx bits in this TRGMUX register cannot be changed until the next system reset clears LK. 0b - Register can be written. 1b - Register cannot be written until the next system Reset.
30	This read-only bit field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
—	
29-24 SEL3	Trigger MUX Input 3 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 3. For the field setting definitions, see Memory map and register definition.
23-22 —	This read-only bit field is reserved and always has the value 0.
21-16 SEL2	Trigger MUX Input 2 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 2. For the field setting definitions, see Memory map and register definition.
15-14 —	This read-only bit field is reserved and always has the value 0.
13-8 SEL1	Trigger MUX Input 1 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 1. For the field setting definitions, see Memory map and register definition.
7-6 —	This read-only bit field is reserved and always has the value 0.
5-0 SEL0	Trigger MUX Input 0 Source Select This read/write bit field is used to configure the MUX select for peripheral trigger input 0. For the field setting definitions, see Memory map and register definition.

Chapter 39
Port Control (PCTL)

39.1 Chip-specific PCTL information

Table 39-1. Reference links to related information

Topic	Related module	Reference
Full description	PCTL	PCTL
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

39.1.1 Port Control

The Port Control (PORT) module provides support for port control and digital filtering functions. All functions can be configured independently for each pin in the 32-bit port and affect the pin regardless of its pin multiplexing state. There is one instance of the PORT module for each port. Not all pins within each port are implemented on a specific device.

Table 39-2. PORT configuration

Parameter	Description
Name	Port Control
Instances	6 (PortA-F)
Configurable features	NA
Interface speed	NA
External I/O pins	See the attached IOMUXC spreadsheet for pin details

39.2 Introduction

39.2.1 Overview

The Port Control and Interrupt (PORT) module provides support for port control, and external interrupt functions.

Most functions can be configured independently for each pin in the 32-bit port and affect the pin regardless of its pin muxing state.

There is one instance of the PORT module for each port. Not all pins within each port are implemented on a specific device.

39.2.2 Features

The PORT module has the following features:

- Pin interrupt
  - Interrupt flag and enable registers for each pin
  - Support for edge sensitive (rising, falling, both) or level sensitive (low, high) configured per pin
  - Support for interrupt or DMA request configured per pin
  - Asynchronous wake-up in low-power modes
  - Pin interrupt is functional in all digital pin muxing modes
  - Peripheral trigger output (active high, low) configured per pin
- Port control
  - Individual mux control field supporting analog or pin disabled, GPIO, and up to 6 chip-specific digital functions
  - Pad configuration fields are functional in all digital pin muxing modes

39.2.3 Modes of operation

39.2.3.1 Run mode

In Run mode, the PORT operates normally.

39.2.3.2 Wait mode

In Wait mode, PORT continues to operate normally and may be configured to exit the Low-Power mode if an enabled interrupt is detected. DMA requests are still generated during the Wait mode, but do not cause an exit from the Low-Power mode.

39.2.3.3 Stop mode

In Stop mode, the PORT can be configured to exit the Low-Power mode via an asynchronous wake-up signal if an enabled interrupt is detected.

In Stop mode, the PORT configuration is static.

39.2.3.4 Debug mode

In Debug mode, PORT operates normally.

39.3 External signal description

The table found here describes the PORT external signal.

Table 39-3. Signal properties

Name	Function	I/O	Reset	Pull
PORTx[31:0]	External interrupt	I/O	0	-

NOTE

Not all pins within each port are implemented on each device.

39.3.1 Detailed signal description

The table found here contains the detailed signal description for the PORT interface.

Table 39-4. PORT interface—detailed signal description

Signal	I/O	Description	
PORTx[31:0]	I/O	External interrupt.	
		State meaning	Asserted—pin is logic 1. Negated—pin is logic 0.

Table continues on the next page...

Table 39-4. PORT interface—detailed signal description (continued)

Signal	I/O	Description	
		Timing	Assertion—may occur at any time and can assert asynchronously to the system clock. Negation—may occur at any time and can assert asynchronously to the system clock.

39.4 Memory map and register definition

Any read or write access to the PORT memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states.

39.4.1 PORT register descriptions

39.4.1.1 PORT Memory map

PORTA base address: 4103_F000h

PORTB base address: 4104_0000h

PORTC base address: 40AE_0000h

PORTD base address: 40AF_0000h

PORTE base address: 40B0_0000h

PORTF base address: 40B1_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 7Ch	Pin Control Register a (PCR0 - PCR31)	32	RW	0000_0000h
80h	Global Pin Control Low Register (GPCLR)	32	WORZ	0000_0000h
84h	Global Pin Control High Register (GPCHR)	32	WORZ	0000_0000h
88h	Global Interrupt Control Low Register (GICLR)	32	WORZ	0000_0000h
8Ch	Global Interrupt Control High Register (GICHR)	32	WORZ	0000_0000h
A0h	Interrupt Status Flag Register (ISFR)	32	W1C	0000_0000h

39.4.1.2 Pin Control Register a (PCR0 - PCR31)

39.4.1.2.1 Offset

For a = 0 to 31:

Register	Offset
PCRa	0h + (a × 4h)

39.4.1.2.2 Function

NOTE

See the Signal Multiplexing and Pin Assignment chapter for the reset values for the pins on this device.

See the Chip specific PORT information section for details on the available functions for each pin.

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

NOTE

Each module instance supports a different number of registers.

Register supported	Register not supported
PORTA_PCR0–PCR31	—
PORTB_PCR0–PCR19	PORTB_PCR20–PCR31
PORTC_PCR0–PCR19	PORTC_PCR20–PCR31

Table continues on the next page...

Register supported	Register not supported
PORTD_PCR0–PCR11	PORTD_PCR12–PCR31
PORTE_PCR0–PCR15	PORTE_PCR16–PCR31
PORTF_PCR0–PCR19	PORTF_PCR20–PCR31

39.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0						ISF	0	0			IRQC			
W								W1C								
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0			0			0	0	0	0	0	0	0	0
W																
Reset	See Register reset values.															

39.4.1.2.4 Register reset values

Register	Reset value
PCR0–PCR11	PORTA–PORTF: 0000_0000h
PCR12–PCR15	PORTA–PORTC: 0000_0000h PORTD: Register not supported PORTE,PORTF: 0000_0000h
PCR16–PCR19	PORTA–PORTC: 0000_0000h PORTD,PORTE: Register not supported PORTF: 0000_0000h
PCR20–PCR31	0000_0000h

39.4.1.2.5 Fields

Field	Function
31 —	Reserved.
30-25 —	Reserved
24 ISF	Interrupt Status Flag The pin interrupt configuration is valid in all digital pin muxing modes. 0b - Configured interrupt is not detected. 1b - Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.
23 —	Reserved.
22-20 —	Reserved
19-16 IRQC	Interrupt Configuration The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt, DMA request or trigger as follows: 0000b - Interrupt Status Flag (ISF) is disabled. 0001b - ISF flag and DMA request on rising edge. 0010b - ISF flag and DMA request on falling edge. 0011b - ISF flag and DMA request on either edge. 0100b - Reserved. 0101b - Flag sets on rising edge. 0110b - Flag sets on falling edge. 0111b - Flag sets on either edge. 1000b - ISF flag and Interrupt when logic 0. 1001b - ISF flag and Interrupt on rising-edge. 1010b - ISF flag and Interrupt on falling-edge. 1011b - ISF flag and Interrupt on either edge. 1100b - ISF flag and Interrupt when logic 1. 1101b - Enable active high trigger output, flag is disabled. [The trigger output goes to the trigger mux, which allows pins to trigger other peripherals (configurable polarity; 1 pin per port; if multiple pins are configured, then they are ORed together to create the trigger)] 1110b - Enable active low trigger output, flag is disabled. 1111b - Reserved.
15 —	Reserved.
14 —	Reserved.
13-11 —	Reserved
10-8 —	Reserved

Table continues on the next page...

Field	Function
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

39.4.1.3 Global Pin Control Low Register (GPCLR)

39.4.1.3.1 Offset

Register	Offset
GPCLR	80h

39.4.1.3.2 Function

Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

39.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GPWE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	GPWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.4.1.3.4 Fields

Field	Function
31-16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (15 through 0) bits update with the value in GPWD. 0b - Corresponding lower 16-bits of Pin Control Register is not updated with the value in GPWD. 1b - Corresponding lower 16-bits of Pin Control Register is updated with the value in GPWD.
15-0 GPWD	Global Pin Write Data GPWD field is written to PCRa[15:0] if GPWEn = 1 (a = n).

39.4.1.4 Global Pin Control High Register (GPCHR)

39.4.1.4.1 Offset

Register	Offset
GPCHR	84h

39.4.1.4.2 Function

Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

39.4.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GPWE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	GPWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.4.1.4.4 Fields

Field	Function
31-16 GPWE	Global Pin Write Enable Selects which Pin Control Registers (31 through 16) update with the value in GPWD. 0b - Corresponding lower 16-bits of Pin Control Register is not updated with the value in GPWD. 1b - Corresponding lower 16-bits of Pin Control Register is updated with the value in GPWD.
15-0 GPWD	Global Pin Write Data GPWD field is written to PCRa[15:0] if GPWEn = 1 (a = n + 16).

39.4.1.5 Global Interrupt Control Low Register (GICLR)

39.4.1.5.1 Offset

Register	Offset
GICLR	88h

39.4.1.5.2 Function

Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

39.4.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GIWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	GIWE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.4.1.5.4 Fields

Field	Function
31-16	Global Interrupt Write Data
GIWD	Write value that is written to upper 16-bits of the Pin Control Registers bits that are selected by GIWE.
15-0	Global Interrupt Write Enable
GIWE	Selects which upper 16-bits of Pin Control Registers (15 through 0) update with the value in GIWD. 0b - Corresponding upper 16-bits of Pin Control Register is not updated with the value in GIWD. 1b - Corresponding upper 16-bits of Pin Control Register is updated with the value in GIWD.

39.4.1.6 Global Interrupt Control High Register (GICHR)

39.4.1.6.1 Offset

Register	Offset
GICHR	8Ch

39.4.1.6.2 Function

Only 32-bit writes are supported to this register, any 16-bit or 8-bit writes are ignored.

39.4.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	GIWD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	GIWE															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.4.1.6.4 Fields

Field	Function
31-16	Global Interrupt Write Data
GIWD	Write value that is written to upper 16-bits of the Pin Control Registers that are selected by GIWE.
15-0	Global Interrupt Write Enable
GIWE	Selects which upper 16-bits of Pin Control Registers (31 through 16) update with the value in GIWD. 0b - Corresponding upper 16-bits of Pin Control Register is not updated with the value in GIWD. 1b - Corresponding upper 16-bits of Pin Register is updated with the value in GIWD.

39.4.1.7 Interrupt Status Flag Register (ISFR)

39.4.1.7.1 Offset

Register	Offset
ISFR	A0h

39.4.1.7.2 Function

The pin interrupt configuration is valid in all digital pin muxing modes. The Interrupt Status Flag for each pin is also visible in the corresponding Pin Control Register, and each flag can be cleared in either location.

39.4.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ISF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

39.4.1.7.4 Fields

Field	Function
31-0 ISF	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt of the same number as the field. 0b - Configured interrupt is not detected. 1b - Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic 1 is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.

39.5 Functional description

39.5.1 Pin control

Each port pin has a corresponding Pin Control register, PORT_PCRn, associated with it. The Pin Control register configures the following functions for each pin within the 32-bit port.

- Pullup enable on selected pins
- Pin Muxing mode

The functions apply across all digital pin muxing modes and individual peripherals do not override the configuration in the Pin Control register. For example, if an I^2C function is enabled on a pin, that does not override the pullup configuration for that pin.

When the Pin Muxing mode is configured for analog or is disabled, all the digital functions on that pin are disabled. This includes the pullup and pulldown enables, output buffer enable, input buffer enable, and passive filter enable.

The configuration of each Pin Control register is retained when the PORT module is disabled.

Whenever a pin is configured in any digital pin muxing mode, the input buffer for that pin is enabled allowing the pin state to be read via the corresponding GPIO Port Data Input Register (GPIO_PDIR) or allowing a pin interrupt or DMA request to be generated. If a pin is ever floating when its input buffer is enabled, then this can cause an increase in power consumption and must be avoided. A pin can be floating due to an input pin that is not connected or an output pin that has tri-stated (output buffer is disabled).

Enabling the internal pull resistor (or implementing an external pull resistor) will ensure a pin does not float when its input buffer is enabled; note that the internal pull resistor is automatically disabled whenever the output buffer is enabled allowing the Pull Enable bit to remain set. Configuring the Pin Muxing mode to disabled or analog will disable the pin's input buffer and results in the lowest power consumption.

39.5.2 Global pin control

The two global pin control registers allow a single register write to update the lower 16-bits of the Pin Control Register for up to 16 pins, all with the same value.

The global pin control registers are designed to enable software to quickly configure multiple pins within the same port with the same peripheral function.

The global pin control registers are write-only registers, that always read as 0.

39.5.3 Global interrupt control

The two global interrupt control registers allow a single register write to update the upper 16-bits of the Pin Control Register for up to 16 pins, all with the same value.

The global interrupt control registers are designed to enable software to quickly configure multiple pins within the same port with the same interrupt configuration.

The global interrupt control registers are write-only registers and always read as 0.

39.5.4 External interrupts

The external interrupt capability of the PORT module is available in all digital pin muxing modes provided the PORT module is enabled.

Each pin can be individually configured for any of the following external interrupt modes:

- Interrupt disabled, default out of reset
- Rising edge flag (for software polling)
- Falling edge flag (for software polling)
- Rising and falling edge flag (for software polling)
- Active high level peripheral trigger (status flag disabled)
- Active low level peripheral trigger (status flag disabled)
- Active high level sensitive interrupt
- Active low level sensitive interrupt
- Rising edge sensitive interrupt
- Falling edge sensitive interrupt
- Rising and falling edge sensitive interrupt
- Rising edge sensitive DMA request
- Falling edge sensitive DMA request
- Rising and falling edge sensitive DMA request

The interrupt status flag is set when the configured edge or level is detected on the pin . When not in Stop mode, the input is first synchronized to the bus clock to detect the configured level or edge transition.

The PORT module generates a single pin interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that port. The interrupt negates after the interrupt status flags for all enabled interrupts have been cleared by writing a logic 1 to the ISF flag in either the PORT_ISFR or PORT_PCRn registers.

The PORT module generates a single DMA request that asserts when the interrupt status flag is set for any enabled DMA request in that port. The DMA request negates after the DMA transfer is completed, because that clears the interrupt status flags for all enabled DMA requests.

During Stop mode, the interrupt status flag for any enabled interrupt is asynchronously set if the required level or edge is detected. This also generates an asynchronous wake-up signal to exit the Low-Power mode.

The PORT module generates a single peripheral trigger output that asserts if any pin configured for active high trigger is logic one, or any pin triggered for active low trigger is logic zero. The peripheral trigger output asynchronously updates from the value on the configured pins.

Chapter 40
Rapid General-Purpose Input and Output with 2 Ports (RGPIO2P)

40.1 Chip-specific RGPIO2P information

Table 40-1. Reference links to related information

Topic	Related module	Reference
Full description	RGPIO2P	RGPIO2P
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

40.1.1 RGPIO2P

The Rapid General-Purpose Input and Output with 2 Ports (RGPIO2P) is similar to the RGPIO module, except it has an IPS port and AHB port. When connecting to the AHB interface, the processor can access RGPIO2P in zero-wait-state.

Table 40-2. RGPIO2P configuration

Parameter	Description
Name	RGPIO2P
Instances	2
Interface speed	RGPIO2P0-1: Set by I/O max
External I/O pins	PTAx, PTBx, PTCx, PTDx, PTEx, and PTFx. See IOMUXC spreadsheet attached with this document for details.

40.1.2 RGPIO2P Port implementation and instantiation in i.MX 7ULP

This device has two instances of RGPIO2P: one in CM4 platform and the other in CA7 platform. In total, RGPIO2P implements 6 ports, A-F. The following figure illustrates the implementation of GPIOA-F through IOMUXC.

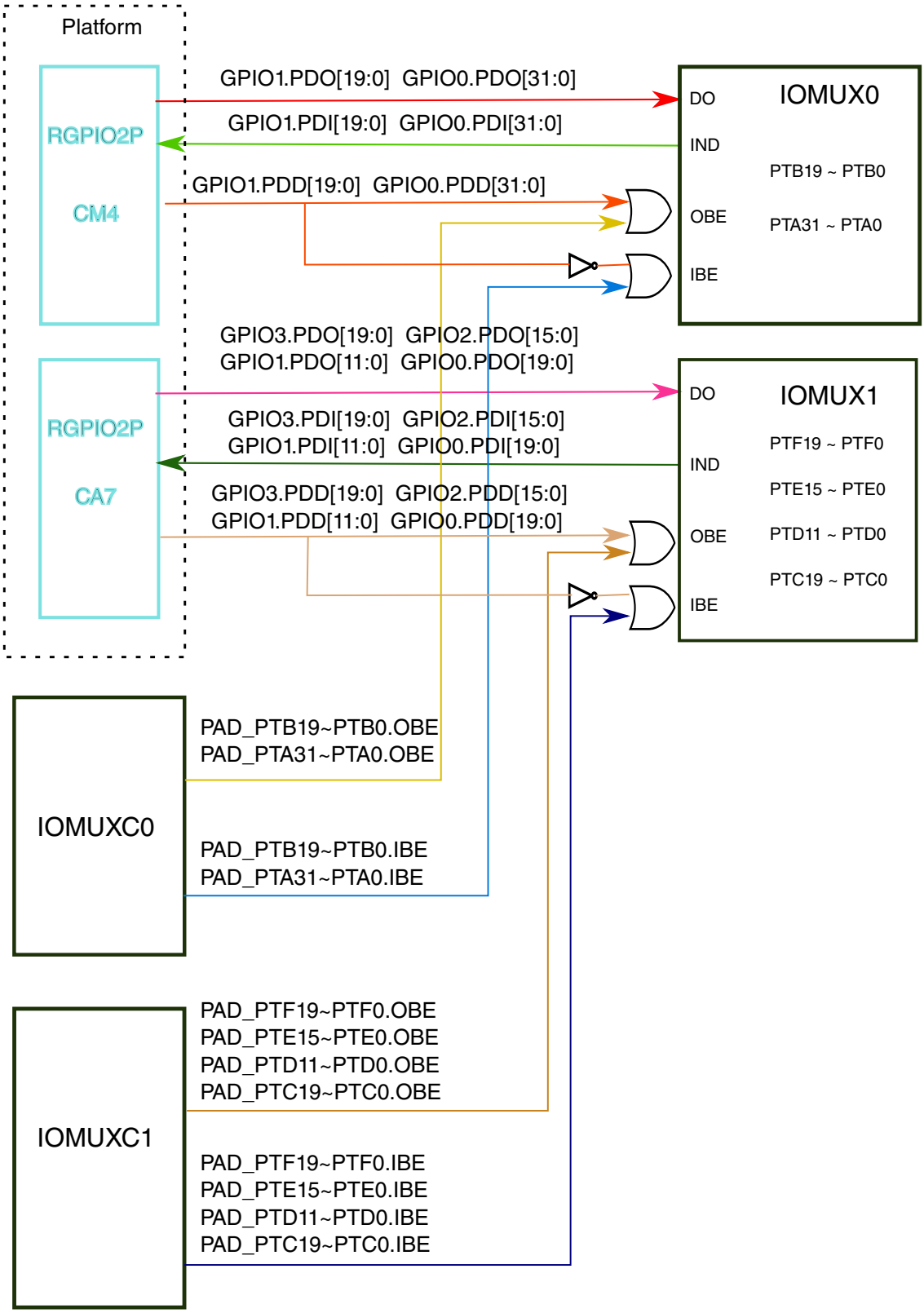


Figure 40-1. RGPIO2P port implementation in dual core

40.1.3 Port interface domain

XRDC protection is available for RGPIO2P for having independent access attributes for each interface domain. See the register section for more details.

- IPS interface
- IOPORT interface
- AHB interface
- APB interface

40.1.4 Port description implementation on this device

On this device, there are two types of GPIOs:

- Fail-safe GPIOs (FSGPIO) on ports A, B, C, E and F can have the GPIO port power supply removed (VDD_PTA, VDD_PTB, etc) and the inputs will not draw any current from either the VDD or ground.
- Standard GPIOs (STGPIO) on Port D are not fail-safe.

The following table depicts the number of ports along with the GPIOs, available on this device.

NOTE

The RGPIO2P has 2 access ports: one is connected to AIPS-Lite Bus and the other is connected to AHB Bus. GPIOA and GPIOB are accessible by both AHB and AIPS buses and hence have two base addresses. So, the user can access the RGPIO2P through AIPS-Lite0 using address 0x4100F000 for GPIOA and 0x4100F040 for GPIOB and access the RGPIO2P through AHB bus using address 0xF9000000 for GPIOA and 0xF9000040 for GPIOB.

Port name	Base address	Voltage (V)	Domain	Number of GPIOs/ Pins
GPIOA	0x4100_F000[1]	3.3/1.8	CM4	32
GPIOB	0x4100_F040[2]	1.8	CM4	20
GPIOC	0x400F_0000	3.3/1.8	CA7	20
GPIOD	0x400F_0040	3.3/1.8	CA7	12
GPIOE	0x400F_0080	3.3/1.8	CA7	16
GPIOF	0x400F_00C0	3.3/1.8	CA7	20

1. 0xF9000000 through the AHB bus
2. 0xF9000040 through AHB bus

40.2 Introduction

The general-purpose input and output (GPIO) module communicates to the processor core via a zero wait state interface (IOPORT) for maximum pin performance.The general-purpose input and output (GPIO) module is also accessible via the peripheral bus. The GPIO registers support 8-bit, 16-bit or 32-bit accesses.

The GPIO data direction and output data registers control the direction and output data of each pin when the pin is configured for the GPIO function. The GPIO input data register displays the logic value on each pin when the pin is configured for any digital function, provided the corresponding Port Control and Interrupt module for that pin is enabled.

Efficient bit manipulation of the general-purpose outputs is supported through the addition of set, clear, and toggle write-only registers for each port output data register.

40.2.1 Features

Features of the GPIO module include:

- Port Data Input register visible in all digital pin-multiplexing modes
- Port Data Output register with corresponding set/clear/toggle registers
- Port Data Direction register
- Zero wait state access to GPIO registers through IOPORT

NOTE

The GPIO module is clocked by system clock.

40.2.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

Table 40-3. Modes of operation

Modes of operation	Description
Run	The GPIO module operates normally.
Wait	The GPIO module operates normally.
Stop	The GPIO module is disabled.
Debug	The GPIO module operates normally.

40.2.3 GPIO signal descriptions

Table 40-4. GPIO signal descriptions

GPIO signal descriptions	Description	I/O
PORTA31–PORTA0	General-purpose input/output	I/O
PORTB31–PORTB0	General-purpose input/output	I/O

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

40.2.3.1 Detailed signal description

Table 40-5. GPIO interface-detailed signal descriptions

Signal	I/O	Description	
PORTA31–PORTA0 PORTB31–PORTB0	I/O	General-purpose input/output	
		State meaning	Asserted: The pin is logic 1. Deasserted: The pin is logic 0.
		Timing	Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock. Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.

NOTE

Not all pins within each port are implemented on each device. See the chapter on signal multiplexing for the number of GPIO ports available in the device.

40.3 Memory map and register definition

The registers for each GPIO port occupy 64-byte of the memory map. Any read or write access to the GPIO slot outside this space results in a bus error.

NOTE

For simplicity, each GPIO port's registers appear with the same width of 32 bits, corresponding to 32 pins. The actual number of pins per port (and therefore the number of usable control bits per port register) is chip-specific. Refer to the chip-specific GPIO information to see the exact control bits for each port.

40.3.1 GPIO register descriptions

40.3.1.1 GPIO Memory map

GPIOA base address: 4100_F000h

GPIOB base address: 4100_F040h

Offset	Register	Width (In bits)	Access	Reset value
0h	Port Data Output Register (PDOR)	32	RW	0000_0000h
4h	Port Set Output Register (PSOR)	32	WORZ	0000_0000h
8h	Port Clear Output Register (PCOR)	32	WORZ	0000_0000h
Ch	Port Toggle Output Register (PTOR)	32	WORZ	0000_0000h
10h	Port Data Input Register (PDIR)	32	RO	0000_0000h
14h	Port Data Direction Register (PDDR)	32	RW	0000_0000h
20h - 2Ch	Port Byte Domain Access Control Register 0 (B0DACP0 - B3DACP0)	32	RW	0000_0000h

40.3.1.2 Port Data Output Register (PDOR)

40.3.1.2.1 Offset

Register	Offset
PDOR	0h

40.3.1.2.2 Function

This register configures the logic levels that are driven on each general-purpose output pin.

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

40.3.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.3.1.2.4 Fields

Field	Function
31-0 PDO	Port Data Output Register bits for unbonded pins return an undefined value when read. 0b - Logic level 0 is driven on pin, provided pin is configured for general-purpose output. 1b - Logic level 1 is driven on pin, provided pin is configured for general-purpose output.

40.3.1.3 Port Set Output Register (PSOR)

40.3.1.3.1 Offset

Register	Offset
PSOR	4h

40.3.1.3.2 Function

This register configures whether to set the fields of the PDOR.

40.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	PTSO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	PTSO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.3.1.3.4 Fields

Field	Function
31-0 PTSO	Port Set Output Writing to this register updates the contents of the corresponding bit in the PDOR as follows: 0b - Corresponding bit in PDORn does not change. 1b - Corresponding bit in PDORn is set to logic 1.

40.3.1.4 Port Clear Output Register (PCOR)

40.3.1.4.1 Offset

Register	Offset
PCOR	8h

40.3.1.4.2 Function

This register configures whether to clear the fields of PDOR.

40.3.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	PTCO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	PTCO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.3.1.4.4 Fields

Field	Function
31-0 PTCO	Port Clear Output Writing to this register updates the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0b - Corresponding bit in PDORn does not change. 1b - Corresponding bit in PDORn is cleared to logic 0.

40.3.1.5 Port Toggle Output Register (PTOR)

40.3.1.5.1 Offset

Register	Offset
PTOR	Ch

40.3.1.5.2 Function

This register toggles the logic levels that are driven on each general-purpose output pin.

40.3.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	PTTO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	PTTO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.3.1.5.4 Fields

Field	Function
31-0 PTTO	Port Toggle Output Writing to this register updates the contents of the corresponding bit in the PDOR as follows: 0b - Corresponding bit in PDORn does not change. 1b - Corresponding bit in PDORn is set to the inverse of its existing logic state.

40.3.1.6 Port Data Input Register (PDIR)

40.3.1.6.1 Offset

Register	Offset
PDIR	10h

40.3.1.6.2 Function

This register captures the logic levels that are driven into each general-purpose input pin.

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

40.3.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.3.1.6.4 Fields

Field	Function
31-0 PDI	Port Data Input Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update. 0b - Pin logic level is logic 0, or is not configured for use by digital function. 1b - Pin logic level is logic 1.

40.3.1.7 Port Data Direction Register (PDDR)

40.3.1.7.1 Offset

Register	Offset
PDDR	14h

40.3.1.7.2 Function

The PDDR configures the individual port pins for input or output.

40.3.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.3.1.7.4 Fields

Field	Function
31-0 PDD	Port Data Direction Configures individual port pins for input or output. 0b - Pin is configured as general-purpose input, for the GPIO function. 1b - Pin is configured as general-purpose output, for the GPIO function.

40.3.1.8 Port Byte Domain Access Control Register 0 (B0DACP0 - B3DACP0)

40.3.1.8.1 Offset

Register	Offset
B0DACP0	20h
B1DACP0	24h
B2DACP0	28h
B3DACP0	2Ch

40.3.1.8.2 Function

The domain access control policy:

Table 40-6. XRDC 1.0 ACP Policy

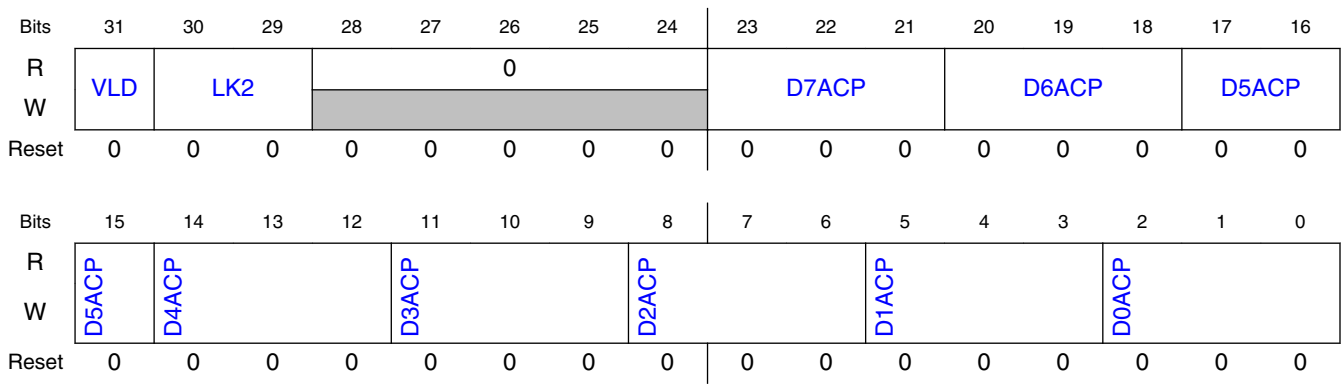
Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	R, W	R, W	R, W	R, W

Table continues on the next page...

Table 40-6. XRDC 1.0 ACP Policy (continued)

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
110	R, W	R, W	R, W	None
101	R, W	R, W	R	R
100	R, W	R, W	R	None
011	R, W	R, W	None	None
010	R, W	None	None	None
001	R	R	None	None
000	None	None	None	None

40.3.1.8.3 Diagram



40.3.1.8.4 Fields

Field	Function
31 VLD	Valid This field indicates the domain access control definition is valid. 0b - The DxACP assignment is invalid. 1b - The DxACP assignment is valid.
30-29 LK2	LK2 This 2-bit field provides a mechanism to limit writes to the D*ACP register to protect its contents. Once set, these bits individually remain asserted until the next reset. 00b - Entire DxACP can be written. 01b - Entire DxACP can be written. 10b - Domain x can only update the DxACP field; no other D*ACP fields can be written. 11b - DxACP is locked (read-only) until the next reset.
28-24 —	Reserved
23-21 D7ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.

Table continues on the next page...

Field	Function
20-18 D6ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
17-15 D5ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
14-12 D4ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
11-9 D3ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
8-6 D2ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
5-3 D1ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
2-0 D0ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.

40.4 FGPIO memory map and register definition

The GPIO registers of certain ports are also aliased to the IOPORT interface. See chip-specific information on FGPIO ports for details.

Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle. This aliased Fast GPIO memory map is called FGPIO.

Any read or write access to the FGPIO memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states, except error accesses which complete with one wait state.

NOTE

For simplicity, each FGPIO port's registers appear with the same width of 32 bits, corresponding to 32 pins. The actual number of pins per port (and therefore the number of usable control bits per port register) is chip-specific. Refer to the Chip Configuration chapter to see the exact control bits for the non-identical port instance.

40.4.1 GPIO register descriptions

40.4.1.1 GPIO Memory map

FGPIOA base address: F900_0000h

FGPIOB base address: F900_0040h

Offset	Register	Width (In bits)	Access	Reset value
0h	Port Data Output Register (PDOR)	32	RW	0000_0000h
4h	Port Set Output Register (PSOR)	32	WORZ	0000_0000h
8h	Port Clear Output Register (PCOR)	32	WORZ	0000_0000h
Ch	Port Toggle Output Register (PTOR)	32	WORZ	0000_0000h
10h	Port Data Input Register (PDIR)	32	RO	0000_0000h
14h	Port Data Direction Register (PDDR)	32	RW	0000_0000h
20h - 2Ch	Port Byte Domain Access Control Register 0 (B0DACP0 - B3DACP0)	32	RW	0000_0000h

40.4.1.2 Port Data Output Register (PDOR)

40.4.1.2.1 Offset

Register	Offset
PDOR	0h

40.4.1.2.2 Function

This register configures the logic levels that are driven on each general-purpose output pin.

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

40.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.4.1.2.4 Fields

Field	Function
31-0 PDO	Port Data Output Register bits for unbonded pins return an undefined value when read. 0b - Logic level 0 is driven on pin, provided pin is configured for general-purpose output. 1b - Logic level 1 is driven on pin, provided pin is configured for general-purpose output.

40.4.1.3 Port Set Output Register (PSOR)

40.4.1.3.1 Offset

Register	Offset
PSOR	4h

40.4.1.3.2 Function

This register configures whether to set the fields of the PDOR.

40.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	PTSO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	PTSO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.4.1.3.4 Fields

Field	Function
31-0 PTSO	Port Set Output Writing to this register updates the contents of the corresponding bit in the PDOR as follows: 0b - Corresponding bit in PDORn does not change. 1b - Corresponding bit in PDORn is set to logic 1.

40.4.1.4 Port Clear Output Register (PCOR)

40.4.1.4.1 Offset

Register	Offset
PCOR	8h

40.4.1.4.2 Function

This register configures whether to clear the fields of PDOR.

40.4.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	PTCO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	PTCO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.4.1.4.4 Fields

Field	Function
31-0 PTCO	Port Clear Output Writing to this register updates the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0b - Corresponding bit in PDORn does not change. 1b - Corresponding bit in PDORn is cleared to logic 0.

40.4.1.5 Port Toggle Output Register (PTOR)

40.4.1.5.1 Offset

Register	Offset
PTOR	Ch

40.4.1.5.2 Function

This register toggles the logic levels that are driven on each general-purpose output pin.

40.4.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	PTTO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	PTTO															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.4.1.5.4 Fields

Field	Function
31-0 PTTO	Port Toggle Output Writing to this register updates the contents of the corresponding bit in the PDOR as follows: 0b - Corresponding bit in PDORn does not change. 1b - Corresponding bit in PDORn is set to the inverse of its existing logic state.

40.4.1.6 Port Data Input Register (PDIR)

40.4.1.6.1 Offset

Register	Offset
PDIR	10h

40.4.1.6.2 Function

This register captures the logic levels that are driven into each general-purpose input pin.

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded pins not available in your package will default to DISABLE state for lowest power consumption.

40.4.1.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.4.1.6.4 Fields

Field	Function
31-0 PDI	Port Data Input Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update. 0b - Pin logic level is logic 0, or is not configured for use by digital function. 1b - Pin logic level is logic 1.

40.4.1.7 Port Data Direction Register (PDDR)

40.4.1.7.1 Offset

Register	Offset
PDDR	14h

40.4.1.7.2 Function

The PDDR configures the individual port pins for input or output.

40.4.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PDD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PDD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

40.4.1.7.4 Fields

Field	Function
31-0 PDD	Port Data Direction Configures individual port pins for input or output. 0b - Pin is configured as general-purpose input, for the GPIO function. 1b - Pin is configured as general-purpose output, for the GPIO function.

40.4.1.8 Port Byte Domain Access Control Register 0 (B0DACP0 - B3DACP0)

40.4.1.8.1 Offset

Register	Offset
B0DACP0	20h
B1DACP0	24h
B2DACP0	28h
B3DACP0	2Ch

40.4.1.8.2 Function

The domain access control policy:

Table 40-7. XRDC 1.0 ACP Policy

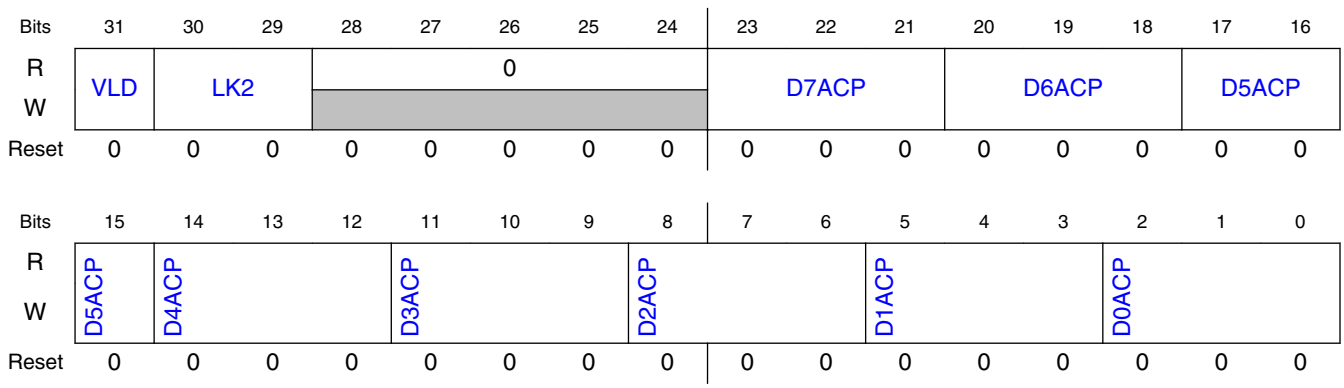
Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	R, W	R, W	R, W	R, W

Table continues on the next page...

Table 40-7. XRDC 1.0 ACP Policy (continued)

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
110	R, W	R, W	R, W	None
101	R, W	R, W	R	R
100	R, W	R, W	R	None
011	R, W	R, W	None	None
010	R, W	None	None	None
001	R	R	None	None
000	None	None	None	None

40.4.1.8.3 Diagram



40.4.1.8.4 Fields

Field	Function
31 VLD	Valid This field indicates the domain access control definition is valid. 0b - The DxACP assignment is invalid. 1b - The DxACP assignment is valid.
30-29 LK2	LK2 This 2-bit field provides a mechanism to limit writes to the D*ACP register to protect its contents. Once set, these bits individually remain asserted until the next reset. 00b - Entire DxACP can be written. 01b - Entire DxACP can be written. 10b - Domain x can only update the DxACP field; no other D*ACP fields can be written. 11b - DxACP is locked (read-only) until the next reset.
28-24 —	Reserved
23-21 D7ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.

Table continues on the next page...

Field	Function
20-18 D6ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
17-15 D5ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
14-12 D4ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
11-9 D3ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
8-6 D2ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
5-3 D1ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.
2-0 D0ACP	Domain Access Control Policy This 3-bit field defines the attributes required to access the corresponding byte of each port in each domain.

40.5 Functional description

40.5.1 General-purpose input

The logic state of each pin is available via the Port Data Input registers, provided the pin is configured for a digital function and the corresponding Port Control and Interrupt module is enabled.

The input pin synchronizers are shared with the Port Control and Interrupt module, so that if the corresponding Port Control and Interrupt module is disabled, then synchronizers are also disabled. This reduces power consumption when a port is not required for general-purpose input functionality.

40.5.2 General-purpose output

The logic state of each pin can be controlled via the port data output registers and port data direction registers, provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input/output.

If	Then
A pin is configured for the GPIO function and the corresponding port data direction register bit is clear.	The pin is configured as an input.
A pin is configured for the GPIO function and the corresponding port data direction register bit is set.	The pin is configured as an output and the logic state of the pin is equal to the corresponding port data output register.

To facilitate efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers exist to allow one or more outputs within one port to be set, cleared, or toggled from a single register write.

The corresponding Port Control and Interrupt module does not need to be enabled to update the state of the port data direction registers and port data output registers including the set/clear/toggle registers.

40.5.3 IOPORT

The GPIO registers of certain ports are also aliased to the IOPORT interface, see chip-specific information on FGPIO ports for details. Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle. If another bus master attempts to access the GPIO registers on the same cycle as an IOPORT access, then the DMA access will stall until any IOPORT accesses have completed.

Chapter 41
Debug

41.1 Overview

This chapter outlines the debug capabilities implemented within the i.MX 7ULP microprocessor. Debug features on i.MX 7ULP consist of static debug and real-time trace for the Arm cores, cross-domain triggering, and input and output of the control and trace data.

Following are the debug features implemented on this device:

- The Arm CoreSight Architecture adapted for software trace and debug.
- Software trace and full instruction trace are supported for A7 and M4 processors.
- A cross-trigger subsystem is included for cross domain triggering of debug resources
- Serial Wire Debug (JTAG/SWD) and Debug Access Port (DAP) to support debugger tools.
- 16/8/4-bit TPIU trace port to efficiently access trace information from the system.
- Data logging trace and debug information through USB or UART serial port.

41.2 System level debug architecture

The following figure depicts the system level debug architecture on this device.

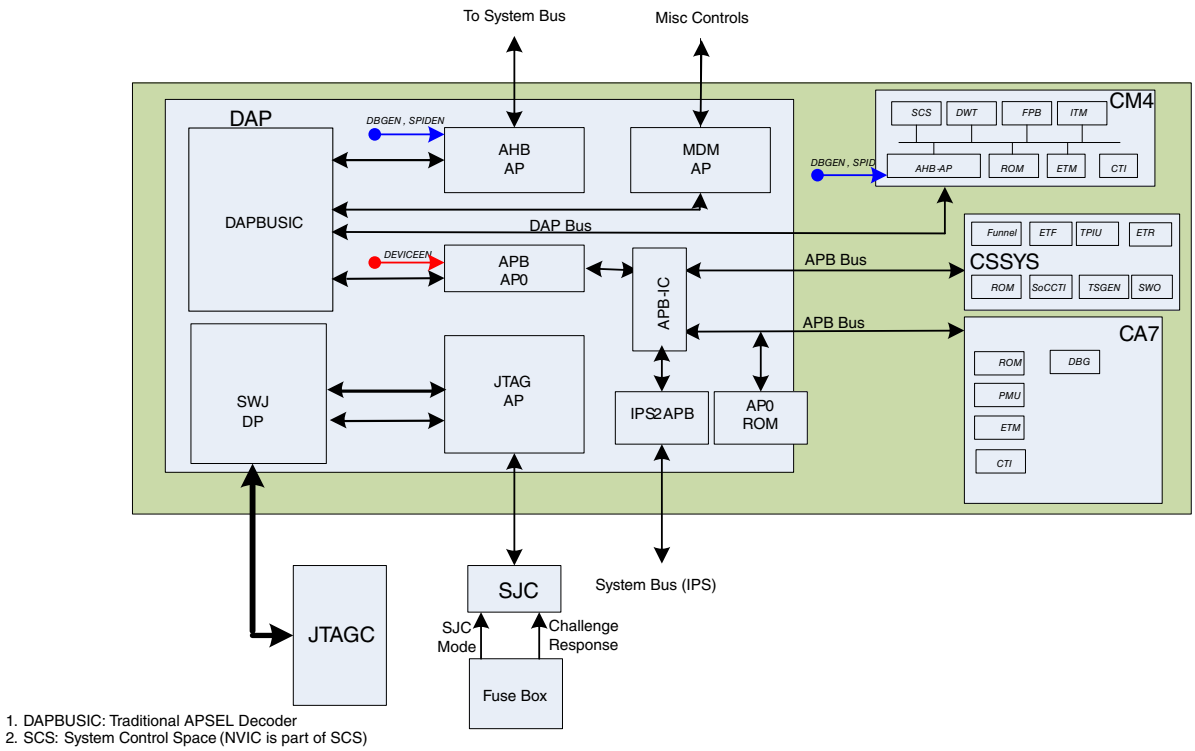


Figure 41-1. System level debug architecture

41.3 Test and debug port connectivity

The following figure shows the shared debug/test port connectivity, including the SWD/JTAG protocol switch block. The debugger can select one of them (JTAG or SWD) to access the DAP. The JTAG protocol is selected by default. JTAG-to-SWD change sequence describes how to change the protocol. The JTAGC can be only accessed by JTAG protocol.

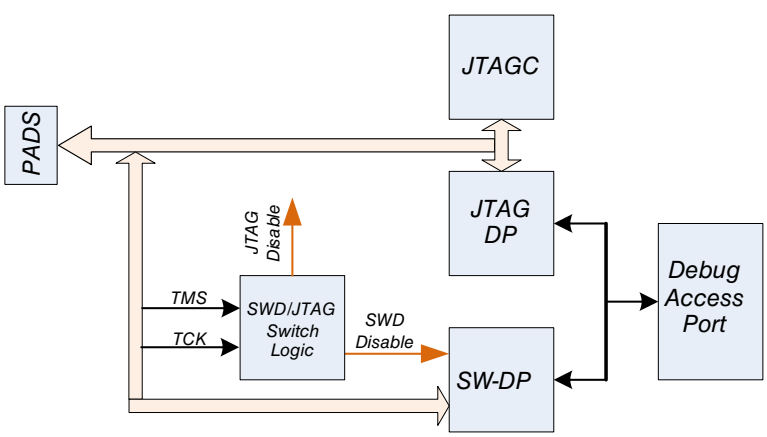


Figure 41-2. Test and debug port connectivity

The table below summarizes the setup of pins used in each protocol.

Table 41-1. JTAG pads summary

Pin Name	JTAG		SWD		Internal pull up/ down
	Type	Description	Type	Description	
TMS/SWDIO	I/O	mode selection	I/O	Data	Pull-up
TCLK/SWCLK	I	Clock	I	Clock	Pull-down
TDI	I	Data input	--	--	Pull-up
TDO/SWO[1]	O	Data output/Trace SWO	--	--	NC

1. The pin will turn in SWO mode, only at “Reduced pin count JTAG – IEEE 1149.7” or Arm SWD protocols.

The following table shows the debug's components ID code and your respective composition.

Table 41-2. ID code of debug components

Project	ID	Version Number	NXP's Use of JTAG ID Part Number			Manufacturer ID	IEEE req'd
			Design Center	Core Number	Chip Derivative Number	1110	1
		[31:28]	[27:22]	[21:17]	[16:12]	[11:1]	[0]
i.MX 7ULP	SJC: 0x188E101D	0001	100010	00111	00001	00000001110	1
	JTAG-DP: 0x6BA0_0477	0110	101110	10000	00000	01000111011	1
	SW-DP: 0x6BA0_2477	0110	101110	10000	00010	01000111011	1

41.3.1 JTAG to SWD switching sequence

41.3.1.1 JTAG-to-SWD change sequence

1. Send more than 50 TCK cycles with TMS(SWDIO)=1
2. Send the 16-bit sequence on TMS(SWDIO)=0111_1001_1110_0111(MSB transmitted first)
3. Send more than 50 TCK cycles with TMS(SWDIO)=1

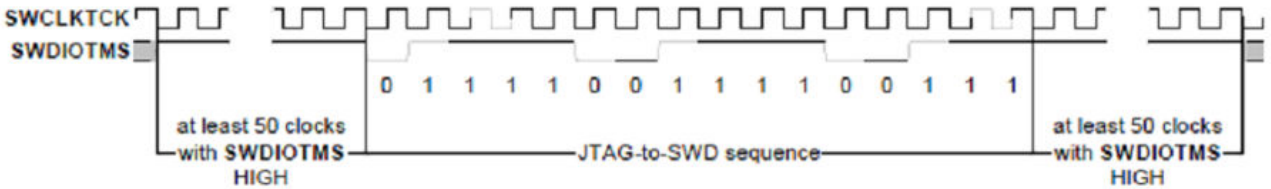


Figure 41-3. JTAG-to-SWD change sequence timing

41.3.2 System JTAG controller (JTAGC)

The system JTAG controller (JTAGC) is connected in parallel with Arm TAP controller (JTAG DP of Arm DAP). The IR length is 4-bits. The JTAGC IR codes overlay the Arm DAP controller IR codes. JTAGC uses 12 instructions and the DAP uses rest 4. The outputs of the TAPs (TDO) are muxed based on the selected IR Code. This design is fully JTAG compliant and appears to the JTAG chain as a single TAP.

The following table lists the IR codes for the system JTAG controller. For the instructions used by Arm DAP TAP, see Table 41-4 Arm DAP IR Codes.

Table 41-3. JTAG instructions for system JTAG controller (JTAGC)

Code	JTAGC IR
4'b0000	IDCODE
4'b0001	CENSOR_CTRL
4'b0010	SAMPLE/PRELOAD
4'b0011	SAMPLE
4'b0100	EXTEST
4'b0101	HIGHZ
4'b0110	Reserved
4'b0111	Reserved
4'b1000	*Not used by JTAGC (used by DAP)*
4'b1001	*AUX ACCESS 1 (TCU selection)*
4'b1010	*Not used by JTAGC (used by DAP)*
4'b1011	*Not used by JTAGC (used by DAP)*
4'b1100	Reserved
4'b1101	CLAMP
4'b1110	*Not used by JTAGC (used by DAP)*
4'b1111	BYPASS

The test functionality is implemented through MDM-AP registers and there is no dedicated TCU.

41.3.3 Debug Access Port (DAP) TAP

DAP is a standard Arm component. The DAP provides multiple master driving ports, all accessible and controlled through a single external interface port to provide system-wide debug. As described earlier, the DAP IR overlays with the system JTAG controller IR. The DAP Instruction Registers are listed in Table 41-4, Arm DAP IR Codes. DAP uses five Instructions but since BYPASS is identical with JTAGC it has been dropped out.

Table 41-4. Arm DAP IR Codes

Code	DAP IR
4'b1000	ABORT
4'b1010	DPACC
4'b1011	APACC
4'b1110	IDCODE

The DAP offers AHB and APB master interfaces to access system buses. It also exports the internal DAP bus to allow to extend the access ports as per the system requirement. It offers JTAG-AP to allow adding auxiliary TAPs. For more information on DAP TAP refer to Arm Debug Interface v5 Architecture specification on arm.com

For i.MX 7ULP, the AHB-AP provides the debugger access to all memory and registers in the system. System access is independent of the processor status. The AHB-AP's AHB transfers are burst size of 1 only with no out-of-order transactions and no multiple outstanding accesses. The APB-AP is used to access the CoreSight components for the CA7 core, and also the shared debug components of the system trace like TPIU and SWO.

The exported DAP bus is used to host AHB-AP (integrated as part of CM4 integration) and the Miscellaneous Debug Module Access Port (MDM-AP). The MDM-AP implement registers are used for Test mode control on i.MX 7ULP. The MDM-AP hosts system-level JTAG status and control registers (see Debug Status and Control registers) which can be used for cross triggering, synchronized debug, low power, and other miscellaneous control/status. The selection of different access ports in the DAP is done based on the APSEL value set in the SELECT register of (JTAG/SWD)-DP. APSEL is 31:24 bit field of the DAP SELECT register. The APSEL will be decoded as given in the following table, APSEL decode.

Table 41-5. APSEL decode

APSEL (SELECT[31:24])	Selection	Identification Register (IDR) value
8'h00 (M0)	AHB-AP to System Bus (NIC-301)	0x8477_0001

Table continues on the next page...

Table 41-5. APSEL decode (continued)

APSEL (SELECT[31:24])	Selection	Identification Register (IDR) value
8'h01 (M1)	APB-AP0 (CSSYS)	0x5477_0002
8'h02 (M2)	JTAG-AP to SJC	0x3476_0010
8'h03 (M3)	AHB-AP to CM4	0x2477_0011
8'h04 (M4)	Miscellaneous Debug module AP (MDM-AP)	0x001C_0040
8'h06 - 8'hff	Reserved (Default AP response)	-

41.4 ROM tables

The ROM tables hold information about different debug components and help identify them. There are three ROM tables:

1. CM4 ROM table: This table resides in the CM4 AHB-AP and contains entries for CM4 debug components. See Table 41-7.
2. ROM tables containing entries corresponding to the APB-APs. See Table 41-8.
3. CA7 ROM tables: These entries identify the common debug components and the CA7 core debug components. See Table 41-9.

Table 41-6. ROM Table Entry format

Bits	Name	Description
31:12	Address offset	Base address of the component, relative to the ROM address. Negative values are permitted using two's complement. ComponentAddress = ROMAddress + (AddressOffset SHL 12).
11:2	-	Reserved
1	Format	1 = 32-bit format. In the DAP Debug ROM, this is set to 1. 0 = 8-bit format.
0	Entry Present	Set HIGH to indicate that an entry is present.

Table 41-7. CM4 ROM table

Address (For Debugger)	i.MX 7ULP Memory Map	Content	Component
0xE00FF000	0xE000E000	0xFFF0F003	SCS
0xE00FF004	0xE0001000	0xFFF02003	DWT
0xE00FF008	0xE0002000	0xFFF03003	FPB
0xE00FF00C	0xE0000000	0xFFF01003	ITM
0xE00FF010	--	0xFFF41002	No component present
0xE00FF014	0xE0041000	0xFFF42003	ETM

Table continues on the next page...

Table 41-7. CM4 ROM table (continued)

Address (For Debugger)	i.MX 7ULP Memory Map	Content	Component
0xE00FF018	--	0xFFF43002	No component present
0xE00FF01C	--	0xFFF44002	No component present
0xE00FF020	0xE0044000	0xFFF45003	CTI
0xE00FF024	--	0x00000000	End marker

Table 41-8. APB AP0 ROM Table

Address	i.MX 7ULP Memory Map	Content	Component
0x80000000	0x41092000	0x00012003	FUNNEL
0x80000004	0x41093000	0x00013003	ETF
0x80000008	0x41094000	0x00014003	TPIU
0x8000000C	0x41095000	0x00015003	ETR
0x80000010	0x41096000	0x00016003	SOCCTI
0x80000014	0x41097000	0x00017003	SWO
0x80000018	0x41098000	0x00018003	TSGEN
0x8000001C	0x4109A000	0x00020003	CA7ROM
0x80000020	--	0x00000000	End marker

Table 41-9. CA7 ROM Table

Address	i.MX 7ULP Memory Map	Content	Component
0x80020000	0x4109B000	0x00010003	DBG
0x80020004	0x4109C000	0x00011003	PMU
0x80020008	--	0x00012002	No component present
0x8002000C	--	0x00013002	No component present
0x80020010	--	0x00014002	No component present
0x80020014	--	0x00015002	No component present
0x80020018	--	0x00016002	No component present
0x8002001C	--	0x00017002	No component present
0x80020020	0x4109D000	0x00018003	CTI
0x80020024	--	0x00019002	No component present
0x80020028	--	0x0001A002	No component present
0x8002002C	--	0x0001B002	No component present
0x80020030	0x4109E000	0x0001C003	ETM
0x80020034	--	0x0001D002	No component present
0x80020038	--	0x0001E002	No component present
0x8002003C	--	0x0001F002	No component present
0x80020040	--	0x00000000	End marker

41.5 Trace architecture/topology

The following figure depicts the debug trace architecture of this device. Components shown in Brown are on A7 clock domain. Everything else is on CM4 clock domain. The 64 bit AHB port is a trace master port on CM4 platform.

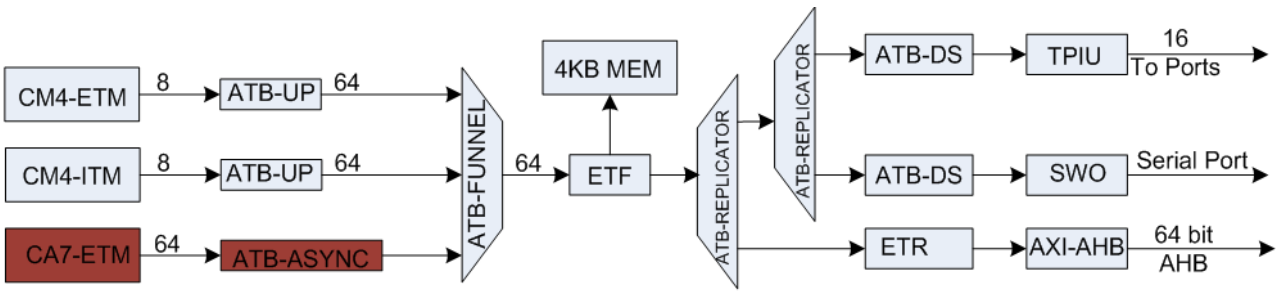


Figure 41-4. Trace architecture/topology

41.6 Debug trace timestamping

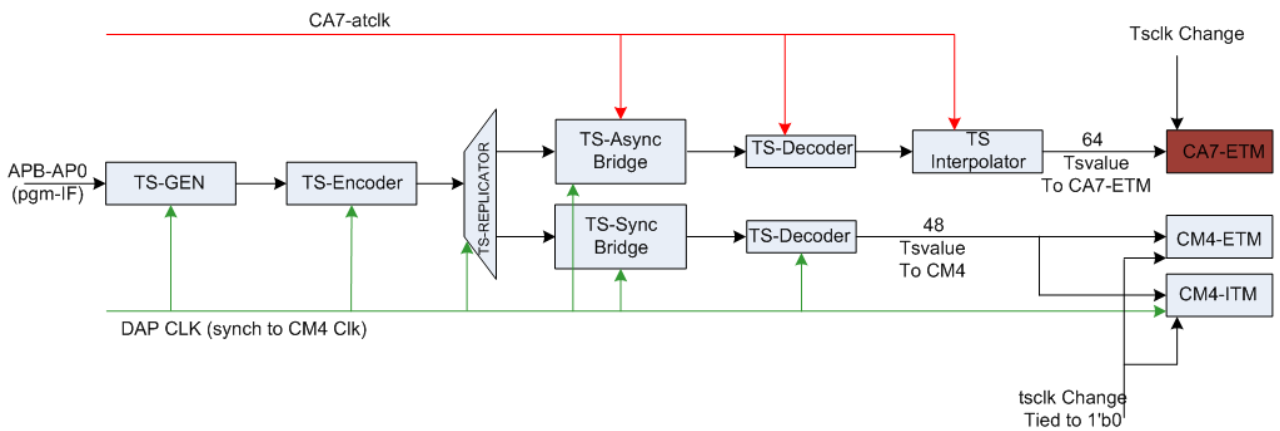


Figure 41-5. Debug trace timestamping

Timestamp solution establishes correlation between the trace data across the two cores. The above figure represents the timestamp topology for the timestamp values observed along with the traces.

- The timestamp generator is a 64-bit counter, working in the CM4 clock domain.
- The TS-Async Bridge separates the two clock domains. The TS-Sync Bridge acts as a register slice to achieve timing closure.
- The CA7 atclk must be faster than cm4clk to assure the correctness of timestamp counter value, otherwise the FIFO inside TS-Async Bridge would overrun.
- Ts value[63:48] are unconnected when connecting to CM4 ETM and ITM

- Since the frequency ratio of the timestamp generator clock to CM4-ETM/ITM clock is always 1:1, tsclk_change input of CM4-ETM and CM4-ITM needs to be tied to 0.
- The SoC clock controller should inform the CA7-ETM through its input clk_change when CA7-ETM's clock is changed.
- The TS-GEN has control register to enable/disable the timestamp counter.

NOTE

The CLKCHANGE (Tsclk Change) signal of CA7-ETM is used when either the processor clock period or timestamp period changes, as mentioned in the Arm documentation: DDI0468A_coresight_etm_a7_trm.

See the following Arm document for timestamp descriptions:

- Chapter 7 of DDI0480G_coresight_soc_r3p2_trm
- DDI0468A_coresight_etm_a7_trm

41.7 Debug Status and Control registers

The Miscellaneous Debug Module (MDM-AP) provides useful interface between debugger and the core and controls debug information in multi-core environment. Besides the core control, the interface also implements registers controlling the Test Registers . The debugger can write to MDM-AP registers only if the DBGEN is set, that is, the debugger security authentication has been cleared. In case the debugger makes an access while DBGEN=0, the access will be simply ignored and no error will be reported. The MDM-AP status registers are always available as long as the JTAG port is available.

NOTE

The DAP Control and Status registers are not memory mapped within the system memory map and are only accessible via the Debug Access Port using JTAG or SWD. The MDM-AP is accessible as Debug access port 04 as shown in Table 41-5, APSEL Decode.

Table 41-10. DAP Status and Control registers

Address	Register
0x00	Status register
0x04	Control register
0x10	Core halt req reg
0x20	Core restart req reg
0x30	Core halt status reg
0x40	Core low power status1 reg

Table continues on the next page...

Table 41-10. DAP Status and Control registers (continued)

Address	Register
0xFC	IDR: Fixed ID register (0x001C0040)

41.8 Register details

The following table presents the descriptions of bit fields contained in Debug registers. All the bits which are not mentioned in the register tables below are reserved.

Table 41-11. Register details

Bit Number	Field	Description
Status (dap_status)		
1	CM4 System in reset	Indicates if CM4 system is in reset 0: System is not in reset 1: System is in reset **NOTE:** CM4 in reset implicitly means CA7 system is in reset too.
2	CA7 System in reset	Indicates if the CA7 system is in reset 0: System is not in reset 1: System is in reset **NOTE:** CA7 can continue to remain in reset while CM4 system is up and running.
3	DBGEN / NIDEN	Debug Access enable: Indicates if the debugger can access to AHB-AP and APB-AP. NIDEN is same as DBGEN in MX7. DBGEN: Invasive Debug Enable NIDEN: Non-invasive Debug Enable
4	SPIDEN / SPNIDEN	Debug secured access enable. Indicates if the debugger can make secured accesses to the debug logic and can trace out secured traces. This bit reflects the status of SPIDEN/ SPINDEN
10	STANDBY Mode Exit	This bit indicates that an exit from Standby (VLLS) mode has occurred. The debugger has lost connection while the system was in Standby mode. Once this connection is re-established, this bit indicates that the system is back and ready to exit Standby. Debugger should use this bit to complete low power exit protocol. This bit is held until the debugger had a chance to recognize that a standby (VLLS) mode is exited and is cleared by writing 1 to the STANDBYDBGACK (bit [6], dap_ctrl).
22	CTI trigger out	CTI interface trigger out
30	CM4 DBGRESTARTED	Indicates that CM4 leaves Debug mode
31	CA7 DBGRESTARTED	Indicates that CA7 leaves Debug mode

Table continues on the next page...

Table 41-11. Register details (continued)

Bit Number	Field	Description
Control(dap_ctrl)		
2	CM4 System Reset Request	Set to force the CM4 system reset. The system remains in reset till this bit remains asserted. When this bit is deasserted, the entire system comes out of reset.
3	CA7 System Reset Request	Set to force the CA7 system reset. The CA7 system remains in reset till this bit remains asserted. When this bit is reset, the CA7 system comes out of reset.
5	STANDBYDBGREQ	This bit is set to configure the system to be held in reset after the next recovery from Low power mode. This bit is sampled by the Reset Control Module (RCM)[1] just before the part moves into low power states and remains latched in always ON domain during low power phase. On exit from low power, the core is held under reset until STANDBYDBGACK is asserted (The STANDBYDBGREQ bit is cleared due to reset generated as a part of low power recovery).
6	STANDBYDBGACK	This bit is set to release the cores being held in reset following a standby recovery. This bit indicates that debugger has finished the configuration of debug components. This bit directly drives the RCM control. This bit is also used to clear “STANDBY Mode Exit” bit. A POR or JTAG can clear this bit.
7	Inhibit Debug in Deepsleep	When this bit is set, the STOP mode entry will ignore debug connection.
10	CM4 Hold Reset	When set, this bit leaves the CM4 core under reset during power-up. Rest of the system will come out of reset as usual.
11	CA7 Hold Reset	When set, this bit leaves the CA7 core under reset during power-up; rest of the system will come out of reset as usual.
12	CM4 Core Reset Request	Set to cause a CM4 core reset.
13	CA7 Core Reset Request	Set to cause a CA7 core reset.
24	CTI Trigger Input	Connects to CTI trigger input
25	CTI Trigger Out ACK	Connects to CTI trigger ACK input
Core halt request (dap_ctrl_halt_req)		
0	CM4 EDBGRQ	The external debug request debug event which causes CM4 to enter Debug state.
2	CA7 EDBGRQ	The external debug request debug event which causes CA7 to enter Debug state.
Core restart request(dap_ctrl_restart_req)		
0	CM4 DBGRESTART	CM4 debug restart input to CM4 core causing CM4 to leave debug state
2	CA7 DBGRESTART[0]	CA7 debug restart input to CA7 core
Core halt (dap_status_halt)		
0	CM4 HALTED	Indicates CM4 has entered debug halted mode
2	CA7 DBGACK[0]	Indicates CA7 CPU#0 has entered debug halted mode
Core low power status (dap_status_lp1)		
0	CM4 SLEEPDEEP	Reflects the status of Cortex M4. Driven by CM4

Table continues on the next page...

Table 41-11. Register details (continued)

Bit Number	Field	Description
1	CM4 SLEEPING	Reflects the status of Cortex M4. Driven by CM4
8	CA7 STANDBYWFI[0]	Indicates that CA7 CPU#0 is in WFI standby mode.
9	CA7 STANDBYWFE[0]	Indicates that CA7 CPU#0 is in WFE standby mode.
10	CA7 STANDBYWFIL2	Indicates if the L2 memory system is in WFI standby mode
Identification Register		
0-3	Type	4'h0. (MDM-AP)
4-7	Variant	4'h4
8-15	Reserved	Read as 8'h00.
16	Class	1'b0. Indicates not a memory access
17-27	JEP 106 Identity + Continuation	11'b00000001110(NXP)
28-31	Revision	4'h0

1. MSMC, CMC, and RMC are 3 separate but tightly coupled blocks. The memory map documented in MSMC chapter encompasses features related to all 3 blocks, being actually implemented into the CMC block alone. Therefore, there might be some references to CMC and RMC blocks throughout this document.

NOTE

STANDBYDBGREQ and ack are not implemented for CA7 system as the MDM-AP is in the M4 domain and A7 can be controlled using the existing signals. If M4 happens to be powered down as well, it is implicit that M4 will always power up first and the debug controls for the A7 core can be restored during this period.

41.9 Debug resets

The debug system receives the following sources of reset:

- Debug Reset (CDBGRSTREQ bit within the SWJ-DP CTRL/STAT register; See Arm Debug Interface Architecture Specification) which is in the TCLK domain that allows the debugger to reset the debug logic.
- System POR request. The debug system gets reset on POR and the JTAG interface on TCK clock domain is available after POR subject to security clearance.

The debug system is capable of generating system reset using the MDM-AP control registers. A functional reset leads to loss of debugger connection and debugger needs to re-initiate the connection as soon as the system is out of reset.

41.10 Embedded cross triggers

Embedded Cross Trigger (ECT) is used for multicore run control and allow subsystems to interact and trigger each other. It allows start/stop for all cores or trigger program trace on a trigger event from multiple trigger sources (other modules). ECT is made up of Cross Trigger Interfaces (CTI) and Cross Trigger Matrix (CTM).

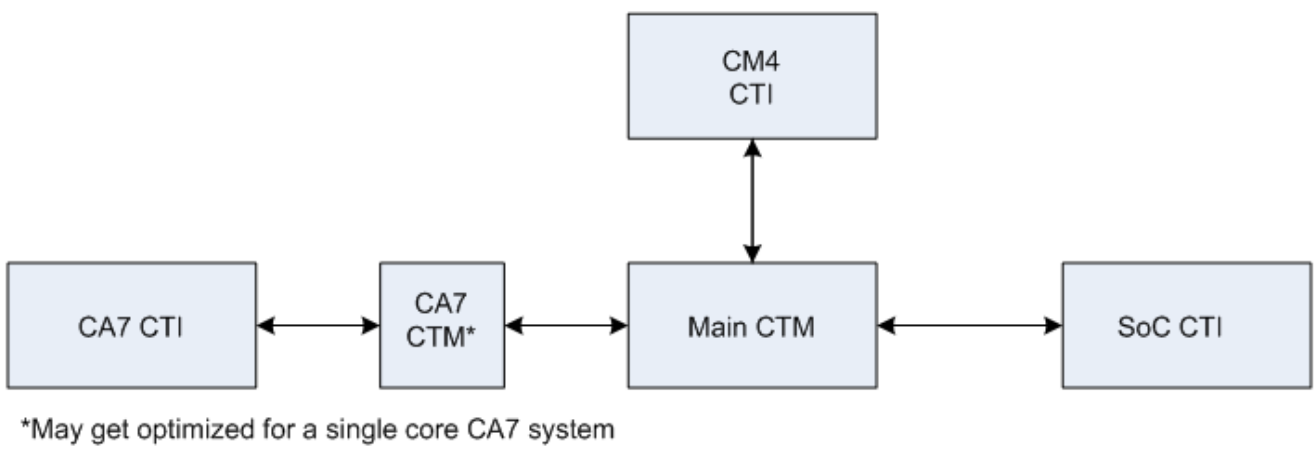
Cross Trigger Interface (CTI) is an Arm IP and provides:

- 8 trigger inputs, 8 trigger outputs, basic signal conditioning
- Programmable channel routing (mapping trigger inputs to channels and channels to trigger outputs)
- Limited to 4 channels

Cross Trigger Matrix (CTM) is an Arm IP and provides:

- Channel routing when more than 2 CTIs are used
- Limited to 4 channels

i.MX 7ULP implements 3 CTIs. Each core has a dedicated CTI and one is available for various triggers at the chip level.



NOTE

It is not possible to trigger between CM4 CTI and SoC CTI when CA7 power domain is turned OFF.

41.10.1 CM4 CTI triggers

Table 41-12. CM4 CTI input triggers

Trigger Bit	Source Signal	Source Module	Description
7	ETMTRIGOUT	CM4 ETM	CM4 ETM trigger
6	ETMTRIGGER[2]	CM4 DWT	CM4 DWT trigger
5	ETMTRIGGER[1]	CM4 DWT	CM4 DWT trigger
4	ETMTRIGGER[0]	CM4 DWT	CM4 DWT trigger
3	--	--	--
2	--	--	--
1	--	--	--
0	HALTED	CM4 core	CM4 core's output indicating that it has entered the debug mode.

Here the "Source Signal" is the output port of the "Source Module". They are connected to the input triggers of the CTI

Table 41-13. CM4 CTI output triggers

Trigger Bit	Destination Signal	Destination Module	Description
7	DBGRESTART	CM4 core	Request CM4 core to quit the debug mode
6	--	--	--
5	EXTIN[1]	CM4 ETM	CM4 ETM external input
4	EXTIN[0]	CM4 ETM	CM4 ETM external input
3	cti_irq	NVIC	CTI interrupt
2	cti_irq	NVIC	CTI interrupt
1	--	--	--
0	EDBGRQ	CM4 core	External request CM4 core to enter the debug mode

Here the "Destination Signal" is the input port of the "Destination Module". The output triggers of the CTI are connected to the input port of the destination modules.

41.10.2 CA7 CTI triggers

Table 41-14. CA7 CTI input triggers

Trigger Bit	Source Signal	Source Module	Description
7	--	--	--
6	ETMTRIGGER	CA7 ETM	CA7 ETM trigger

Table continues on the next page...

Table 41-14. CA7 CTI input triggers (continued)

Trigger Bit	Source Signal	Source Module	Description
5	COMMRX	CA7 Debug	Debug communication receive channel is full
4	COMMTX	CA7 Debug	Debug communication transmit channel is empty
3	EXTOUT[1]	CA7 ETM	CA7 ETM external output
2	EXTOUT[0]	CA7 ETM	CA7 ETM external output
1	nPMUIRQ	CA7 PMU	CA7 PMU generated interrupt
0	DBGTRIGGER	CA7 Debug	Trigger entry to debug state

Table 41-15. CA7 CTI output trigger

Trigger Bit	Destination Signal	Destination Module	Description
7	DBGRESTART	CA7 Debug	Cause the CA7 processor to exit debug state
6	nCTIIRQ	GIC	CTI interrupt
5	CTIEXTTRIG		CA7 CTI external trigger
4	EXTIN[3]	CA7 ETM	CA7 ETM external input
3	EXTIN[2]	CA7 ETM	CA7 ETM external input
2	EXTIN[1]	CA7 ETM	CA7 ETM external input
1	EXTIN[0]	CA7 ETM	CA7 ETM external input
0	DBGRQ	CA7 Debug	Cause the CA7 processor to enter debug state

41.11 Low power debug

It is required that the system provides control so that debugger can have control of the device when entering and exiting low power modes. During low power modes, the clock and power (based on modes) will not be available to multiple components, the following handshake allows the debugger to maintain connection during low power. There are two modes supported:

- Clock gated mode
- Power gated mode

41.11.1 Clock gated modes

- If the DAP CxxxPWRUPREQ is high and the system attempts to enter in STOP mode, the core will enter the SLEEP mode but the mode of the system will not change, except if the MDM-AP dap_ctrl bit [7], Inhibit Debug in Deepsleep, is set. In this case, the system will ignore the debugger connection and will enter the STOP mode.
- If the device is in STOP mode (clock gated), the debugger can assert EDBGRQ (CM4 EDBGRQ, CA7 EDBGRQ) in dap_ctrl_halt_req register. This will wake up the core and will move it into a halted state.

41.11.2 Power gated modes

This mode is applicable for CM4 domain only. During Power gated modes, the debugger will lose connection and it cannot gather any data in the duration when the device stays in low-power mode. If the device is expected to go into low-power mode and it is desired to gain control of device when it exits low power modes, the following sequence/handshake should be followed.

- Low power entry
  a. Set the DAP CxxxPWRUPREQ (inhibit_sleep) to indicate to the device that low power handshake should take place, before entering power gated modes
  b. Low-power entry requested by CM4.
  c. Set the MDM-AP dap_ctrl bit [5], STANDBYDBGREQ, which tells the device to remain in reset until the handshake takes place on standby(VLLS) exit. Note that this bit needs to have a local version inside DGO (RCM) which will keep this information during low power exit. When the reset sequence completes (after debug handshake), the DGO version of the bit should be cleared.
  d. As the device is going to get into low power mode, debugger should store all contexts
  e. Reset the DAP CxxxPWRUPREQ, without modifying any other bit, the device will enter Standby (VLLS) mode.
- Low power exit
  a. A wakeup event wakes up the device
  b. Debugger tries to re-establish connection. Once the connection has been established, it will know if the device is ready to come out of low power mode, by reading the status of bit 10 “STANDBY Mode Exit” in dap_status. If this bit is set, the device is ready for low power exit.
  c. Set the MDM-AP, dap_ctrl_halt_req bit [0], CM4 EDBGRQ, to indicate that the CM4 should enter the debug mode out of reset.
  d. To complete the exit protocol write to STANDBYDBGACK, bit[6] of MDM-AP dap_ctrl.

e. Debugger should check MDM-AP, dap_status bit[1], CM4 System in reset, to check if CM4 is out of reset.
f. Debugger should monitor MDM-AP, dap_status_halt bit[0], CM4 HALTED, indicating that CM4 has entered into debug halt mode.
g. Debugger can now restore contents of CM4 debug peripherals
h. Restart the CM4 from debug mode by setting MDM-AP, dap_ctrl_restart_req bit [0], CM4 DBGRESTART
i. Monitor dap_status, bit[30], CM4 DBGRESTARTED to ensure CM4 exited the debug mode.

41.12 Instruction trace

The A7 instruction trace has the same functional capabilities as the M4 instruction trace.

41.13 Secured JTAG

The SJC block offers a range of security levels. The SJC blocks the access to debug based on eFuse configuration and authentication. This is summarized in the table below:

Name	Security level	Field return	Comments	Fuse used	Debug signals	Note
No JTAG	Highest	The field return is supported using the Fuse bit FIELD_RETURN[1]	All JTAG features are disabled.	SJC_DISABLE = 1	DBGEN = 0 NIDEN = 0 SPIDEN = 0 SPNIDEN = 0	TCK had been gated
No debug	High		All JTAG features are disabled except for BS	SJC_DISABLE = 0 JTAG_SMODE = 11	DBGEN = 0 NIDEN = 0 SPIDEN = 0 SPNIDEN = 0	
Secure JTAG	Medium		Authentication based on challenge-response	SJC_DISABLE = 0 JTAG_SMODE = 01	DBGEN = 1 NIDEN = 1 SPIDEN/ SPNIDEN[2]= SIM_DGO_GP11[0] (register)	SPIDEN/ SPNIDEN based on challenge-response.
JTAG enabled	No secure	Full supported possible for field return	Open	SJC_DISABLE = 0 JTAG_SMODE = 00	DBGEN = 1 NIDEN = 1	

				SEC_CONFIG[0] = 1	SPIDEN = 1 SPNIDEN = 1	

1. For details on FIELD_RETURN and other eFuse bits, see the i.MX 7ULP Security Reference Manual.
2. These bits use only one SIM register: SIM_DGO_GP11[0] register, which is:
   - Under supervisor access control
   - Can be retained through Low power (DGO)
   - The reset value of this register should be 0
   - These are used by device running in secure mode

NOTE

The field return is supported using Fuse bit "FIELD_RETURN". This fuse is blown based on an authenticated image. The part which is received as field return is not returned to the customer. The field return part offers complete access to TCU and debug

i.MX 7ULP comes out of POR in "No JTAG" security level for security reasons. Security level is set after fuses are read during the reset phase coming out of POR.

41.14 Additional Authentication Interface

Authentication interface aims to restrict access to debug and trace functionality. This is done by controlling the Arm debug authentication signals. For more details, see the CoreSight v1.0 Architecture Specification. These are listed below:

Table 41-16. Arm debug authentication signals

Control Signal name	Meaning	Controlled by
DBGEN	Invasive Debug Enable	SJC based on security authentication
NIDEN	Non invasive Debug Enable	SJC based on security authentication
SPIDEN	Secure invasive debug enable	control bit in SIM_DGO_GP11[0] register (can be SET by only s/w running in secure world)
SPNIDEN	Secure non-invasive debug enable	control bit In SIM_DGO_GP11[0] register (can be SET by only s/w running in secure world)

The SJC controls the DBGEN and NIDEN based on the security authentication and perceived security incident related to power up. This allows debugger access to non-secure coresight components and allows non-secure traces to be traced out. In case debug is required in the secure world, SPIDEN and SPNIDEN need to be set. This is feasible only by the software running in the secure world.

NOTE

For details about security features, see the i.MX 7ULP Security Reference Manual.

Chapter 42
Secured JTAG Controller (SJC)

42.1 Chip-specific SJC information

Table 42-1. Reference links to related information

Topic	Related module	Reference
Full description	SJC	SJC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

42.1.1 Secured JTAG Controller (SJC)

SJC is an authenticated debug module that implements a challenge/response mechanism using a standard cryptographic algorithm. This allows post production silicon debug without compromising security requirements. The SJC is connected as a tap of DAP JTAG-AP and is used for authenticated debug.

Table 42-2. SJC configuration

Parameter	Description
Name	SJC
Supported standard version	IEEE 1149.1-2001
Instances	1
Configurable features	NA
Interface speed	NA
External I/O pins	-

42.1.2 No Debug mode of operation

On i.MX 7ULP, MBIST and LBIST modes are not allowed at No Debug mode.

NOTE

In No Debug mode, the access to DAP (DP and APs) will be enabled but to rest of the system, blocked. The access to registers of MDM-AP will be read-only.

42.1.3 Challenge/Response mechanism

In fixed challenge-response pair mode, each chip has its individual challenge/response pair which is determined at manufacturing time. In this device, the SJC compares the user's response to the expected response. The update of the authentication result will be delayed two cycles of TCK clock after the Update-DR state.

42.2 Introduction

Secure JTAG Controller (SJC) provides the security authentication for debug access to the chip. It is accessible through the JTAG Access Port (AP) of the ARM DAP.

This figure shows SJC connections to external contacts and other blocks.

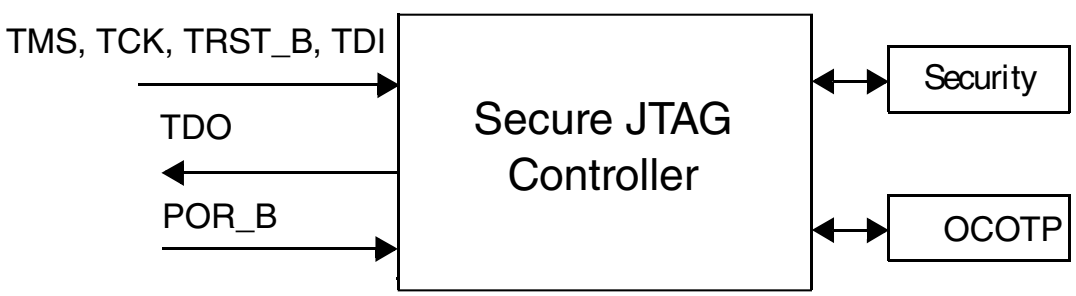


Figure 42-1. SJC connections

42.3 External signal description

42.3.1 External signal overview

SJC provides test and debug control with a minimum number of contacts.

This table describes the SJC external connections along with the signal properties.

Table 42-3. SJC signal properties

Name/Port	Function
POR_B	POR reset input.
TCK	Test Clock (TCK). This is used to synchronize the test logic.
TDI	Test Data Input (TDI). Serial test instruction and data are received through the test data input (TDI) pin. TDI is sampled on the rising edge of TCK.
TDO	Test Data Output (TDO). The serial output for test instructions and data. TDO is tri-statable and is actively driven in the shift-IR and shift-DR controller states. TDO changes on the falling edge of TCK.
TMS	Test Mode Select (TMS). This is used to sequence the test controller's state machine. TMS is sampled on the rising edge of TCK.

42.3.2 TAP controller

The TAP controller is responsible for interpreting the sequence of logical values on the TMS signal. It is a synchronous state machine that controls the operation of the JTAG logic. For a description of the TAP controller states, refer to the appropriate IEEE 1149.1 document.

The state machine is shown in this figure. The value shown adjacent to each arc represents the value of the TMS signal sampled on the rising edge of TCK signal.

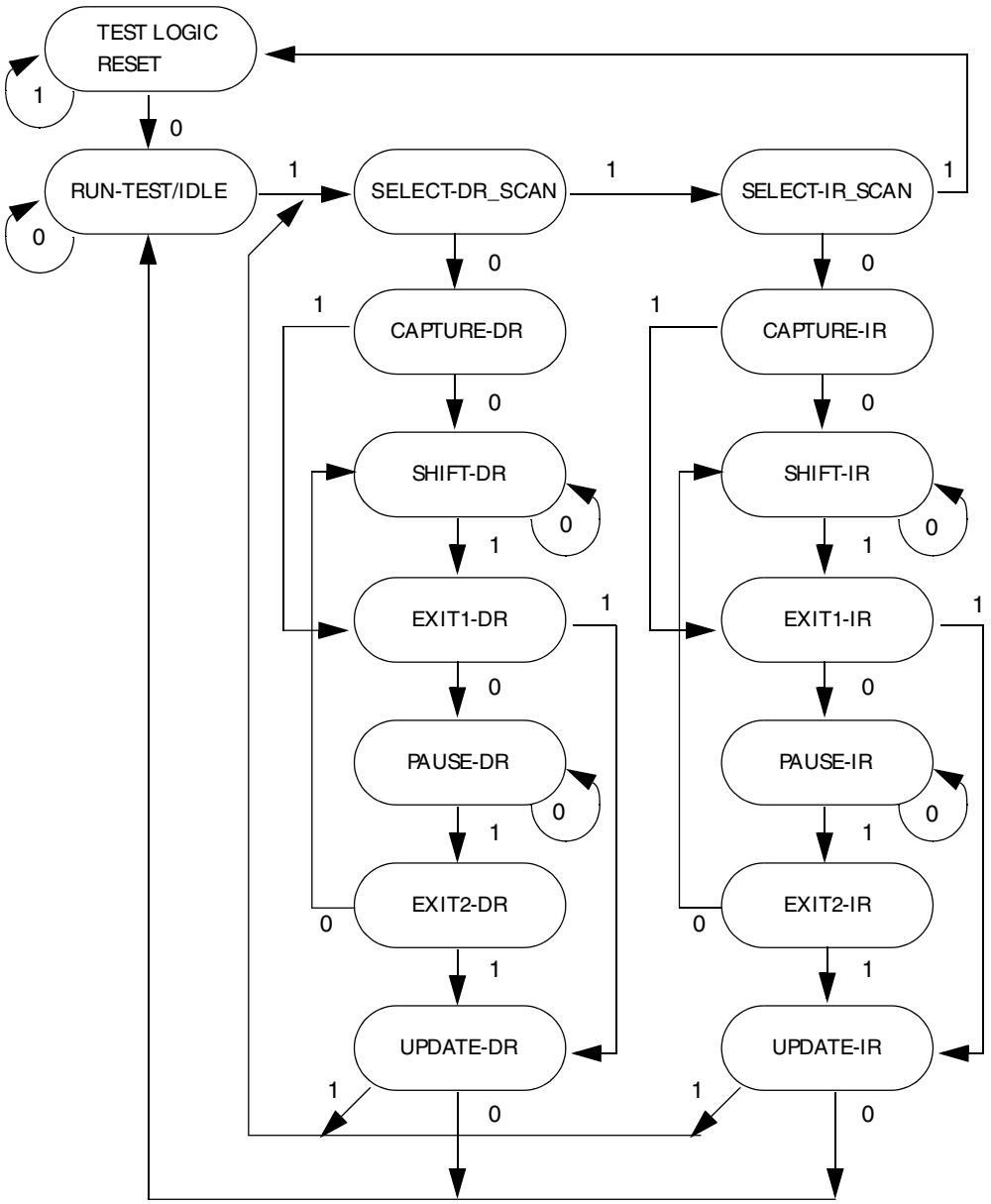


Figure 42-2. TAP controller state machine

State changes occurs on the rising edge of TCK. TMS and TDI change on the falling edge of TCK. TDO also changes on the falling edge of TCK following entry into the Shift_DR or Shift_IR states.

This figure shows the SJC signal timings.

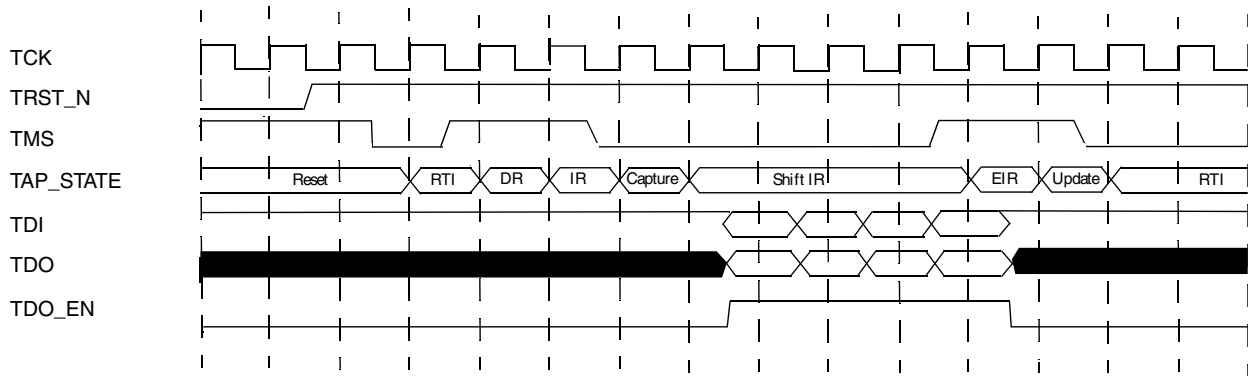


Figure 42-3. SJC signals timing diagram

42.3.3 Accessing ExtraDebug register

Accessed through the Select-DR-Scan path, the ExtraDebug shift register consists of 38 bits (maximum) including a 32-bit data field (max length, see the ExtraDebug register description), a 5 bit address field and read/write bit.

The write occurs when the JTAG TAP controller enters the Update-DR state. On a read, the data field is ignored (the user should shift only 5 times to enter Read=1 and the address), the read takes place on the next path through DR at the Capture-DR state, and the data is shifted-out during the Shift-DR state.

On the second path for a read access, simultaneous write access is not supported. Command converter software shifts in zeros so the TAP decodes a write to the CSR (read-only register) which does not have any effect.

The number of shift depends on the width of the accessed register as explained in the following diagrams. First a write access (one path through Select-DR-Scan):

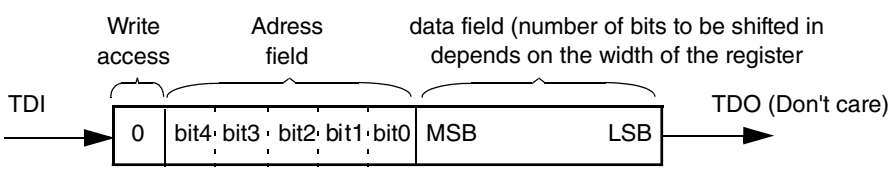


Figure 42-4. TDI/TDO on write access

Then a read access (requires two paths through JTAG DR Scan path):

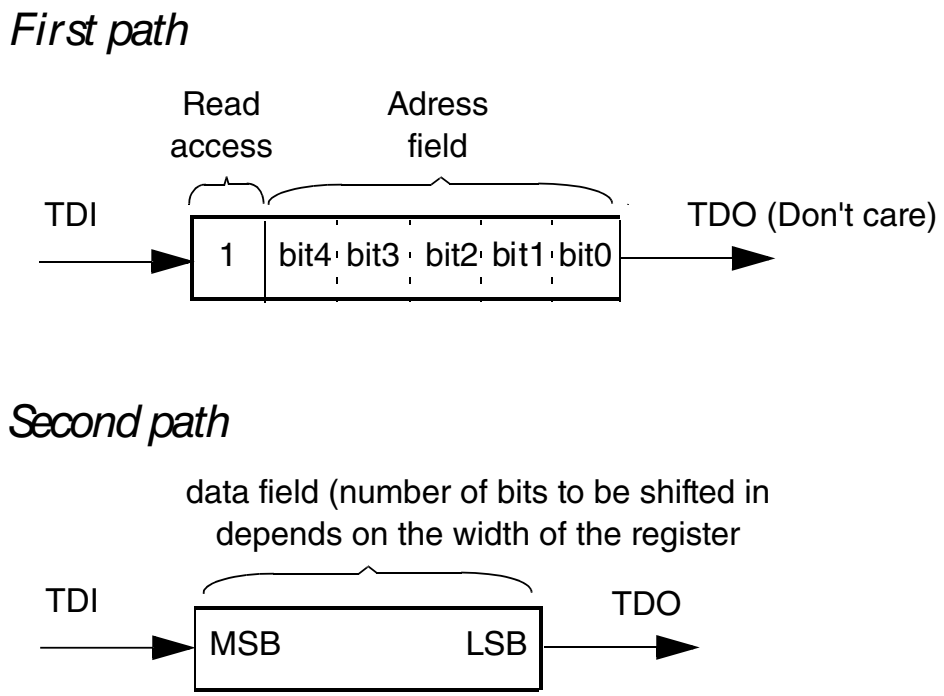


Figure 42-5. TDI/TDO on read access

For example, write value 1010_1100b to Debug Control Register (address = 00110b).

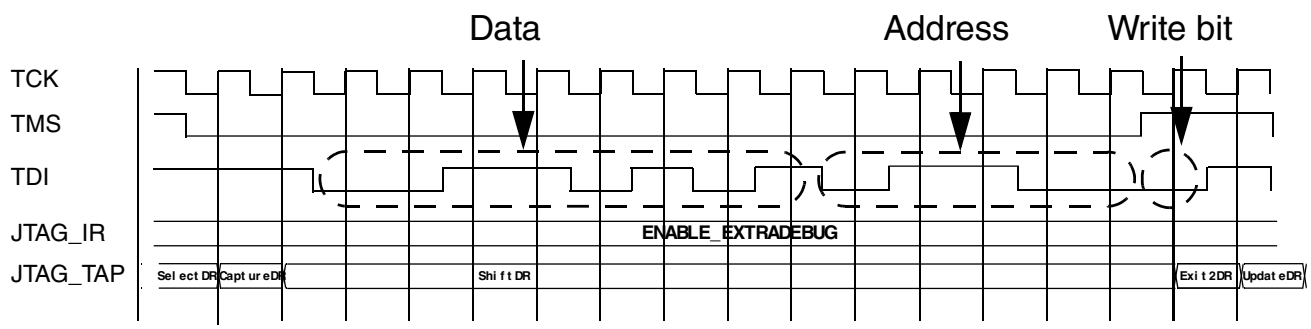


Figure 42-6. Example: Write access to DCR

The SJC registers have different levels of security (refer to JTAG Security Modes):

- Secured—accessible only in Mode 2 (correct response entered), Mode 3 and Mode 4.
- Unsecured—accessible in all modes

42.4 JTAG Instruction Register (SJIR)

The JTAG Instruction register is 5 bits wide.

Code					SJC IR
B4	B3	B2	B1	B0	
0	0	0	0	0	Reserved
0	0	0	0	1	Reserved
0	0	0	1	0	Reserved
0	0	0	1	1	Reserved
0	0	1	0	0	Enable Extra misc registers
0	0	1	0	1	Reserved
0	0	1	1	0	Reserved
0	0	1	1	1	Reserved
0	1	0	0	0	Reserved
0	1	0	0	1	Reserved
0	1	0	1	0	Reserved
0	1	0	1	1	Reserved
0	1	1	0	0	Security Output Challenge
0	1	1	0	1	Security input response
01110–01111					Reserved
1	1	1	1	1	BYPASS

The instruction register is reset to 00000b in the test-logic-reset controller state which is equivalent to the IDCODE instruction.

During the capture-IR controller state, the parallel inputs to the instruction register are loaded with the code 01b in the least significant bits as required by the standard; the most significant bits are loaded with the values 00b, leading to a capture value of 00001b.

42.4.1 BYPASS instruction

This instruction selects the single bit bypass register and the system logic controls the I/O pins. This creates a shift-register path from TDI to the bypass register and, finally, to TDO, circumventing the boundary scan register.

When the bypass register is selected, the shift-register stage is set to a logic zero on the rising edge of TCK in the capture-DR controller state. Therefore, the first bit to be shifted out after selecting the bypass register is always a logic zero.

For more details on the function and use of BYPASS, refer to the appropriate IEEE 1149.1 document.

42.4.2 ENABLE_ExtraDebug instruction

This instruction connects the TDI and TDO pins directly to the ExtraDebug registers, the SJC TAP controller remaining connected to TDI and TMS.

The ExtraDebug shift register consists of 38 bits (maximum) including a 32-bits data field (maximum length, see Accessing ExtraDebug register), a 5 bits address field and read/write bit. On a register read, the data field does not need to be filled in. The particular ExtraDebug register connected between TDI and TDO at a given time is selected by the ExtraDebug controller depending on the ExtraDebug address being currently decoded. All communication with the ExtraDebug controller is done through the Select-DR-Scan path of the JTAG TAP Controller.

42.5 Security

The SJC provideds means to block any malicious JTAG access. The JTAG security modes are as follows:

- Mode 1: No Debug—maximum security. All security sensitive JTAG features are permanently blocked.
- Mode 2: Secure JTAG—high security. JTAG use is regulated by secret key based authentication mechanism.
- Mode 3: JTAG Enabled—low security. JTAG always enabled.

The JTAG security modes are configured using fuses, which can burned after packaging by applying electrical signals. Fuse burning is irreversible process. Once a fuse is burned it is not possible to change the fuse back to the un-burned state.

42.5.1 JTAG Security Modes

42.5.1.1 Mode 1: no debug—maximum security

No debug security mode provides the highest security level.

In this mode:

- All security sensitive JTAG features are permanently blocked
- Boundary scan still allowed

These features do not reduce the security level, and they allow important tests and board connectivity checks.

42.5.1.2 Mode 2: secure JTAG–high security

The secure JTAG mode limits the JTAG access by using challenge/response based authentication mechanism. Any access to JTAG port is checked. Only authorized debug devices (that is, devices having the right response) can access the JTAG. Unauthorized JTAG access attempts are denied.

42.5.1.2.1 Challenge/response mechanism

When SJC is in system JTAG mode, the authentication process is as follows:

1. Shift the output challenge instruction to the IR
2. Pass through Capture-DR state and by performing Shift-DR operations Challenge code can be accessed from TDO.
3. Shift the enter response instruction to the IR. By performing Shift-DR, operations enter response code value through TDI. As Update-DR state is entered, response code is compared with the correct one.

In fixed challenge-response pair mode, each chip has its individual challenge/response pair which is determined at manufacturing time. The SJC compares the user's response to the expected response.

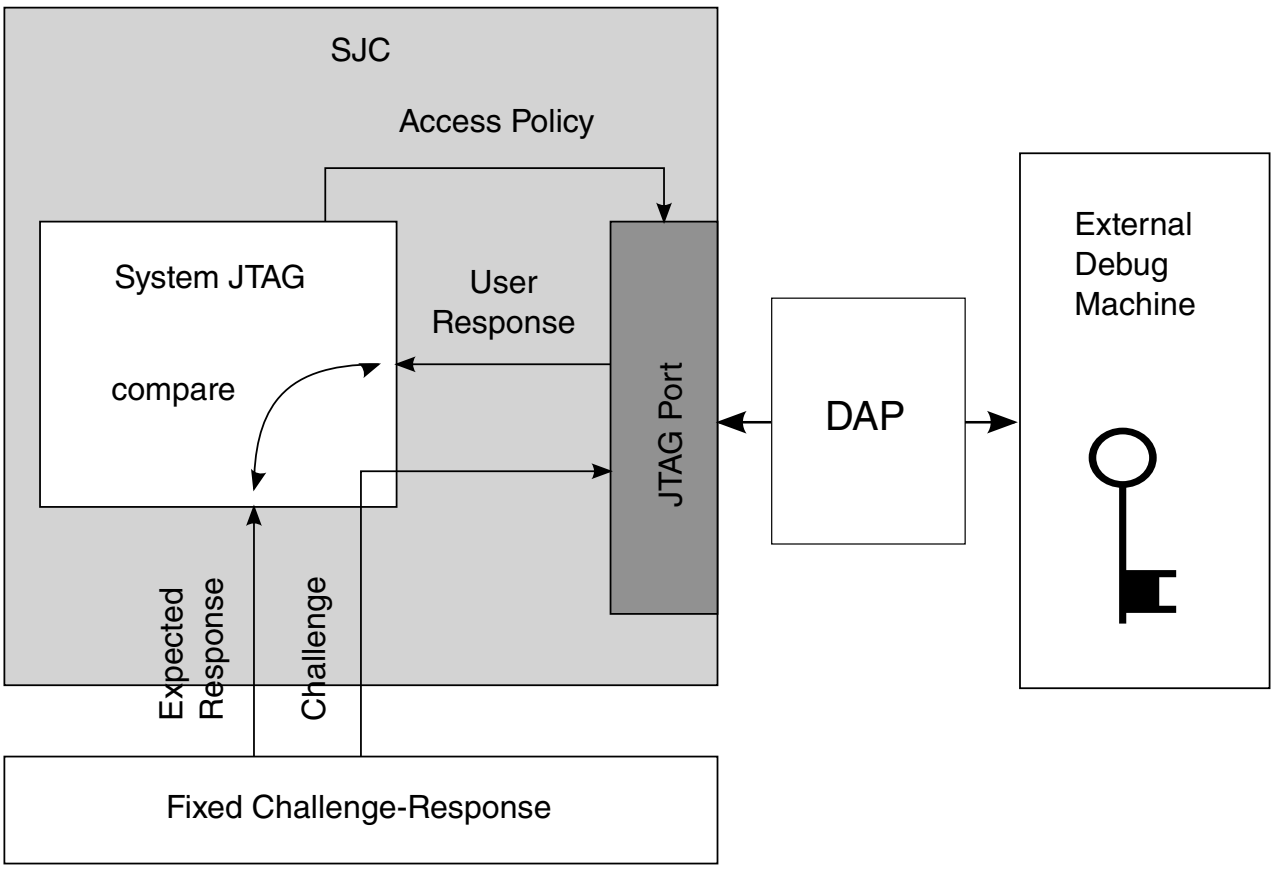


Figure 42-7. Mode #2 - Secure JTAG with fixed challenge/response pair

42.5.1.3 Mode 3: JTAG enabled—low security

In the JTAG enabled security mode, all JTAG features are enabled.

42.5.2 Software enabled JTAG

To increase the flexibility of SJC, an option to enable the JTAG using software is available in secure JTAG mode. By writing a 1 to the HAB_JDE (HAB JTAG DEBUG ENABLE) bit in the eFuse controller module (OCOTP_CTRL), the JTAG is opened, regardless of its security mode. It is the responsibility of software to assert or negate this bit.

Additionally, a corresponding LOCK bit is available (in the eFuse control module) to ensure that only trusted software is able to set the HAB_JDE bit. When the LOCK bit is set, no future change of HAB_JDE is possible, until the next POR (power-on-reset) cycle.

The platform initialization software should set the LOCK bit before transferring control to the application code.

The software JTAG enable allows JTAG enabling without activating the challenge/response mechanism (which requires JTAG access tool enhancement or special hardware). The JTAG software enable does not allow debug in case of boot or memory fault as it requires reset before entering debug.

This feature can be permanently blocked by burning the dedicated fuse.

NOTE

The software enabled JTAG feature reduces the overall security level of the system as it relies on software protections. If this feature is not required, it is strongly recommended to burn the JTAG_HEO fuse to disable this feature.

42.6 Programmable registers

This section lists additional registers to the standard accessible JTAG registers (per IEEE1149.1 standard). The following registers are accessed using the ExtraDebug mechanism, controlled by the ENABLE_ExtraDebug IR instruction.

NOTE

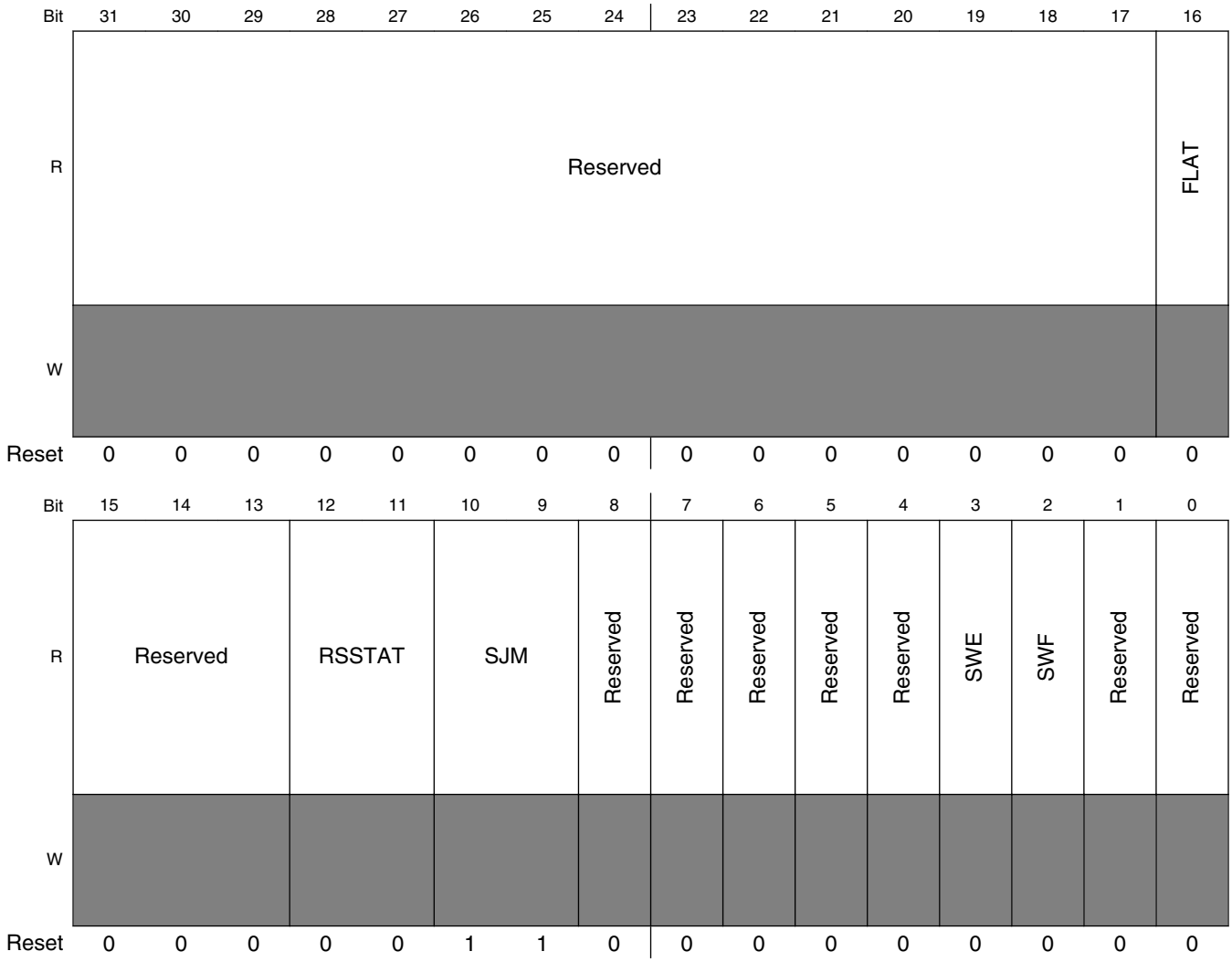
SJC registers are only accessible by JTAG interface and are not memory mapped to processor address space.

SJC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
5	Security Status Register (SJC_SSR)	32	R	0000_0600h	42.6.1/1760

42.6.1 Security Status Register (SJC_SSR)

Address: 0h base + 5h offset = 5h



SJC_SSR field descriptions

Field	Description
31–17 Reserved	This field is reserved.
16 FLAT	Fuse latch 0 Fuses are not yet latched 1 Fuses have been latched
15–13 Reserved	This field is reserved.
12–11 RSSTAT	Response status 00 Response not entered

Table continues on the next page...

SJC_SSR field descriptions (continued)

Field	Description
	01 Response entered but not verified 10 Response entered and is incorrect 11 Response is correct
10–9 SJM	SJC mode. Secure JTAG mode, as set by external fuses. These bits do not include the setting of the BSF fuse. 00 No debug (Mode 1) 01 Secure JTAG (Mode 2) 10 JTAG enabled (Mode 3) 11 Reserved
8 Reserved	This field is reserved.
7 Reserved	This field is reserved.
6 Reserved	This field is reserved.
5 Reserved	This field is reserved.
4 Reserved	This field is reserved.
3 SWE	SW JTAG enable status 1 Enabled 0 Disabled
2 SWF	Software JTAG enable fuse. Status of the no software disable JTAG fuse 0 Intact - software enable possible 1 Intact - no software enable possible
1 Reserved	This field is reserved.
0 Reserved	This field is reserved.

Chapter 43
JTAG Controller (JTAGC)

43.1 Chip-specific JTAGC information

Table 43-1. Reference links to related information

Topic	Related module	Reference
Full description	JTAGC	JTAGC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

43.1.1 JTAG Controller (JTAGC)

The JTAGC provides the means to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from the JTAGC block is communicated in serial format. The following table shows the configuration of the JTAGC.

Table 43-2. JTAGC configuration

Parameter	Description
Name	JTAGC
Supported standard verison	IEEE 1149.1-2001
Instances	1
Configurable features	NA
Interface speed	
External I/O pins	TCK, TDI, TDO, TMS
Interrupts	NA
DMA requests	NA

NOTE

The JTAGC is used in final testing, though the scan chains will be connected to the DAP, taking advantage of the Arm tool suites for independently disabling/enabling the various scan chains. e-fuses are to be used to permanently disable each of the scan chains as well as the interface between the DAP and external memory.

43.1.2 Boundary scan

It is required to have CA7 domain powered to run Boundary Scan on CA7 domain related pads: PTD* ~ PTF* GPIOs, RESET1_B, DDR* and HSIC*.

CA7 domain can be powered ON by application code or alternatively by means of JTAGC instruction "0110". This instruction selects a special register for connection as the shift path between TDI and TDO. This register implements 2 bits to enable CA7 regulator and/or CA7 power switch as follows:

Bit 0 : Forces Power Switch ON

Bit 1 : Forces A7 regulator ON

Once enabled, regulator cannot be disabled, so the device needs to go through a true POR.

If A7 is not powered ON, then Boundary scan will be performed only on CM4 domain pads : PTA*, PTB* , RESET1_B

43.2 Introduction

The JTAGC block provides the means to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from the JTAGC block is communicated in serial format.

43.2.1 Block diagram

The following is a simplified block diagram of the JTAG Controller (JTAGC) block. See the chip-specific configuration information as well as Register description for more information about the JTAGC registers.

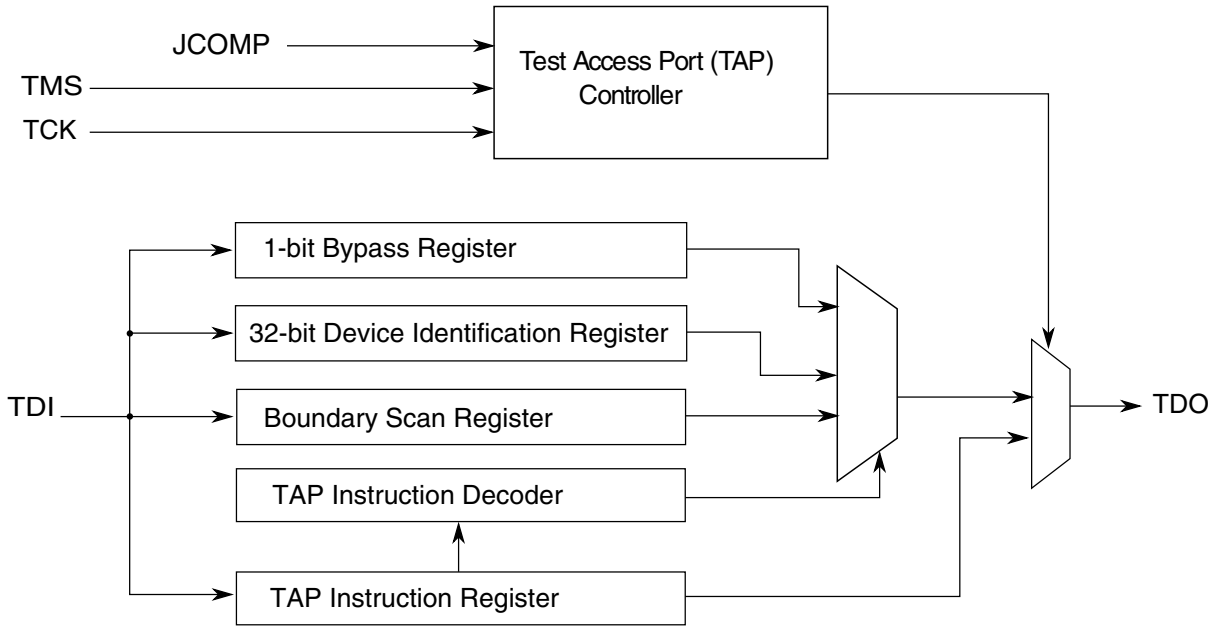


Figure 43-1. JTAG (IEEE 1149.1) block diagram

43.2.2 Features

The JTAGC block is compliant with the IEEE 1149.1-2001 standard, and supports the following features:

- IEEE 1149.1-2001 Test Access Port (TAP) interface
  - Four pins (TDI, TMS, TCK, and TDO)
- JCOMP input that provides reset control
- Instruction register that supports several IEEE 1149.1–2001 defined instructions as well as several public and private device-specific instructions (see JTAGC block instructions for a list of supported instructions).
- Bypass register, boundary scan register, and device identification register
- TAP controller state machine that controls the operation of the data registers, instruction register, and associated circuitry

43.2.3 Modes of operation

The JTAGC block uses JCOMP and a power-on reset indication as its primary reset signals. Several IEEE 1149.1–2001 defined test modes are supported, as well as a bypass mode.

43.2.3.1 Reset

The JTAGC block is placed in reset when:

- Power-on reset is asserted
- JCOMP is negated
- TMS input is held high for enough consecutive rising edges of TCK to sequence the TAP controller state machine into the Test-Logic-Reset state

Holding TMS high for five consecutive rising edges of TCK guarantees entry into the Test-Logic-Reset state regardless of the current TAP controller state. Asserting power-on reset or setting JCOMP to a value other than the value required to enable the JTAGC block results in asynchronous entry into the reset state.

When in reset, the following actions occur:

- The TAP controller is forced into the Test-Logic-Reset state, thereby disabling the test logic and allowing normal operation of the on-chip system logic to continue unhindered.
- The instruction register is loaded with the IDCODE instruction.

43.2.3.2 IEEE 1149.1–2001 defined test modes

The JTAGC block supports several IEEE 1149.1–2001 defined test modes. A test mode is selected by loading the appropriate instruction into the instruction register when the JTAGC is enabled. Supported test instructions include EXTEST, HIGHZ, CLAMP, SAMPLE, and SAMPLE/PRELOAD.

Each instruction defines the set of data register(s) that may operate and interact with the on-chip system logic when the instruction is current. Only one test data register path is enabled to shift data between TDI and TDO for each instruction.

The boundary scan register is enabled for serial access between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are active. The single-bit bypass register shift stage is enabled for serial access between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

The functionality of each test mode is explained in more detail in JTAGC block instructions.

43.2.3.3 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. When in bypass mode, the single-bit bypass shift register is used to provide a minimum-length serial path to shift data between TDI and TDO.

43.3 External signal description

The JTAGC consists of a set of signals that connect to off-chip development tools and allow access to test support functions. The JTAGC signals are outlined in the following table and described in the following sections.

Table 43-3. JTAG signal properties

Name	I/O	Function	Reset state
TCK	Input	Test clock	Weak pullup
TDI	Input	Test data in	Weak pullup
TDO	Output	Test data out	High Z[1]
TMS	Input	Test mode select	Weak pullup
JCOMP	Input	JTAG compliancy	Weak pulldown

1. TDO output buffer enable is negated when the JTAGC is not in the Shift-IR or Shift-DR states. A weak pull may be implemented at the TDO pad for use when JTAGC is inactive.

43.3.1 Test clock input (TCK)

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

43.3.2 Test data input (TDI)

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

43.3.3 Test data output (TDO)

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is tristateable and is actively driven only in the Shift-IR and Shift-DR states of the TAP controller state machine, which is described in TAP controller state machine.

43.3.4 Test mode select (TMS)

Test Mode Select (TMS) is an input pin used to sequence the IEEE 1149.1-2001 test control state machine. TMS is sampled on the rising edge of TCK.

43.3.5 JCOMP JTAG compliancy

The JCOMP signal provides IEEE 1149.1-2001 compatibility and provides the ability to share the TAP. The JTAGC TAP controller is enabled when JCOMP is set to the JTAGC enable encoding—otherwise the JTAGC TAP controller remains in reset.

43.4 Register description

This section provides a detailed description of the JTAGC block registers accessible through the TAP interface, including data registers and the instruction register. Individual bit-level descriptions and reset states of each register are included. These registers are not memory-mapped and can only be accessed through the TAP.

43.4.1 Instruction register

The JTAGC block uses a 4-bit instruction register as shown in the following figure.

The instruction register allows instructions to be loaded into the block to select the test to be performed, or the test data register to be accessed, or both. Instructions are shifted in through TDI when the TAP controller is in the Shift-IR state, and latched on the falling edge of TCK in the Update-IR state. The latched instruction value can only be changed in the Update-IR and Test-Logic-Reset TAP controller states.

Synchronous entry into the Test-Logic-Reset state results in the IDCODE instruction being loaded on the falling edge of TCK. Asynchronous entry into the Test-Logic-Reset state results in asynchronous loading of the IDCODE instruction. During the Capture-IR

TAP controller state, the instruction shift register is loaded with the value 0b1, making this value the register's read value when the TAP controller is sequenced into the Shift-IR state.

	3	2	1	0
R	0	0	0	1
W	Instruction Code			
Reset:	0	0	0	1

Figure 43-2. Instruction register

43.4.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

After entry into the Capture-DR state, the single-bit shift register is set to a logic 0. Therefore, the first bit shifted out after selecting the bypass register is always a logic 0.

43.4.3 Device identification register

The device identification (JTAG ID) register, shown in the following figure, allows the revision number, part number, manufacturer, and design center responsible for the design of the part to be determined through the TAP.

The device identification register is selected for serial data transfer between TDI and TDO when the IDCODE instruction is active. Entry into the Capture-DR state when the device identification register is selected loads the IDCODE into the shift register to be shifted out on TDO in the Shift-DR state. No action occurs in the Update-DR state.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Part Revision Number				Design Center						Part Identification Number					
W																
Reset	PRN				DC						PIN					

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Part Identification Number				Manufacturer Identity Code											1
W																
Reset	PIN (contd.)				MIC											1

The following table describes the device identification register functions.

Table 43-4. Device identification register field descriptions

Field	Description
PRN	Part Revision Number. Contains the revision number of the part. Value is cccc.
DC	Design Center. Indicates the design center. Value is 0x22.
PIN	Part Identification Number. Contains the part number of the device. Value is 0bccccccccccc.
MIC	Manufacturer Identity Code. Contains the reduced Joint Electron Device Engineering Council (JEDEC) ID. Value is 0x00E.
IDCODE ID	IDCODE Register ID. Identifies this register as the device identification register and not the bypass register. Always set to 1.

43.4.4 Boundary scan register

The boundary scan register is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are active. It is used to:

- Capture input pin data
- Force fixed values on output pins
- Select a logic value and direction for bidirectional pins

Each bit of the boundary scan register represents a separate boundary scan register cell, as described in the IEEE 1149.1-2001 standard and discussed in Boundary scan. The size of the boundary scan register and bit ordering is device-dependent and can be found in the device BSDL file.

43.5 Functional description

This section explains the JTAGC functional description.

43.5.1 JTAGC reset configuration

When in reset, the TAP controller is forced into the Test-Logic-Reset state, thus disabling the test logic and allowing normal operation of the on-chip system logic. In addition, the instruction register is loaded with the IDCODE instruction.

43.5.2 IEEE 1149.1-2001 (JTAG) TAP

The JTAGC block uses the IEEE 1149.1-2001 TAP for accessing registers. This port can be shared with other TAP controllers on the MCU. Ownership of the port is determined by the value of the currently loaded instruction.

Data is shifted between TDI and TDO through the selected register starting with the least significant bit, as illustrated in the following figure. This applies for the instruction register, test data registers, and the bypass register.

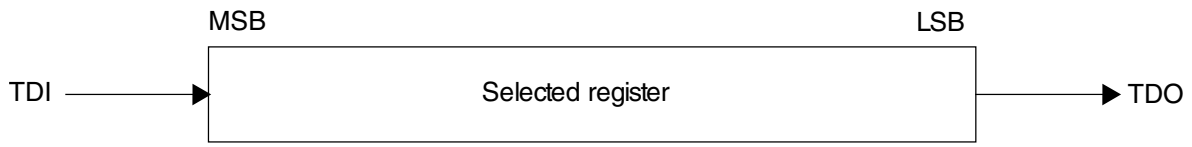
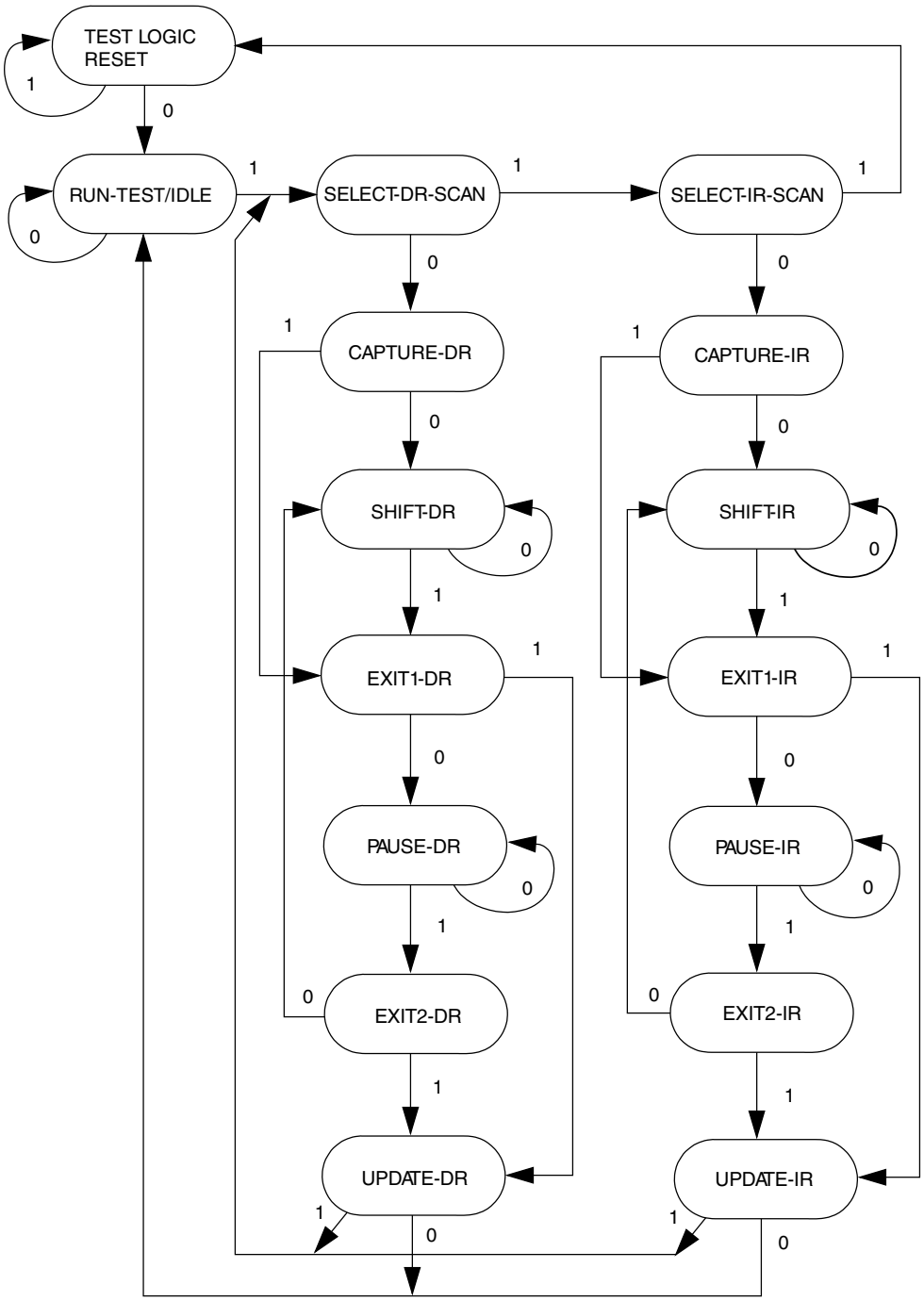


Figure 43-3. Shifting data through a register

43.5.3 TAP controller state machine

The TAP controller is a synchronous state machine that interprets the sequence of logical values on the TMS pin.

The following figure shows the machine's states. The value shown next to each state is the value of the TMS signal sampled on the rising edge of the TCK signal. As the figure shows, holding TMS at logic 1 when clocking TCK through a sufficient number of rising edges also causes the state machine to enter the Test-Logic-Reset state.



The value shown adjacent to each state transition in this figure represents the value of TMS at the time of a rising edge of TCK.

Figure 43-4. IEEE 1149.1-2001 TAP controller finite state machine

43.5.3.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting JCOMP to a logic 1 value.

43.5.3.2 Selecting an IEEE 1149.1-2001 register

Access to the JTAGC data registers is achieved by loading the instruction register with any of the JTAGC block instructions when the JTAGC is enabled. Instructions are shifted in via the Select-IR-Scan path and loaded in the Update-IR state. At this point, all data register access is performed via the Select-DR-Scan path.

The Select-DR-Scan path is used to read or write the register data by shifting in the data (LSB first) during the Shift-DR state. When reading a register, the register value is loaded into the IEEE 1149.1-2001 shifter during the Capture-DR state. When writing a register, the value is loaded from the IEEE 1149.1-2001 shifter to the register during the Update-DR state. When reading a register, there is no requirement to shift out the entire register contents. Shifting may be terminated after the required number of bits have been acquired.

43.5.4 JTAGC block instructions

The JTAGC block implements the IEEE 1149.1-2001 defined instructions listed in the following table. This section gives an overview of each instruction. See the IEEE 1149.1-2001 standard for more details. All undefined opcodes are reserved.

Table 43-5. 4-bit JTAG instructions

Instruction	Code[3:0]	Instruction summary
IDCODE	0000	Selects device identification register for shift
SAMPLE/PRELOAD	0010	Selects boundary scan register for shifting, sampling, and preloading without disturbing functional operation
SAMPLE	0011	Selects boundary scan register for shifting and sampling without disturbing functional operation
EXTEST	0100	Selects boundary scan register and applies preloaded values to output pins. **NOTE:** Execution of this instruction asserts functional reset.
TEST_LEAKAGE[1,2]	0101	Selects bypass register when tristating all output pins and asserts the jtag_leakage signal
Factory debug reserved	0110	Intended for factory debug only. See chip-specific information for possible alternate usage of this code.
Factory debug reserved	0111	Intended for factory debug only
Arm JTAG-DP Reserved	1000	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
HIGHZ	1001	Selects bypass register and tristates all output pins. **NOTE:** Execution of this instruction asserts functional reset.

Table continues on the next page...

Table 43-5. 4-bit JTAG instructions (continued)

Instruction	Code[3:0]	Instruction summary
Arm JTAG-DP Reserved	1010	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
Arm JTAG-DP Reserved	1011	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
CLAMP	1100	Selects bypass register and applies preloaded values to output pins. **NOTE:** Execution of this instruction asserts functional reset.
Arm JTAG-DP Reserved	1110	This instruction goes the Arm JTAG-DP controller. See the Arm JTAG-DP documentation for more information.
BYPASS	1111	Selects bypass register for data operations

1. Instruction for manufacturing purposes only.
2. Assertion of power-on reset or setting JCOMP for non-JTAGC operation required to exit this instruction.

43.5.4.1 IDCODE instruction

IDCODE selects the 32-bit device identification register as the shift path between TDI and TDO. This instruction allows interrogation of the MCU to determine its version number and other part identification data. IDCODE is the instruction placed into the instruction register when the JTAGC block is reset.

43.5.4.2 SAMPLE/PRELOAD instruction

The SAMPLE/PRELOAD instruction has two functions:

- The SAMPLE portion of the instruction obtains a sample of the system data and control signals present at the MCU input pins, and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE/PRELOAD instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. Both the data capture and the shift operation are transparent to system operation.
- The PRELOAD portion of the instruction initializes the boundary scan register cells before selecting the EXTEST or CLAMP instructions to perform boundary scan tests. This is achieved by shifting in initialization data to the boundary scan register during the Shift-DR state. The initialization data is transferred to the parallel outputs of the boundary scan register cells on the falling edge of TCK in the Update-DR state. The data is applied to the external output pins by the EXTEST or CLAMP instruction. System operation is not affected.

43.5.4.3 SAMPLE instruction

The SAMPLE instruction obtains a sample of the system data and control signals present at the MCU input pins, and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. There is no defined action in the Update-DR state. Both the data capture and the shift operation are transparent to system operation.

43.5.4.4 EXTEST external test instruction

EXTEST selects the boundary scan register as the shift path between TDI and TDO. It allows testing of off-chip circuitry and board-level interconnections by driving preloaded data contained in the boundary scan register onto the system output pins. Typically, the preloaded data is loaded into the boundary scan register using the SAMPLE/PRELOAD instruction before the selection of EXTEST. EXTEST asserts the internal system reset for the MCU to force a predictable internal state when performing external boundary scan operations.

43.5.4.5 TEST_LEAKAGE instruction

The TEST_LEAKAGE instruction forces the jtag_leakage output signal to high. It is intended to tristate all output pad buffers and disable all of the part's pad input buffers except JCOMP and TEST. The jtag_leakage signal is asserted at the falling edge of TCK following the TAP controller state machine transition from the Update-IR state to the Run-Test-Idle state. When asserted, the part disables TCK, TMS, and TDI inputs and forces them to a logic 1. The TAP controller state machine remains in the Run-Test-Idle state until the JCOMP input is set to a value other than the JTAGC enable encoding. TEST_LEAKAGE also asserts the internal system reset for the MCU to force a predictable internal state.

43.5.4.6 ENABLE_SOC_DATA1 instruction

The ENABLE_SOC_DATA1 instruction captures SoC data and selects the SOC_DATA register for connection as the shift path between TDI and TDO.

43.5.4.7 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. When HIGHZ is active all output drivers are placed in an inactive drive state (for example, high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

43.5.4.8 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register when the bypass register is selected as the serial path between TDI and TDO. CLAMP enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) when conducting an EXTEST type of instruction through the boundary scan register. CLAMP also asserts the internal system reset for the MCU to force a predictable internal state.

43.5.4.9 BYPASS instruction

BYPASS selects the bypass register, creating a single-bit shift register path between TDI and TDO. BYPASS enhances test efficiency by reducing the overall shift path when no test operation of the MCU is required. This allows more rapid movement of test data to and from other components on a board that are required to perform test functions. When the BYPASS instruction is active the system logic operates normally.

43.5.5 Boundary scan

The boundary scan technique allows signals at component boundaries to be controlled and observed through the shift-register stage associated with each pad. Each stage is part of a larger boundary scan register cell, and cells for each pad are interconnected serially to form a shift-register chain around the border of the design. The boundary scan register consists of this shift-register chain, and is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are loaded. The shift-register chain contains a serial input and serial output, as well as clock and control signals.

43.6 Initialization/application information

The test logic is a static logic design, and TCK can be stopped in either a high or low state without loss of data. However, the system clock is not synchronized to TCK internally. Any mixed operation using both the test logic and the system functional logic requires external synchronization.

To initialize the JTAGC block and enable access to registers, the following sequence is required:

1. Set the JCOMP signal to the JTAGC enable value, thereby enabling the JTAGC TAP controller.
2. Load the appropriate instruction for the test or action to be performed.

Chapter 44
Integrated Interchip Sound (I2S) / Synchronous Audio Interface (SAI)

44.1 Chip-specific SAI information

Table 44-1. Reference links to related information

Topic	Related module	Reference
Full description	SAI	SAI
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

44.1.1 Synchronous Audio Interface (SAI/I2S)

The Synchronous Audio Interface (SAI) implements full-duplex serial interfaces with frame synchronization such as I2S, AC97, and CODEC/DSP interfaces. This device implements 2 SAI modules. The following table shows the configuration of the SAI.

Table 44-2. SAI configuration

Parameter	Description
Name	Synchronous Audio Interface (SAI)
Instances	2
Configurable features	SAI0: • 2 TX/RX channels • 16 word FIFOs • Max 32 words per frame SAI1: • 4 TX/RX channels • 16 word FIFOs • Max 32 words per frame

Table continues on the next page...

Table 44-2. SAI configuration (continued)

Parameter	Description
Interface speed	25 MHz
External I/O pins	• SAI0: TXD[1:0], TX_BCLK, TX_FS, RX_FS, RX_BCLK, RXD[1:0], MCLK. See attached IOMUXC spreadsheet for pin details. • SAI1: TXD[1:0], TX_BCLK, RX_FS, TX_FS, RX_BCLK, RXD[1:0], MCLK, TXD[3:2], and RXD[3:2]. See attached IOMUXC spreadsheet for detail

44.1.2 Signal names

The signal names used in this device are different from the signal names mentioned in the block guide. Refer the following tables for getting the actual signal names implemented in i.MX 7ULP for both instances.

Table 44-3. SAI0/I2S0 signal names

Signal name in block guide	Actual signal names used in i.MX 7ULP
TXDATA[1:0]	I2S0_TXD[1:0]
TXBCLK	I2S0_TX_BCLK
RXBCLK	I2S0_RX_BCLK
TXSYNC	I2S0_TX_FS
RXSYNC	I2S0_RX_FS
RXDATA[1:0]	I2S0_RXD[1:0]
MCLK	I2S0_MCLK

Table 44-4. SAI1/I2S1 signal names

Signal name in block guide	Actual signal names used in i.MX 7ULP
TXDATA[1:0]	I2S1_TXD[1:0]
TXBCLK	I2S1_TX_BCLK
RXBCLK	I2S1_RX_BCLK
TXSYNC	I2S1_TX_FS
RXSYNC	I2S1_RX_FS
RXDATA[1:0]	I2S1_RXD[1:0]
MCLK	I2S1_MCLK

44.1.3 Master clock options

For details on clocking and availability of master clock (MCLK) options in SAI0 and SAI1, see SAI clocking. MCLK option can be selected through RCR2[MSEL] bitfield. To enable I2S_MCLK0 or I2S_MCLK1 as an output on a PAD, it is necessary to configure the function on IOMUXC0 and set pad OBE=1.

44.2 Introduction

The I^2S (or I2S) module provides a synchronous audio interface (SAI) that supports full-duplex serial interfaces with frame synchronization such as I^2S, AC97, TDM, and codec/DSP interfaces.

44.2.1 Features

Note that some of the features are not supported across all SAI instances; see the chip-specific information in the first section of this chapter.

- Transmitter with independent bit clock and frame sync supporting 2 data lines
- Receiver with independent bit clock and frame sync supporting 2 data lines
- Each data line can support a maximum Frame size of 32 words
- Word size of between 8-bits and 32-bits
- Word size configured separately for first word and remaining words in frame
- Asynchronous 16 x 32-bit FIFO for each transmit and receive data line
- Supports graceful restart after FIFO error
- Supports automatic restart after FIFO error without software intervention
- Supports packing of 8-bit and 16-bit data into each 32-bit FIFO word
- Supports combining multiple data line FIFOs into single data line FIFO

44.2.2 Block diagram

The following block diagram also shows the module clocks.

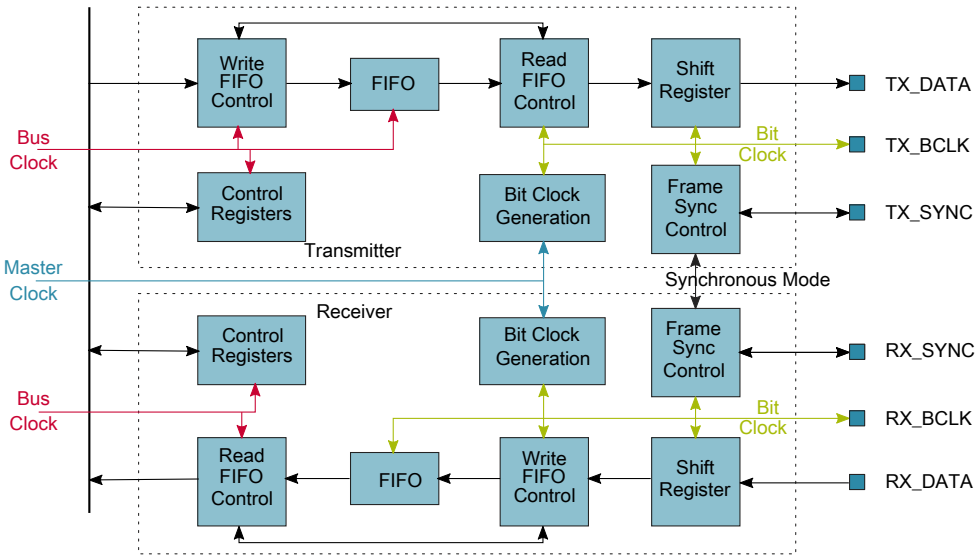


Figure 44-1. I²S/SAI block diagram

44.2.3 Modes of operation

Module power modes include Run mode, Stop modes, low-leakage stop modes, and Debug mode.

44.2.3.1 Run mode

In Run mode, the SAI transmitter and receiver operate normally.

44.2.3.2 Stop modes

In Stop mode, the SAI transmitter and/or receiver can continue operating provided the appropriate Stop Enable bit is set (TCSR[STOPE] and/or RCSR[STOPE], respectively), and provided the transmitter and/or receiver is/are using an externally generated bit clock or an Audio Master Clock that remains operating in Stop mode. The SAI transmitter and/or receiver can generate an asynchronous interrupt to wake the CPU from Stop mode.

In Stop mode, if the Transmitter Stop Enable (TCSR[STOPE]) bit is clear, the transmitter is disabled after completing the current transmit frame, and, if the Receiver Stop Enable (RCSR[STOPE]) bit is clear, the receiver is disabled after completing the current receive frame. Entry into Stop mode is prevented (not acknowledged) while waiting for the transmitter and receiver to be disabled at the end of the current frame.

44.2.3.3 Low-leakage stop modes

When entering low-leakage stop modes, the Stop Enable (TCSR[STOPE] and RCSR[STOPE]) bits are ignored and the SAI is disabled after completing the current transmit and receive Frames. Entry into stop mode is prevented (not acknowledged) while waiting for the transmitter and receiver to be disabled at the end of the current frame.

44.2.3.4 Debug mode

In Debug mode, the SAI transmitter and/or receiver can continue operating provided the Debug Enable bit is set. When TCSR[DBGE] or RCSR[DBGE] bit is clear and Debug mode is entered, the SAI is disabled after completing the current transmit or receive frame. The transmitter and receiver bit clocks are not affected by Debug mode.

44.3 External signals

Name	Function	I/O
TX_BCLK	Transmit Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
TX_SYNC	Transmit Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O
TX_DATA[1:0]	Transmit Data. The transmit data is generated synchronously by the bit clock and is tristated whenever not transmitting a word.	O
RX_BCLK	Receive Bit Clock. The bit clock is an input when externally generated and an output when internally generated.	I/O
RX_SYNC	Receive Frame Sync. The frame sync is an input sampled synchronously by the bit clock when externally generated and an output generated synchronously by the bit clock when internally generated.	I/O
RX_DATA[1:0]	Receive Data. The receive data is sampled synchronously by the bit clock.	I

44.4 Memory map and register definition

A read or write access to an address from offset 0x100 and above will result in a bus error.

44.4.1 I2S register descriptions

44.4.1.1 I2S Memory map

I2S0 base address: 4103_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0300_0000h
4h	Parameter Register (PARAM)	32	RO	0005_0402h
8h	SAI Transmit Control Register (TCSR)	32	RW	0000_0000h
Ch	SAI Transmit Configuration 1 Register (TCR1)	32	RW	0000_0000h
10h	SAI Transmit Configuration 2 Register (TCR2)	32	RW	0000_0000h
14h	SAI Transmit Configuration 3 Register (TCR3)	32	RW	0000_0000h
18h	SAI Transmit Configuration 4 Register (TCR4)	32	RW	0000_0000h
1Ch	SAI Transmit Configuration 5 Register (TCR5)	32	RW	0000_0000h
20h - 24h	SAI Transmit Data Register (TDR0 - TDR1)	32	WORZ	0000_0000h
40h - 44h	SAI Transmit FIFO Register (TFR0 - TFR1)	32	RO	0000_0000h
60h	SAI Transmit Mask Register (TMR)	32	RW	0000_0000h
88h	SAI Receive Control Register (RCSR)	32	RW	0000_0000h
8Ch	SAI Receive Configuration 1 Register (RCR1)	32	RW	0000_0000h
90h	SAI Receive Configuration 2 Register (RCR2)	32	RW	0000_0000h
94h	SAI Receive Configuration 3 Register (RCR3)	32	RW	0000_0000h
98h	SAI Receive Configuration 4 Register (RCR4)	32	RW	0000_0000h
9Ch	SAI Receive Configuration 5 Register (RCR5)	32	RW	0000_0000h
A0h - A4h	SAI Receive Data Register (RDR0 - RDR1)	32	RO	0000_0000h
C0h - C4h	SAI Receive FIFO Register (RFR0 - RFR1)	32	RO	0000_0000h
E0h	SAI Receive Mask Register (RMR)	32	RW	0000_0000h

44.4.1.2 Version ID Register (VERID)

44.4.1.2.1 Offset

Register	Offset
VERID	0h

44.4.1.2.2 Function

Contains version numbers for the module design and feature set.

44.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000000000000b - Standard feature set.

44.4.1.3 Parameter Register (PARAM)

44.4.1.3.1 Offset

Register	Offset
PARAM	4h

44.4.1.3.2 Function

Contains parameter values that were implemented in the module.

44.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FRAME			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FIFO				0				DATALINE			
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0

44.4.1.3.4 Fields

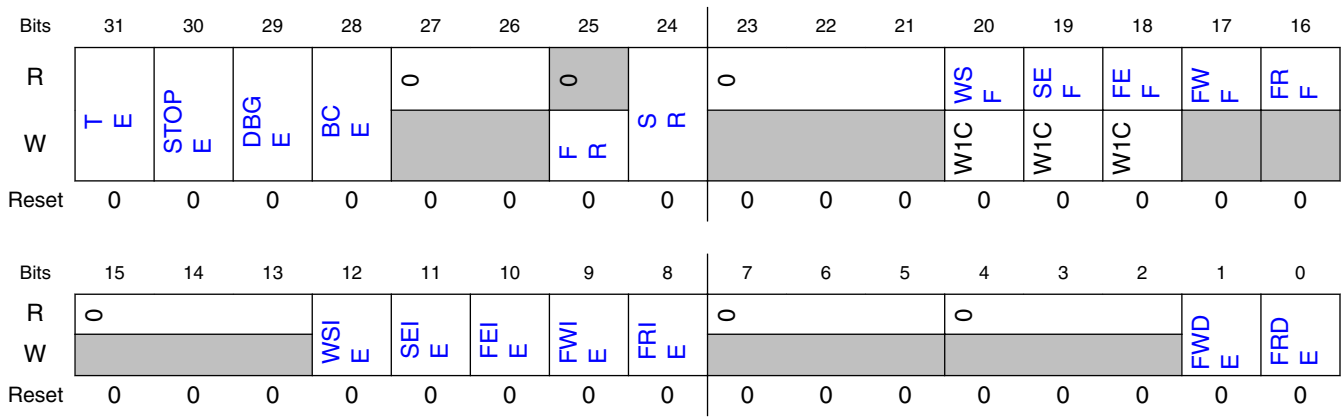
Field	Function
31-20 —	Reserved
19-16 FRAME	Frame Size The maximum number of slots per frame is 2^FRAME.
15-12 —	Reserved
11-8 FIFO	FIFO Size The number of words in each FIFO is 2^FIFO.
7-4 —	Reserved
3-0 DATALINE	Number of Datalines The number of datalines implemented.

44.4.1.4 SAI Transmit Control Register (TCSR)

44.4.1.4.1 Offset

Register	Offset
TCSR	8h

44.4.1.4.2 Diagram



44.4.1.4.3 Fields

Field	Function
31 TE	Transmitter Enable Enables/disables the transmitter. When software clears this field, the transmitter remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmitter is disabled. 1b - Transmitter is enabled, or transmitter has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures transmitter operation in Stop mode. The Stop Enable field is ignored and the transmitter is disabled in all low-leakage stop modes. 0b - Transmitter disabled in Stop mode. 1b - Transmitter enabled in Stop mode.
29 DBGE	Debug Enable Enables/disables transmitter operation in Debug mode. The transmit bit clock is not affected by debug mode. 0b - Transmitter is disabled in Debug mode, after completing the current frame. 1b - Transmitter is enabled in Debug mode.
28 BCE	Bit Clock Enable

Table continues on the next page...

Field	Function
	Enables the transmit bit clock, separately from the TE. This field is automatically set whenever TE is set. When software clears this field, the transmit bit clock remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmit bit clock is disabled. 1b - Transmit bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the transmitter is disabled or the FIFO error flag is set. 0b - No effect. 1b - FIFO reset.
24 SR	Software Reset When set, resets the internal transmitter logic including the FIFO read and write pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled transmit FIFO has underrun. Write a logic 1 to this field to clear this flag. 0b - Transmit underrun not detected. 1b - Transmit underrun detected.
17 FWF	FIFO Warning Flag Indicates that an enabled transmit FIFO is empty. 0b - No enabled transmit FIFO is empty. 1b - Enabled transmit FIFO is empty.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled transmit channel FIFO is less than or equal to the transmit FIFO watermark. 0b - Transmit FIFO watermark has not been reached. 1b - Transmit FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts.

Table continues on the next page...

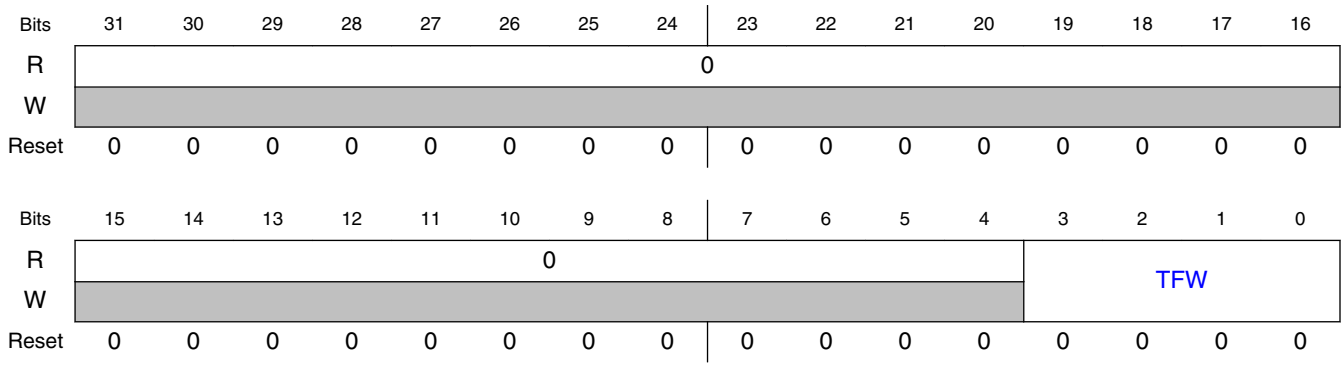
Field	Function
	0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

44.4.1.5 SAI Transmit Configuration 1 Register (TCR1)

44.4.1.5.1 Offset

Register	Offset
TCR1	Ch

44.4.1.5.2 Diagram



44.4.1.5.3 Fields

Field	Function
31-4 —	Reserved
3-0 TFW	Transmit FIFO Watermark Configures the watermark level for all enabled transmit channels.

44.4.1.6 SAI Transmit Configuration 2 Register (TCR2)

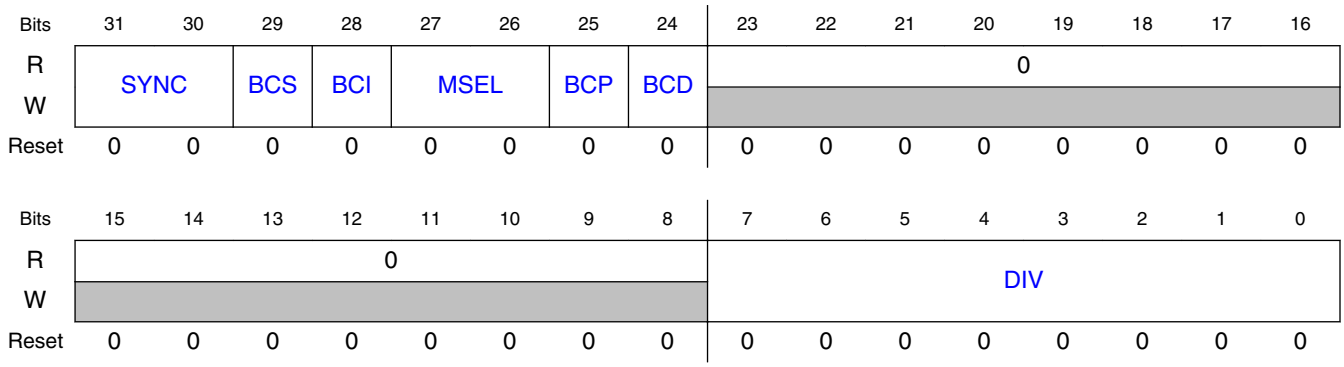
44.4.1.6.1 Offset

Register	Offset
TCR2	10h

44.4.1.6.2 Function

This register must not be altered when TCSR[TE] is set.

44.4.1.6.3 Diagram



44.4.1.6.4 Fields

Field	Function
31-30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the receiver or other SAI peripheral must be configured for asynchronous operation. 00b - Asynchronous mode. 01b - Synchronous with receiver. 10b - Synchronous with another SAI transmitter. 11b - Synchronous with another SAI receiver.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the transmitter. When the transmitter is configured in asynchronous mode and this bit is set, the transmitter is clocked by the receiver bit clock (RX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the transmitter continues to use the transmit frame sync (TX_SYNC). When the transmitter is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the transmitter bit clock (TX_BCLK) but use the receiver frame sync (RX_SYNC). This field has no effect when synchronous to another SAI peripheral. 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the transmitter is delayed by the pad output delay (the transmitter is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the transmitter when this bit is set. In synchronous mode, this bit allows the transmitter to use the slave mode timing from the datasheet, while the receiver uses the master mode timing. This field has no effect when configured for an externally generated bit clock or when synchronous to another SAI peripheral . 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26	MCLK Select

Table continues on the next page...

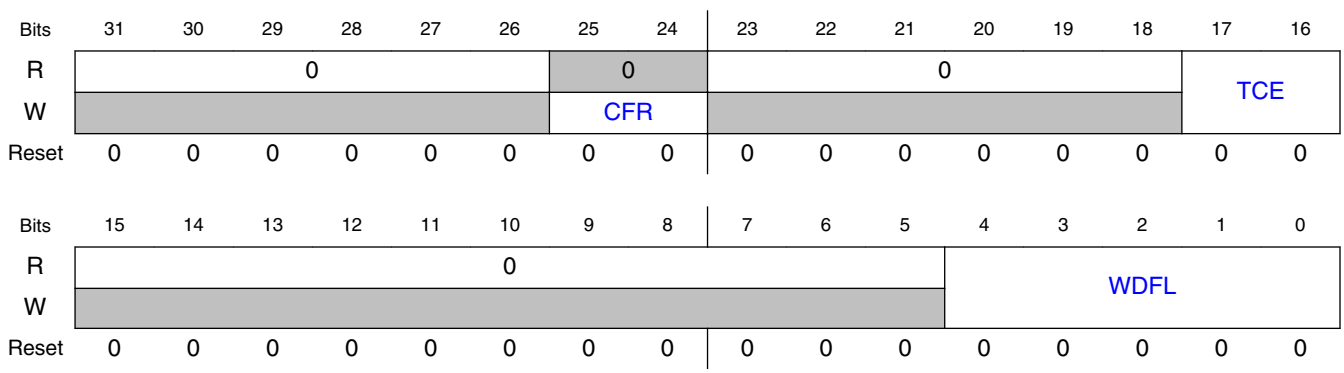
Field	Function
MSEL	Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

44.4.1.7 SAI Transmit Configuration 3 Register (TCR3)

44.4.1.7.1 Offset

Register	Offset
TCR3	14h

44.4.1.7.2 Diagram



44.4.1.7.3 Fields

Field	Function
31-26 —	Reserved
25-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field will always return zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of transmit channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to transmit channel 1 FIFO pointer and bit position 25 refers to transmit channel 2 FIFO pointer. Setting bit 24 resets transmit channel 1 FIFO pointer, and setting bit 25 enables transmit channel 2 FIFO pointer. Setting bit N will reset transmit channel N FIFO pointer. 0b - No effect. 1b - Transmit data channel N FIFO is reset.
23-18 —	Reserved
17-16 TCE	Transmit Channel Enable Enables the corresponding data channel for transmit operation. Changing TCE field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for transmit operation. The width of TCE field = the number of transmit channels (call it N). For example, if TCE field is 2 bits wide, then bit position 16 refers to transmit channel 1 and bit position 17 refers to transmit channel 2. Setting bit 16 enables transmit channel 1, and setting bit 17 enables transmit channel 2. Setting bit N will enable transmit channel N. 0b - Transmit data channel N is disabled. 1b - Transmit data channel N is enabled.
15-5 —	Reserved
4-0 WDFL	Word Flag Configuration Configures which word sets the start of word flag. The value written must be one less than the word number. For example, writing 0 configures the first word in the frame. When configured to a value greater than TCR4[FRSZ], then the start of word flag is never set.

44.4.1.8 SAI Transmit Configuration 4 Register (TCR4)

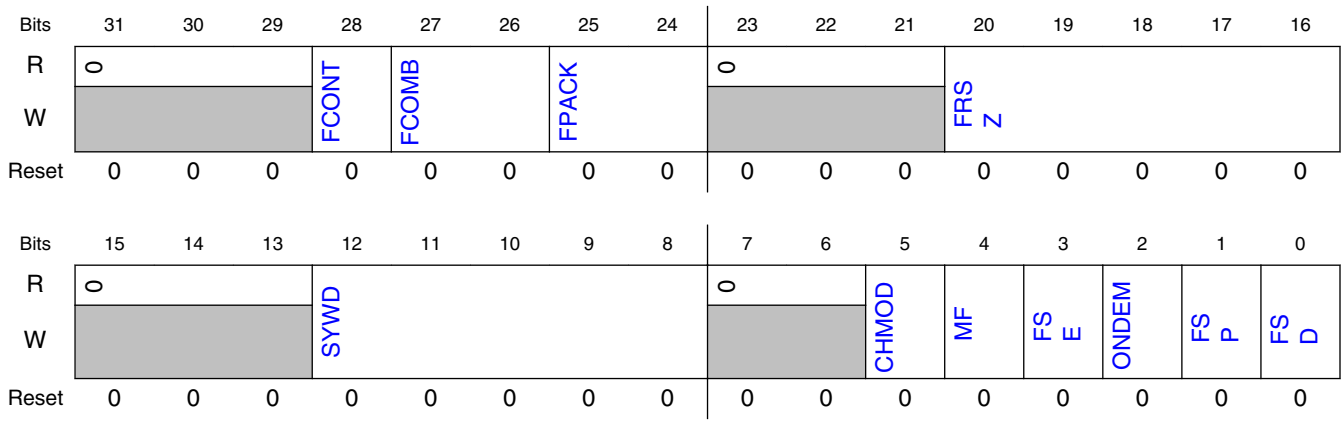
44.4.1.8.1 Offset

Register	Offset
TCR4	18h

44.4.1.8.2 Function

This register must not be altered when TCSR[TE] is set.

44.4.1.8.3 Diagram



44.4.1.8.4 Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when the SAI will continue transmitting after a FIFO error has been detected. 0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27-26 FCOMB	FIFO Combine Mode When FIFO combine mode is enabled for FIFO writes, software writing to any FIFO data register will alternate the write among the enabled data channel FIFOs. For example, if two data channels are enabled then the first write will be performed to the first enabled data channel FIFO and the second write will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO writes will reset the pointer back to the first enabled data channel. When FIFO combine mode is enabled for FIFO reads from the transmit shift registers, the transmit data channel output will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked word will be transmitted from the first enabled data channel FIFO and the second unmasked word will be transmitted from the second enabled data channel FIFO. Since the first word of the frame is always transmitted from the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels. 00b - FIFO combine mode disabled. 01b - FIFO combine mode enabled on FIFO reads (from transmit shift registers). 10b - FIFO combine mode enabled on FIFO writes (by software). 11b - FIFO combine mode enabled on FIFO reads (from transmit shift registers) and writes (by software).

Table continues on the next page...

Field	Function
25-24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are loaded from the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO write pointer will only increment when the full 32-bit FIFO word has been written by software. 00b - FIFO packing is disabled 01b - Reserved 10b - 8-bit FIFO packing is enabled 11b - 16-bit FIFO packing is enabled
23-21 —	Reserved
20-16 FRSZ	Frame size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 CHMOD	Channel Mode Configures if transmit data pins are configured for TDM mode or Output mode. 0b - TDM mode, transmit data pins are tri-stated when slots are masked or channels are disabled. 1b - Output mode, transmit data pins are never tri-stated and will output zero when slots are masked or channels are disabled.
4 MF	MSB First Configures whether the LSB or the MSB is transmitted first. 0b - LSB is transmitted first. 1b - MSB is transmitted first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync.

Field	Function
	0b - Frame sync is generated externally in Slave mode. 1b - Frame sync is generated internally in Master mode.

44.4.1.9 SAI Transmit Configuration 5 Register (TCR5)

44.4.1.9.1 Offset

Register	Offset
TCR5	1Ch

44.4.1.9.2 Function

This register must not be altered when TCSR[TE] is set.

44.4.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WNW					0			W0W				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FBT					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.9.4 Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved

Table continues on the next page...

Field	Function
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit transmitted for each word in the frame. If configured for MSB First, the index of the next bit transmitted is one less than the current bit transmitted. If configured for LSB First, the index of the next bit transmitted is one more than the current bit transmitted. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

44.4.1.10 SAI Transmit Data Register (TDR0 - TDR1)

44.4.1.10.1 Offset

Register	Offset
TDR0	20h
TDR1	24h

44.4.1.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.10.3 Fields

Field	Function
31-0 TDR	Transmit Data Register Writes to this register when the transmit FIFO is not full will push the data written into the transmit data FIFO. Writes to this register when the transmit FIFO is full are ignored.

44.4.1.11 SAI Transmit FIFO Register (TFR0 - TFR1)

44.4.1.11.1 Offset

Register	Offset
TFR0	40h
TFR1	44h

44.4.1.11.2 Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

44.4.1.11.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WCP	0										WFP				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											RFP				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.11.4 Fields

Field	Function
31 WCP	Write Channel Pointer

Table continues on the next page...

Field	Function
	When FIFO Combine mode is enabled for writes, indicates that this data channel is the next FIFO to be written. 0b - No effect. 1b - FIFO combine is enabled for FIFO writes and this FIFO will be written on the next FIFO write.
30-21 —	Reserved
20-16 WFP	Write FIFO Pointer FIFO write pointer for transmit data channel.
15-5 —	Reserved
4-0 RFP	Read FIFO Pointer FIFO read pointer for transmit data channel.

44.4.1.12 SAI Transmit Mask Register (TMR)

44.4.1.12.1 Offset

Register	Offset
TMR	60h

44.4.1.12.2 Function

This register is double-buffered and updates:

1. When TCSR[TE] is first set
2. At the end of each frame.

This allows the masked words in each frame to change from frame to frame.

44.4.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.12.4 Fields

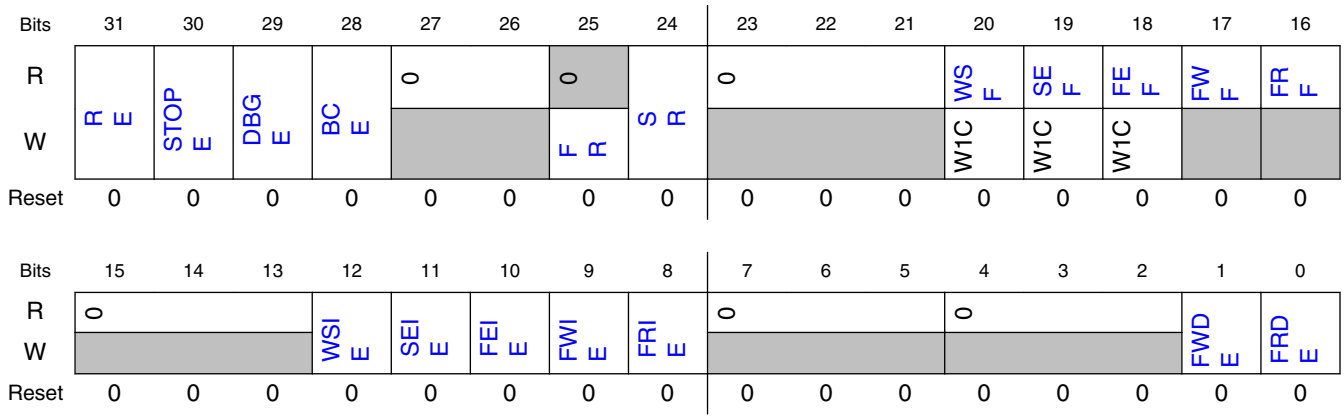
Field	Function
31-0 TWM	Transmit Word Mask Configures whether the transmit word is masked (transmit data pins are tri-stated or drive zero and transmit data not read from FIFO) for the corresponding word in the frame. 00000000000000000000000000000000b - Word N is enabled. 00000000000000000000000000000001b - Word N is masked. The transmit data pins are tri-stated or drive zero when masked.

44.4.1.13 SAI Receive Control Register (RCSR)

44.4.1.13.1 Offset

Register	Offset
RCSR	88h

44.4.1.13.2 Diagram



44.4.1.13.3 Fields

Field	Function
31 RE	Receiver Enable Enables/disables the receiver. When software clears this field, the receiver remains enabled, and this bit remains set, until the end of the current frame. 0b - Receiver is disabled. 1b - Receiver is enabled, or receiver has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures receiver operation in Stop mode. The Stop Enable field is ignored and the receiver is disabled in all low-leakage stop modes. 0b - Receiver disabled in Stop mode. 1b - Receiver enabled in Stop mode.
29 DBGE	Debug Enable Enables/disables receiver operation in Debug mode. The receive bit clock is not affected by Debug mode. 0b - Receiver is disabled in Debug mode, after completing the current frame. 1b - Receiver is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the receive bit clock, separately from RE. This field is automatically set whenever RE is set. When software clears this field, the receive bit clock remains enabled, and this field remains set, until the end of the current frame. 0b - Receive bit clock is disabled. 1b - Receive bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the receiver is disabled or the FIFO error flag is set. 0b - No effect. 1b - FIFO reset.
24	Software Reset

Table continues on the next page...

Field	Function
SR	Resets the internal receiver logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled receive FIFO has overflowed. Write a logic 1 to this field to clear this flag. 0b - Receive overflow not detected. 1b - Receive overflow detected.
17 FWF	FIFO Warning Flag Indicates that an enabled receive FIFO is full. 0b - No enabled receive FIFO is full. 1b - Enabled receive FIFO is full.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled receive channel FIFO is greater than the receive FIFO watermark. 0b - Receive FIFO watermark not reached. 1b - Receive FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.

Table continues on the next page...

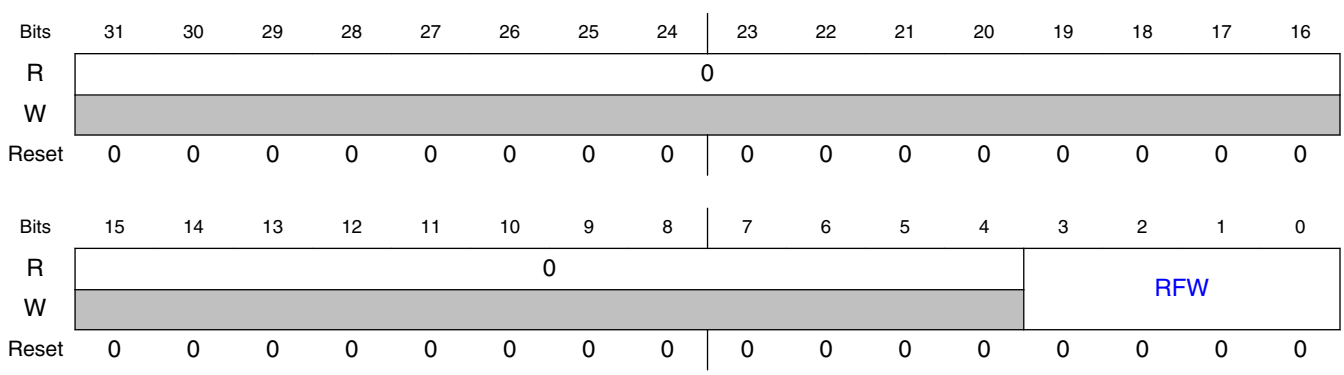
Field	Function
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

44.4.1.14 SAI Receive Configuration 1 Register (RCR1)

44.4.1.14.1 Offset

Register	Offset
RCR1	8Ch

44.4.1.14.2 Diagram



44.4.1.14.3 Fields

Field	Function
31-4 —	Reserved
3-0 RFW	Receive FIFO Watermark Configures the watermark level for all enabled receiver channels.

44.4.1.15 SAI Receive Configuration 2 Register (RCR2)

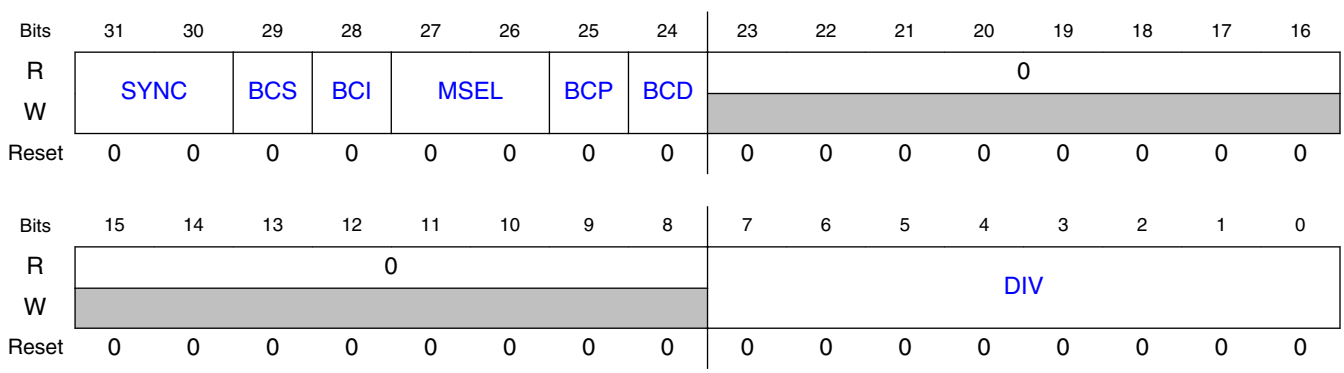
44.4.1.15.1 Offset

Register	Offset
RCR2	90h

44.4.1.15.2 Function

This register must not be altered when RCSR[RE] is set.

44.4.1.15.3 Diagram



44.4.1.15.4 Fields

Field	Function
31-30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the transmitter or other SAI peripheral must be configured for asynchronous operation.

Table continues on the next page...

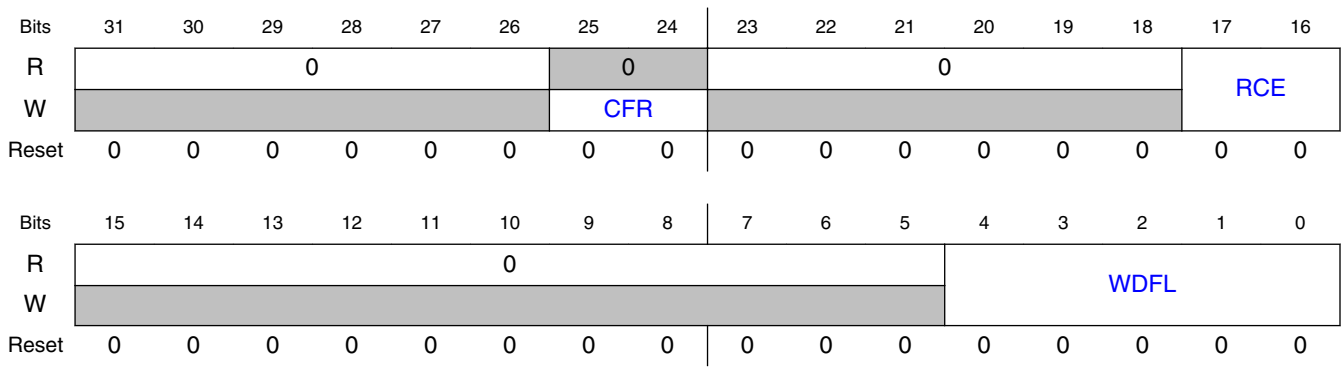
Field	Function
	00b - Asynchronous mode. 01b - Synchronous with transmitter. 10b - Synchronous with another SAI receiver. 11b - Synchronous with another SAI transmitter.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the receiver. When the receiver is configured in asynchronous mode and this bit is set, the receiver is clocked by the transmitter bit clock (TX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the receiver continues to use the receiver frame sync (RX_SYNC). When the receiver is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the receiver bit clock (RX_BCLK) but use the transmitter frame sync (TX_SYNC). This field has no effect when synchronous to another SAI peripheral. 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the receiver is delayed by the pad output delay (the receiver is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the receiver when this bit is set. In synchronous mode, this bit allows the receiver to use the slave mode timing from the datasheet, while the transmitter uses the master mode timing. This field has no effect when configured for an externally generated bit clock or when synchronous to another SAI peripheral . 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the availability and chip-specific meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit Clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit Clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

44.4.1.16 SAI Receive Configuration 3 Register (RCR3)

44.4.1.16.1 Offset

Register	Offset
RCR3	94h

44.4.1.16.2 Diagram



44.4.1.16.3 Fields

Field	Function
31-26 —	Reserved
25-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field will always return zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of receive channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to receive channel 1 FIFO pointer and bit position 25 refers to receive channel 2 FIFO pointer. Setting bit 24 resets receive channel 1 FIFO pointer, and setting bit 25 enables receive channel 2 FIFO pointer. Setting bit N will reset receive channel N FIFO pointer. 0b - No effect. 1b - Receive data channel N FIFO is reset.
23-18 —	Reserved
17-16 RCE	Receive Channel Enable Enables the corresponding data channel for receive operation. Changing this field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for receive operation.

Table continues on the next page...

Field	Function
	The width of RCE field = the number of receive channels (call it N). For example, if RCE field is 2 bits wide, then bit position 16 refers to receive channel 1 and bit position 17 refers to receive channel 2. Setting bit 16 enables receive channel 1, and setting bit 17 enables receive channel 2. Setting bit N will enable receive channel N. 0b - Receive data channel N is disabled. 1b - Receive data channel N is enabled.
15-5 —	Reserved
4-0 WDFL	Word Flag Configuration Configures which word the start of word flag is set. The value written should be one less than the word number (for example, write zero to configure for the first word in the frame). When configured to a value greater than the Frame Size field, then the start of word flag is never set.

44.4.1.17 SAI Receive Configuration 4 Register (RCR4)

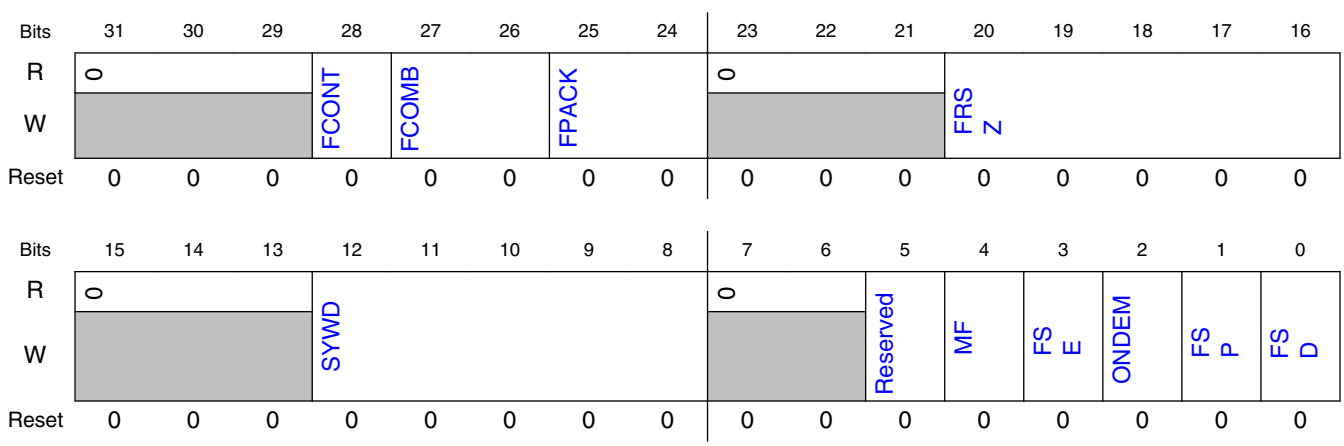
44.4.1.17.1 Offset

Register	Offset
RCR4	98h

44.4.1.17.2 Function

This register must not be altered when RCSR[RE] is set.

44.4.1.17.3 Diagram



44.4.1.17.4 Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when the SAI will continue receiving after a FIFO error has been detected. 0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27-26 FCOMB	FIFO Combine Mode When FIFO combine mode is enabled for FIFO reads, software reading any FIFO data register will alternate the read among the enabled data channel FIFOs. For example, if two data channels are enabled then the first read will be performed to the first enabled data channel FIFO and the second read will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO reads will reset the pointer back to the first enabled data channel. When FIFO combine mode is enabled for FIFO writes from the receive shift registers, the first enabled data channel input will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked received word will be stored in the first enabled data channel FIFO and the second unmasked received word will be stored in the second enabled data channel FIFO. Since the first word of the frame is always stored in the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels. 00b - FIFO combine mode disabled. 01b - FIFO combine mode enabled on FIFO writes (from receive shift registers). 10b - FIFO combine mode enabled on FIFO reads (by software). 11b - FIFO combine mode enabled on FIFO writes (from receive shift registers) and reads (by software).
25-24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are stored to the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO read pointer will only increment when the full 32-bit FIFO word has been read by software. 00b - FIFO packing is disabled 01b - Reserved. 10b - 8-bit FIFO packing is enabled 11b - 16-bit FIFO packing is enabled
23-21 —	Reserved
20-16 FRSZ	Frame Size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words.
15-13 —	Reserved
12-8 SYWD	Sync Width

Table continues on the next page...

Field	Function
	Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 —	Reserved. Software should only write zero to this bit.
4 MF	MSB First Configures whether the LSB or the MSB is received first. 0b - LSB is received first. 1b - MSB is received first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame Sync is generated externally in Slave mode. 1b - Frame Sync is generated internally in Master mode.

44.4.1.18 SAI Receive Configuration 5 Register (RCR5)

44.4.1.18.1 Offset

Register	Offset
RCR5	9Ch

44.4.1.18.2 Function

This register must not be altered when RCSR[RE] is set.

44.4.1.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WNW					0			W0W				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FBT					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.18.4 Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit received for each word in the frame. If configured for MSB First, the index of the next bit received is one less than the current bit received. If configured for LSB First, the index of the next bit received is one more than the current bit received. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

44.4.1.19 SAI Receive Data Register (RDR0 - RDR1)

44.4.1.19.1 Offset

Register	Offset
RDR0	A0h
RDR1	A4h

44.4.1.19.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.19.3 Fields

Field	Function
31-0 RDR	Receive Data Register Reads from this register when the receive FIFO is not empty will return the data from the top of the receive FIFO. Reads from this register when the receive FIFO is empty are ignored.

44.4.1.20 SAI Receive FIFO Register (RFR0 - RFR1)

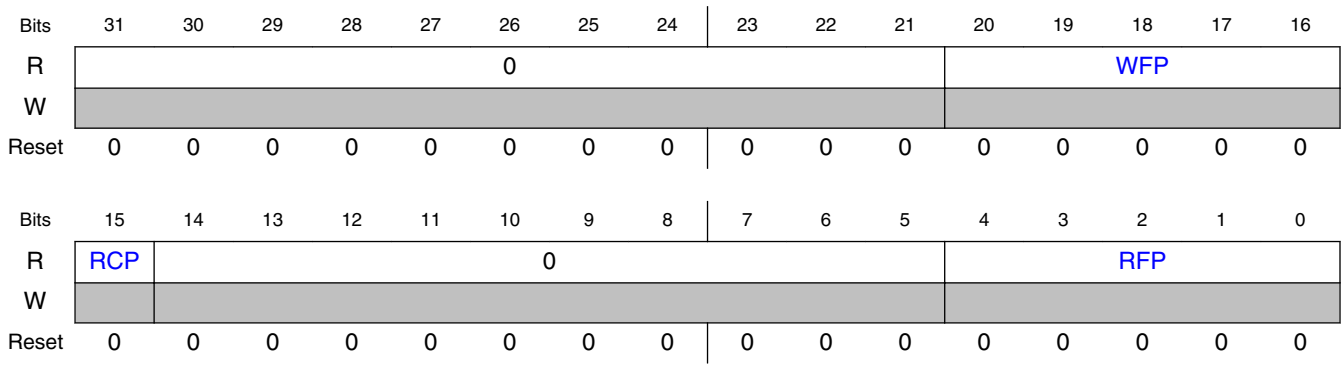
44.4.1.20.1 Offset

Register	Offset
RFR0	C0h
RFR1	C4h

44.4.1.20.2 Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

44.4.1.20.3 Diagram



44.4.1.20.4 Fields

Field	Function
31-21 —	Reserved
20-16 WFP	Write FIFO Pointer FIFO write pointer for receive data channel.
15 RCP	Receive Channel Pointer When FIFO Combine mode is enabled for reads, indicates that this data channel is the next FIFO to be read. 0b - No effect. 1b - FIFO combine is enabled for FIFO reads and this FIFO will be read on the next FIFO read.
14-5 —	Reserved
4-0 RFP	Read FIFO Pointer FIFO read pointer for receive data channel.

44.4.1.21 SAI Receive Mask Register (RMR)

44.4.1.21.1 Offset

Register	Offset
RMR	E0h

44.4.1.21.2 Function

This register is double-buffered and updates:

1. When RCSR[RE] is first set
2. At the end of each frame

This allows the masked words in each frame to change from frame to frame.

44.4.1.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.1.21.4 Fields

Field	Function
31-0 RWM	Receive Word Mask Configures whether the receive word is masked (received data ignored and not written to receive FIFO) for the corresponding word in the frame. 00000000000000000000000000000000b - Word N is enabled. 00000000000000000000000000000001b - Word N is masked.

44.4.2 I2S register descriptions

44.4.2.1 I2S Memory map

I2S1 base address: 410A_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0300_0000h
4h	Parameter Register (PARAM)	32	RO	0005_0404h
8h	SAI Transmit Control Register (TCSR)	32	RW	0000_0000h
Ch	SAI Transmit Configuration 1 Register (TCR1)	32	RW	0000_0000h
10h	SAI Transmit Configuration 2 Register (TCR2)	32	RW	0000_0000h
14h	SAI Transmit Configuration 3 Register (TCR3)	32	RW	0000_0000h
18h	SAI Transmit Configuration 4 Register (TCR4)	32	RW	0000_0000h
1Ch	SAI Transmit Configuration 5 Register (TCR5)	32	RW	0000_0000h
20h - 2Ch	SAI Transmit Data Register (TDR0 - TDR3)	32	WORZ	0000_0000h
40h - 4Ch	SAI Transmit FIFO Register (TFR0 - TFR3)	32	RO	0000_0000h
60h	SAI Transmit Mask Register (TMR)	32	RW	0000_0000h
88h	SAI Receive Control Register (RCSR)	32	RW	0000_0000h
8Ch	SAI Receive Configuration 1 Register (RCR1)	32	RW	0000_0000h
90h	SAI Receive Configuration 2 Register (RCR2)	32	RW	0000_0000h
94h	SAI Receive Configuration 3 Register (RCR3)	32	RW	0000_0000h
98h	SAI Receive Configuration 4 Register (RCR4)	32	RW	0000_0000h
9Ch	SAI Receive Configuration 5 Register (RCR5)	32	RW	0000_0000h
A0h - ACh	SAI Receive Data Register (RDR0 - RDR3)	32	RO	0000_0000h
C0h - CCh	SAI Receive FIFO Register (RFR0 - RFR3)	32	RO	0000_0000h
E0h	SAI Receive Mask Register (RMR)	32	RW	0000_0000h

44.4.2.2 Version ID Register (VERID)

44.4.2.2.1 Offset

Register	Offset
VERID	0h

44.4.2.2.2 Function

Contains version numbers for the module design and feature set.

44.4.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the specification.
15-0 FEATURE	Feature Specification Number This read only field returns the feature set number. 0000000000000000b - Standard feature set.

44.4.2.3 Parameter Register (PARAM)

44.4.2.3.1 Offset

Register	Offset
PARAM	4h

44.4.2.3.2 Function

Contains parameter values that were implemented in the module.

44.4.2.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												FRAME			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FIFO				0				DATALINE			
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

44.4.2.3.4 Fields

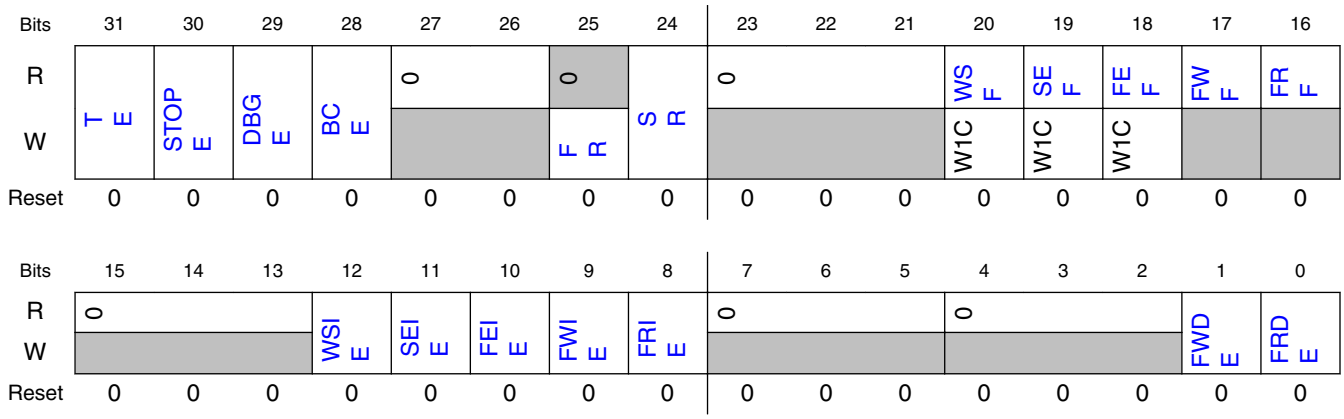
Field	Function
31-20 —	Reserved
19-16 FRAME	Frame Size The maximum number of slots per frame is 2^FRAME.
15-12 —	Reserved
11-8 FIFO	FIFO Size The number of words in each FIFO is 2^FIFO.
7-4 —	Reserved
3-0 DATALINE	Number of Datalines The number of datalines implemented.

44.4.2.4 SAI Transmit Control Register (TCSR)

44.4.2.4.1 Offset

Register	Offset
TCSR	8h

44.4.2.4.2 Diagram



44.4.2.4.3 Fields

Field	Function
31 TE	Transmitter Enable Enables/disables the transmitter. When software clears this field, the transmitter remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmitter is disabled. 1b - Transmitter is enabled, or transmitter has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures transmitter operation in Stop mode. The Stop Enable field is ignored and the transmitter is disabled in all low-leakage stop modes. 0b - Transmitter disabled in Stop mode. 1b - Transmitter enabled in Stop mode.
29 DBGE	Debug Enable Enables/disables transmitter operation in Debug mode. The transmit bit clock is not affected by debug mode. 0b - Transmitter is disabled in Debug mode, after completing the current frame. 1b - Transmitter is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the transmit bit clock, separately from the TE. This field is automatically set whenever TE is set. When software clears this field, the transmit bit clock remains enabled, and this bit remains set, until the end of the current frame. 0b - Transmit bit clock is disabled. 1b - Transmit bit clock is enabled.
27-26 —	Reserved
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the transmitter is disabled or the FIFO error flag is set. 0b - No effect. 1b - FIFO reset.

Table continues on the next page...

Field	Function
24 SR	Software Reset When set, resets the internal transmitter logic including the FIFO read and write pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled transmit FIFO has underrun. Write a logic 1 to this field to clear this flag. 0b - Transmit underrun not detected. 1b - Transmit underrun detected.
17 FWF	FIFO Warning Flag Indicates that an enabled transmit FIFO is empty. 0b - No enabled transmit FIFO is empty. 1b - Enabled transmit FIFO is empty.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled transmit channel FIFO is less than or equal to the transmit FIFO watermark. 0b - Transmit FIFO watermark has not been reached. 1b - Transmit FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt.

Table continues on the next page...

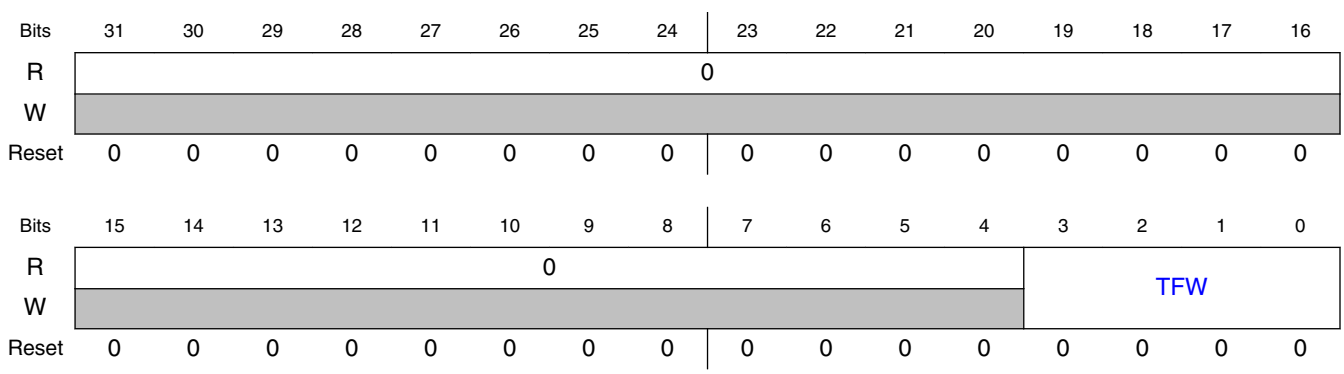
Field	Function
	1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

44.4.2.5 SAI Transmit Configuration 1 Register (TCR1)

44.4.2.5.1 Offset

Register	Offset
TCR1	Ch

44.4.2.5.2 Diagram



44.4.2.5.3 Fields

Field	Function
31-4 —	Reserved
3-0 TFW	Transmit FIFO Watermark Configures the watermark level for all enabled transmit channels.

44.4.2.6 SAI Transmit Configuration 2 Register (TCR2)

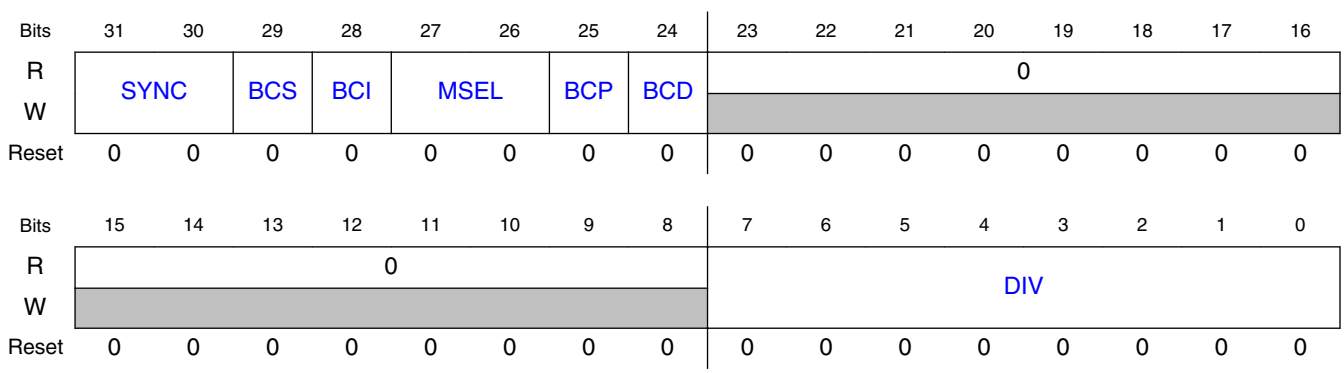
44.4.2.6.1 Offset

Register	Offset
TCR2	10h

44.4.2.6.2 Function

This register must not be altered when TCSR[TE] is set.

44.4.2.6.3 Diagram



44.4.2.6.4 Fields

Field	Function
31-30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the receiver or other SAI peripheral must be configured for asynchronous operation.

Table continues on the next page...

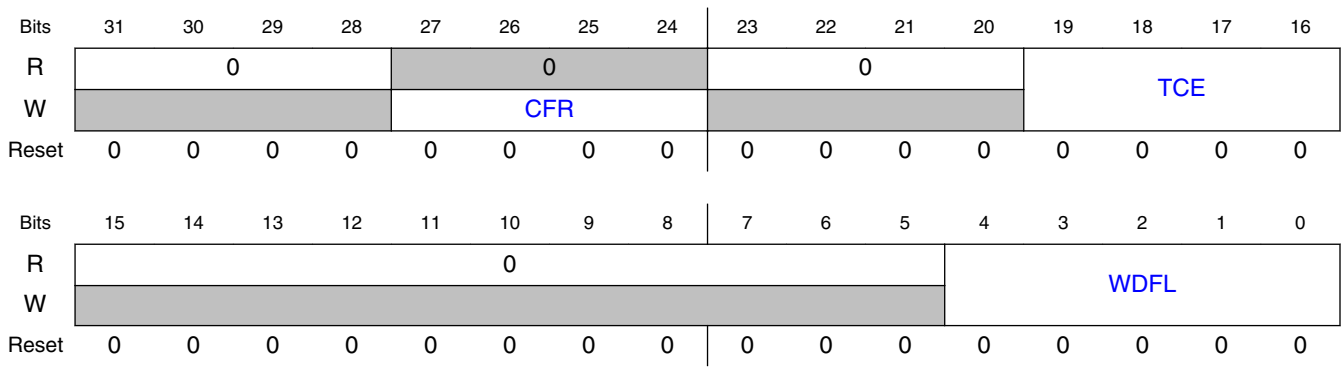
Field	Function
	00b - Asynchronous mode. 01b - Synchronous with receiver. 10b - Synchronous with another SAI transmitter. 11b - Synchronous with another SAI receiver.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the transmitter. When the transmitter is configured in asynchronous mode and this bit is set, the transmitter is clocked by the receiver bit clock (RX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the transmitter continues to use the transmit frame sync (TX_SYNC). When the transmitter is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the transmitter bit clock (TX_BCLK) but use the receiver frame sync (RX_SYNC). This field has no effect when synchronous to another SAI peripheral. 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the transmitter is delayed by the pad output delay (the transmitter is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the transmitter when this bit is set. In synchronous mode, this bit allows the transmitter to use the slave mode timing from the datasheet, while the receiver uses the master mode timing. This field has no effect when configured for an externally generated bit clock or when synchronous to another SAI peripheral . 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26 MSEL	MCLK Select Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

44.4.2.7 SAI Transmit Configuration 3 Register (TCR3)

44.4.2.7.1 Offset

Register	Offset
TCR3	14h

44.4.2.7.2 Diagram



44.4.2.7.3 Fields

Field	Function
31-28 —	Reserved
27-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field will always return zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of transmit channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to transmit channel 1 FIFO pointer and bit position 25 refers to transmit channel 2 FIFO pointer. Setting bit 24 resets transmit channel 1 FIFO pointer, and setting bit 25 enables transmit channel 2 FIFO pointer. Setting bit N will reset transmit channel N FIFO pointer. 0b - No effect. 1b - Transmit data channel N FIFO is reset.
23-20 —	Reserved
19-16 TCE	Transmit Channel Enable Enables the corresponding data channel for transmit operation. Changing TCE field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for transmit operation.

Table continues on the next page...

Field	Function
	The width of TCE field = the number of transmit channels (call it N). For example, if TCE field is 2 bits wide, then bit position 16 refers to transmit channel 1 and bit position 17 refers to transmit channel 2. Setting bit 16 enables transmit channel 1, and setting bit 17 enables transmit channel 2. Setting bit N will enable transmit channel N. 0b - Transmit data channel N is disabled. 1b - Transmit data channel N is enabled.
15-5 —	Reserved
4-0 WDFL	Word Flag Configuration Configures which word sets the start of word flag. The value written must be one less than the word number. For example, writing 0 configures the first word in the frame. When configured to a value greater than TCR4[FRSZ], then the start of word flag is never set.

44.4.2.8 SAI Transmit Configuration 4 Register (TCR4)

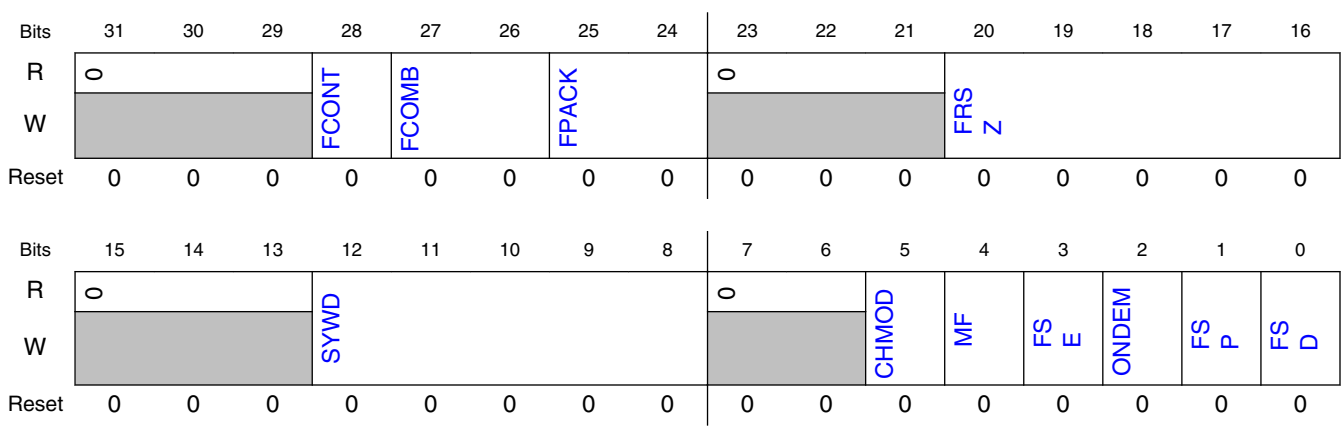
44.4.2.8.1 Offset

Register	Offset
TCR4	18h

44.4.2.8.2 Function

This register must not be altered when TCSR[TE] is set.

44.4.2.8.3 Diagram



44.4.2.8.4 Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when the SAI will continue transmitting after a FIFO error has been detected. 0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27-26 FCOMB	FIFO Combine Mode When FIFO combine mode is enabled for FIFO writes, software writing to any FIFO data register will alternate the write among the enabled data channel FIFOs. For example, if two data channels are enabled then the first write will be performed to the first enabled data channel FIFO and the second write will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO writes will reset the pointer back to the first enabled data channel. When FIFO combine mode is enabled for FIFO reads from the transmit shift registers, the transmit data channel output will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked word will be transmitted from the first enabled data channel FIFO and the second unmasked word will be transmitted from the second enabled data channel FIFO. Since the first word of the frame is always transmitted from the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels. 00b - FIFO combine mode disabled. 01b - FIFO combine mode enabled on FIFO reads (from transmit shift registers). 10b - FIFO combine mode enabled on FIFO writes (by software). 11b - FIFO combine mode enabled on FIFO reads (from transmit shift registers) and writes (by software).
25-24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are loaded from the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO write pointer will only increment when the full 32-bit FIFO word has been written by software. 00b - FIFO packing is disabled 01b - Reserved 10b - 8-bit FIFO packing is enabled 11b - 16-bit FIFO packing is enabled
23-21 —	Reserved
20-16 FRSZ	Frame size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words.
15-13 —	Reserved
12-8 SYWD	Sync Width

Table continues on the next page...

Field	Function
	Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 CHMOD	Channel Mode Configures if transmit data pins are configured for TDM mode or Output mode. 0b - TDM mode, transmit data pins are tri-stated when slots are masked or channels are disabled. 1b - Output mode, transmit data pins are never tri-stated and will output zero when slots are masked or channels are disabled.
4 MF	MSB First Configures whether the LSB or the MSB is transmitted first. 0b - LSB is transmitted first. 1b - MSB is transmitted first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame sync is generated externally in Slave mode. 1b - Frame sync is generated internally in Master mode.

44.4.2.9 SAI Transmit Configuration 5 Register (TCR5)

44.4.2.9.1 Offset

Register	Offset
TCR5	1Ch

44.4.2.9.2 Function

This register must not be altered when TCSR[TE] is set.

44.4.2.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WNW					0			W0W				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FBT					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.9.4 Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13 —	Reserved
12-8 FBT	First Bit Shifted Configures the bit index for the first bit transmitted for each word in the frame. If configured for MSB First, the index of the next bit transmitted is one less than the current bit transmitted. If configured for LSB First, the index of the next bit transmitted is one more than the current bit transmitted. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

44.4.2.10 SAI Transmit Data Register (TDR0 - TDR3)

44.4.2.10.1 Offset

Register	Offset
TDR0	20h
TDR1	24h
TDR2	28h
TDR3	2Ch

44.4.2.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.10.3 Fields

Field	Function
31-0 TDR	Transmit Data Register Writes to this register when the transmit FIFO is not full will push the data written into the transmit data FIFO. Writes to this register when the transmit FIFO is full are ignored.

44.4.2.11 SAI Transmit FIFO Register (TFR0 - TFR3)

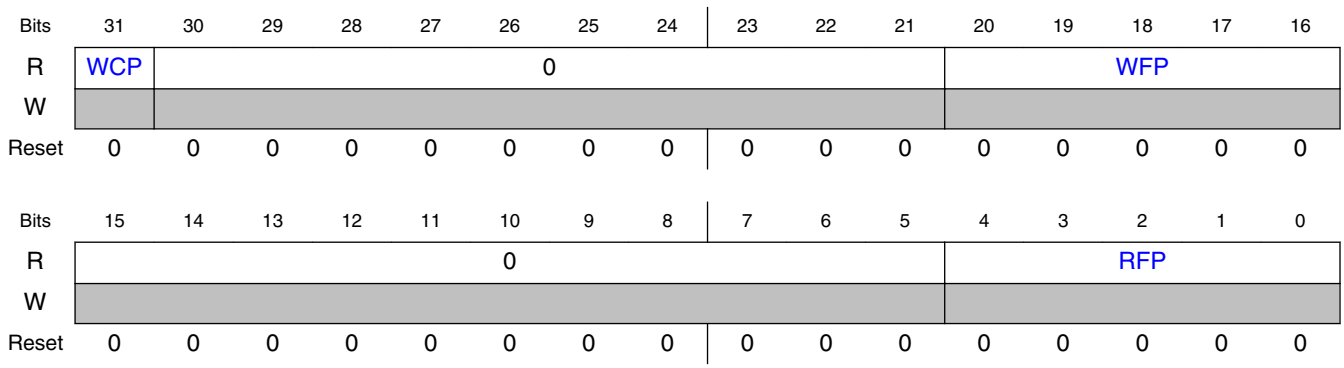
44.4.2.11.1 Offset

Register	Offset
TFR0	40h
TFR1	44h
TFR2	48h
TFR3	4Ch

44.4.2.11.2 Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

44.4.2.11.3 Diagram



44.4.2.11.4 Fields

Field	Function
31 WCP	Write Channel Pointer When FIFO Combine mode is enabled for writes, indicates that this data channel is the next FIFO to be written. 0b - No effect. 1b - FIFO combine is enabled for FIFO writes and this FIFO will be written on the next FIFO write.
30-21 —	Reserved
20-16 WFP	Write FIFO Pointer FIFO write pointer for transmit data channel.
15-5 —	Reserved
4-0 RFP	Read FIFO Pointer FIFO read pointer for transmit data channel.

44.4.2.12 SAI Transmit Mask Register (TMR)

44.4.2.12.1 Offset

Register	Offset
TMR	60h

44.4.2.12.2 Function

This register is double-buffered and updates:

1. When TCSR[TE] is first set
2. At the end of each frame.

This allows the masked words in each frame to change from frame to frame.

44.4.2.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.12.4 Fields

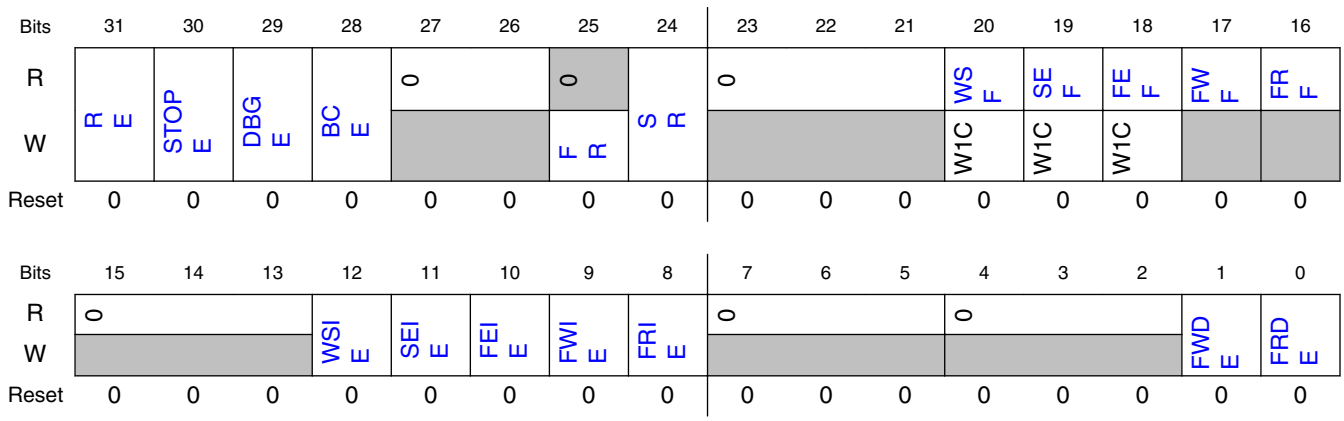
Field	Function
31-0 TWM	Transmit Word Mask Configures whether the transmit word is masked (transmit data pins are tri-stated or drive zero and transmit data not read from FIFO) for the corresponding word in the frame. 00000000000000000000000000000000b - Word N is enabled. 00000000000000000000000000000001b - Word N is masked. The transmit data pins are tri-stated or drive zero when masked.

44.4.2.13 SAI Receive Control Register (RCSR)

44.4.2.13.1 Offset

Register	Offset
RCSR	88h

44.4.2.13.2 Diagram



44.4.2.13.3 Fields

Field	Function
31 RE	Receiver Enable Enables/disables the receiver. When software clears this field, the receiver remains enabled, and this bit remains set, until the end of the current frame. 0b - Receiver is disabled. 1b - Receiver is enabled, or receiver has been disabled and has not yet reached end of frame.
30 STOPE	Stop Enable Configures receiver operation in Stop mode. The Stop Enable field is ignored and the receiver is disabled in all low-leakage stop modes. 0b - Receiver disabled in Stop mode. 1b - Receiver enabled in Stop mode.
29 DBGE	Debug Enable Enables/disables receiver operation in Debug mode. The receive bit clock is not affected by Debug mode. 0b - Receiver is disabled in Debug mode, after completing the current frame. 1b - Receiver is enabled in Debug mode.
28 BCE	Bit Clock Enable Enables the receive bit clock, separately from RE. This field is automatically set whenever RE is set. When software clears this field, the receive bit clock remains enabled, and this field remains set, until the end of the current frame. 0b - Receive bit clock is disabled. 1b - Receive bit clock is enabled.
27-26	Reserved

Table continues on the next page...

Field	Function
—	
25 FR	FIFO Reset Empties the FIFO, and sets the FIFO read and write pointers to the same value, which may or may not be zero. Reading this field will always return zero. FIFO pointers should only be reset when the receiver is disabled or the FIFO error flag is set. 0b - No effect. 1b - FIFO reset.
24 SR	Software Reset Resets the internal receiver logic including the FIFO pointers. Software-visible registers are not affected, except for the status registers. 0b - No effect. 1b - Software reset.
23-21 —	Reserved
20 WSF	Word Start Flag Indicates that the start of the configured word has been detected. Write a logic 1 to this field to clear this flag. 0b - Start of word not detected. 1b - Start of word detected.
19 SEF	Sync Error Flag Indicates that an error in the externally-generated frame sync has been detected. Write a logic 1 to this field to clear this flag. 0b - Sync error not detected. 1b - Frame sync error detected.
18 FEF	FIFO Error Flag Indicates that an enabled receive FIFO has overflowed. Write a logic 1 to this field to clear this flag. 0b - Receive overflow not detected. 1b - Receive overflow detected.
17 FWF	FIFO Warning Flag Indicates that an enabled receive FIFO is full. 0b - No enabled receive FIFO is full. 1b - Enabled receive FIFO is full.
16 FRF	FIFO Request Flag Indicates that the number of words in an enabled receive channel FIFO is greater than the receive FIFO watermark. 0b - Receive FIFO watermark not reached. 1b - Receive FIFO watermark has been reached.
15-13 —	Reserved
12 WSIE	Word Start Interrupt Enable Enables/disables word start interrupts. 0b - Disables interrupt. 1b - Enables interrupt.
11 SEIE	Sync Error Interrupt Enable Enables/disables sync error interrupts. 0b - Disables interrupt. 1b - Enables interrupt.

Table continues on the next page...

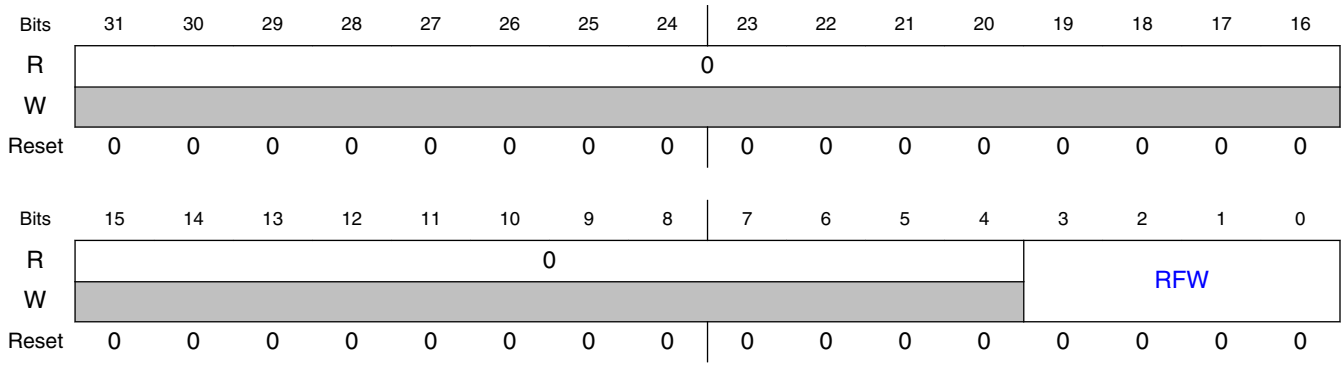
Field	Function
10 FEIE	FIFO Error Interrupt Enable Enables/disables FIFO error interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
9 FWIE	FIFO Warning Interrupt Enable Enables/disables FIFO warning interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
8 FRIE	FIFO Request Interrupt Enable Enables/disables FIFO request interrupts. 0b - Disables the interrupt. 1b - Enables the interrupt.
7-5 —	Reserved
4-2 —	Reserved
1 FWDE	FIFO Warning DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.
0 FRDE	FIFO Request DMA Enable Enables/disables DMA requests. 0b - Disables the DMA request. 1b - Enables the DMA request.

44.4.2.14 SAI Receive Configuration 1 Register (RCR1)

44.4.2.14.1 Offset

Register	Offset
RCR1	8Ch

44.4.2.14.2 Diagram



44.4.2.14.3 Fields

Field	Function
31-4 —	Reserved
3-0 RFW	Receive FIFO Watermark Configures the watermark level for all enabled receiver channels.

44.4.2.15 SAI Receive Configuration 2 Register (RCR2)

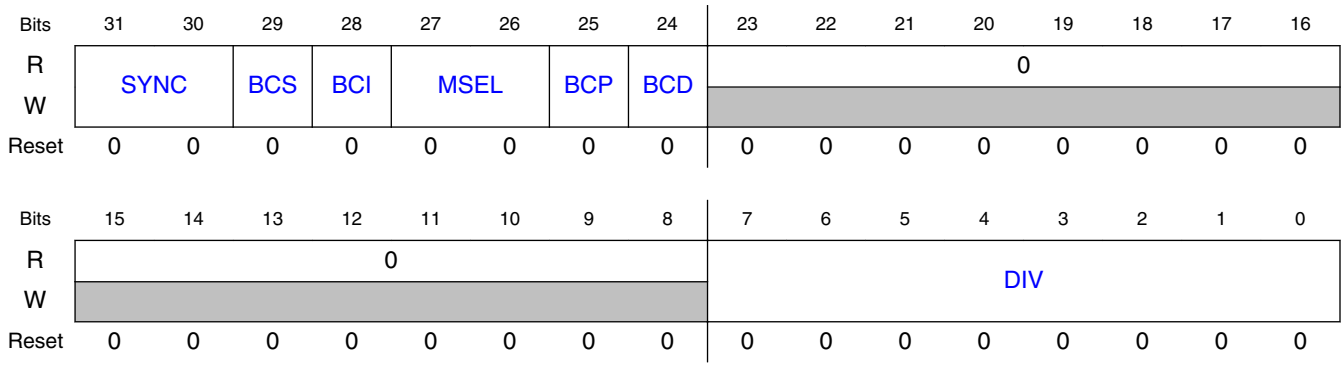
44.4.2.15.1 Offset

Register	Offset
RCR2	90h

44.4.2.15.2 Function

This register must not be altered when RCSR[RE] is set.

44.4.2.15.3 Diagram



44.4.2.15.4 Fields

Field	Function
31-30 SYNC	Synchronous Mode Configures between asynchronous and synchronous modes of operation. When configured for a synchronous mode of operation, the transmitter or other SAI peripheral must be configured for asynchronous operation. 00b - Asynchronous mode. 01b - Synchronous with transmitter. 10b - Synchronous with another SAI receiver. 11b - Synchronous with another SAI transmitter.
29 BCS	Bit Clock Swap This field swaps the bit clock used by the receiver. When the receiver is configured in asynchronous mode and this bit is set, the receiver is clocked by the transmitter bit clock (TX_BCLK). This allows the transmitter and receiver to share the same bit clock, but the receiver continues to use the receiver frame sync (RX_SYNC). When the receiver is configured in synchronous mode, the transmitter BCS field and receiver BCS field must be set to the same value. When both are set, the transmitter and receiver are both clocked by the receiver bit clock (RX_BCLK) but use the transmitter frame sync (TX_SYNC). This field has no effect when synchronous to another SAI peripheral. 0b - Use the normal bit clock source. 1b - Swap the bit clock source.
28 BCI	Bit Clock Input When this field is set and using an internally generated bit clock in either synchronous or asynchronous mode, the bit clock actually used by the receiver is delayed by the pad output delay (the receiver is clocked by the pad input as if the clock was externally generated). This has the effect of decreasing the data input setup time, but increasing the data output valid time. The slave mode timing from the datasheet should be used for the receiver when this bit is set. In synchronous mode, this bit allows the receiver to use the slave mode timing from the datasheet, while the transmitter uses the master mode timing. This field has no effect when configured for an externally generated bit clock or when synchronous to another SAI peripheral . 0b - No effect. 1b - Internal logic is clocked as if bit clock was externally generated.
27-26	MCLK Select

Table continues on the next page...

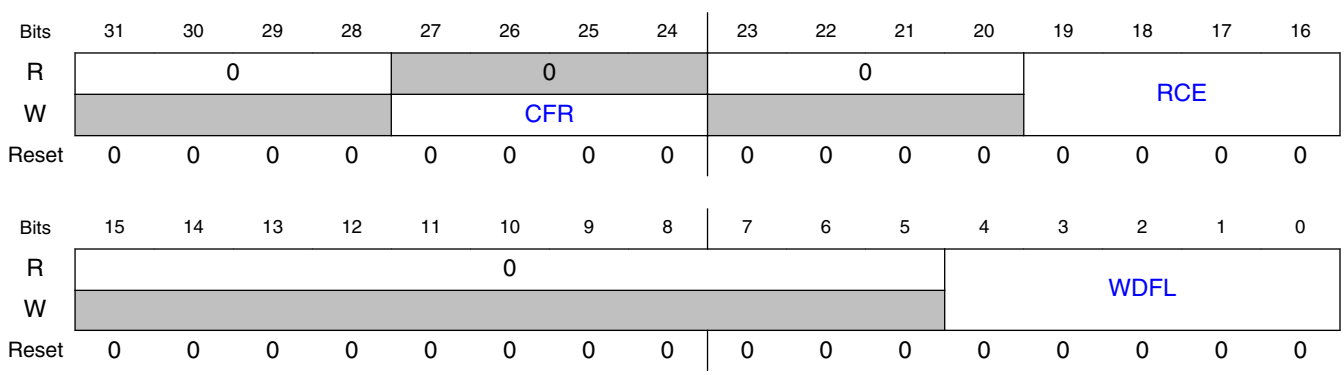
Field	Function
MSEL	Selects the audio Master Clock option used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. **NOTE:** Depending on the device, some Master Clock options might not be available. See the chip-specific information for the availability and chip-specific meaning of each option. 00b - Bus Clock selected. 01b - Master Clock (MCLK) 1 option selected. 10b - Master Clock (MCLK) 2 option selected. 11b - Master Clock (MCLK) 3 option selected.
25 BCP	Bit Clock Polarity Configures the polarity of the bit clock. 0b - Bit Clock is active high with drive outputs on rising edge and sample inputs on falling edge. 1b - Bit Clock is active low with drive outputs on falling edge and sample inputs on rising edge.
24 BCD	Bit Clock Direction Configures the direction of the bit clock. 0b - Bit clock is generated externally in Slave mode. 1b - Bit clock is generated internally in Master mode.
23-8 —	Reserved
7-0 DIV	Bit Clock Divide Divides down the audio master clock to generate the bit clock when configured for an internal bit clock. The division value is (DIV + 1) * 2.

44.4.2.16 SAI Receive Configuration 3 Register (RCR3)

44.4.2.16.1 Offset

Register	Offset
RCR3	94h

44.4.2.16.2 Diagram



44.4.2.16.3 Fields

Field	Function
31-28 —	Reserved
27-24 CFR	Channel FIFO Reset Resets the FIFO pointers for a specific channel. Reading this field will always return zero. FIFO pointers should only be reset when a channel is disabled or the FIFO error flag is set. The width of CFR field = the number of receive channels (call it N). For example, if CFR is 2 bits wide, then bit position 24 refers to receive channel 1 FIFO pointer and bit position 25 refers to receive channel 2 FIFO pointer. Setting bit 24 resets receive channel 1 FIFO pointer, and setting bit 25 enables receive channel 2 FIFO pointer. Setting bit N will reset receive channel N FIFO pointer. 0b - No effect. 1b - Receive data channel N FIFO is reset.
23-20 —	Reserved
19-16 RCE	Receive Channel Enable Enables the corresponding data channel for receive operation. Changing this field will take effect immediately for generating the FIFO request and warning flags, but at the end of each frame for receive operation. The width of RCE field = the number of receive channels (call it N). For example, if RCE field is 2 bits wide, then bit position 16 refers to receive channel 1 and bit position 17 refers to receive channel 2. Setting bit 16 enables receive channel 1, and setting bit 17 enables receive channel 2. Setting bit N will enable receive channel N. 0b - Receive data channel N is disabled. 1b - Receive data channel N is enabled.
15-5 —	Reserved
4-0 WDFL	Word Flag Configuration Configures which word the start of word flag is set. The value written should be one less than the word number (for example, write zero to configure for the first word in the frame). When configured to a value greater than the Frame Size field, then the start of word flag is never set.

44.4.2.17 SAI Receive Configuration 4 Register (RCR4)

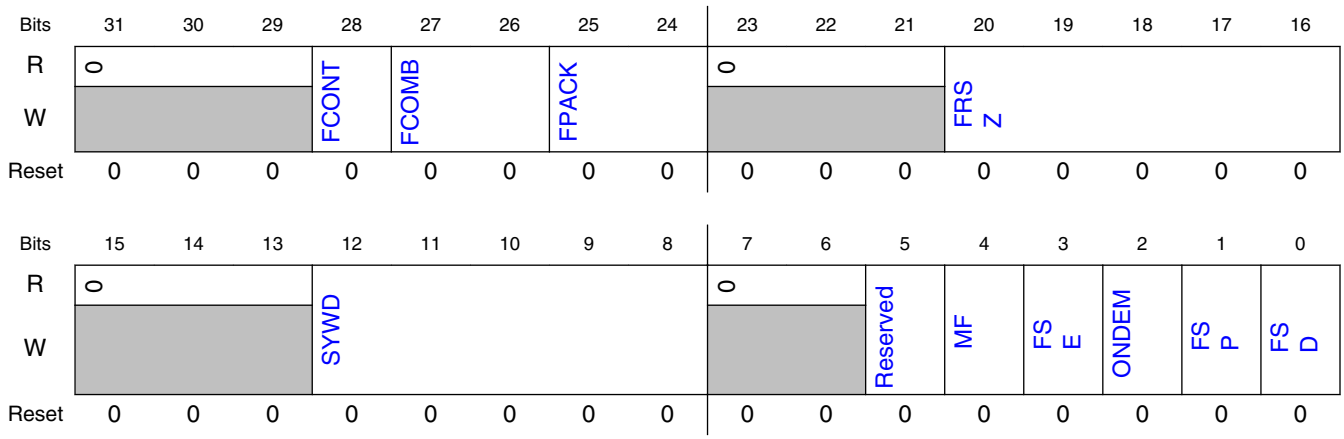
44.4.2.17.1 Offset

Register	Offset
RCR4	98h

44.4.2.17.2 Function

This register must not be altered when RCSR[RE] is set.

44.4.2.17.3 Diagram



44.4.2.17.4 Fields

Field	Function
31-29 —	Reserved
28 FCONT	FIFO Continue on Error Configures when the SAI will continue receiving after a FIFO error has been detected. 0b - On FIFO error, the SAI will continue from the start of the next frame after the FIFO error flag has been cleared. 1b - On FIFO error, the SAI will continue from the same word that caused the FIFO error to set after the FIFO warning flag has been cleared.
27-26 FCOMB	FIFO Combine Mode When FIFO combine mode is enabled for FIFO reads, software reading any FIFO data register will alternate the read among the enabled data channel FIFOs. For example, if two data channels are enabled then the first read will be performed to the first enabled data channel FIFO and the second read will be performed to the second enabled data channel FIFO. Reseting the FIFO or disabling FIFO combine mode for FIFO reads will reset the pointer back to the first enabled data channel. When FIFO combine mode is enabled for FIFO writes from the receive shift registers, the first enabled data channel input will alternate between the enabled data channel FIFOs. For example, if two data channels are enabled then the first unmasked received word will be stored in the first enabled data channel FIFO and the second unmasked received word will be stored in the second enabled data channel FIFO. Since the first word of the frame is always stored in the first enabled data channel FIFO, it is recommended that the number of unmasked words per frame is evenly divisible by the number of enabled data channels. 00b - FIFO combine mode disabled. 01b - FIFO combine mode enabled on FIFO writes (from receive shift registers). 10b - FIFO combine mode enabled on FIFO reads (by software). 11b - FIFO combine mode enabled on FIFO writes (from receive shift registers) and reads (by software).

Table continues on the next page...

Field	Function
25-24 FPACK	FIFO Packing Mode Enables packing of 8-bit data or 16-bit data into each 32-bit FIFO word. If the word size is greater than 8-bit or 16-bit then only the first 8-bit or 16-bits are stored to the FIFO. The first word in each frame always starts with a new 32-bit FIFO word and the first bit shifted must be configured within the first packed word. When FIFO packing is enabled, the FIFO read pointer will only increment when the full 32-bit FIFO word has been read by software. 00b - FIFO packing is disabled 01b - Reserved. 10b - 8-bit FIFO packing is enabled 11b - 16-bit FIFO packing is enabled
23-21 —	Reserved
20-16 FRSZ	Frame Size Configures the number of words in each frame. The value written must be one less than the number of words in the frame. For example, write 0 for one word per frame. The maximum supported frame size is 32 words.
15-13 —	Reserved
12-8 SYWD	Sync Width Configures the length of the frame sync in number of bit clocks. The value written must be one less than the number of bit clocks. For example, write 0 for the frame sync to assert for one bit clock only. The sync width cannot be configured longer than the first word of the frame.
7-6 —	Reserved
5 —	Reserved. Software should only write zero to this bit.
4 MF	MSB First Configures whether the LSB or the MSB is received first. 0b - LSB is received first. 1b - MSB is received first.
3 FSE	Frame Sync Early 0b - Frame sync asserts with the first bit of the frame. 1b - Frame sync asserts one bit before the first bit of the frame.
2 ONDEM	On Demand Mode When set, and the frame sync is generated internally, a frame sync is only generated when the FIFO warning flag is clear. 0b - Internal frame sync is generated continuously. 1b - Internal frame sync is generated when the FIFO warning flag is clear.
1 FSP	Frame Sync Polarity Configures the polarity of the frame sync. 0b - Frame sync is active high. 1b - Frame sync is active low.
0 FSD	Frame Sync Direction Configures the direction of the frame sync. 0b - Frame Sync is generated externally in Slave mode. 1b - Frame Sync is generated internally in Master mode.

44.4.2.18 SAI Receive Configuration 5 Register (RCR5)

44.4.2.18.1 Offset

Register	Offset
RCR5	9Ch

44.4.2.18.2 Function

This register must not be altered when RCSR[RE] is set.

44.4.2.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			WNW					0			W0W				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			FBT					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.18.4 Fields

Field	Function
31-29 —	Reserved
28-24 WNW	Word N Width Configures the number of bits in each word, for each word except the first in the frame. The value written must be one less than the number of bits per word. Word width of less than 8 bits is not supported.
23-21 —	Reserved
20-16 W0W	Word 0 Width Configures the number of bits in the first word in each frame. The value written must be one less than the number of bits in the first word. Word width of less than 8 bits is not supported if there is only one word per frame.
15-13	Reserved

Table continues on the next page...

Field	Function
—	
12-8 FBT	First Bit Shifted Configures the bit index for the first bit received for each word in the frame. If configured for MSB First, the index of the next bit received is one less than the current bit received. If configured for LSB First, the index of the next bit received is one more than the current bit received. The value written must be greater than or equal to the word width when configured for MSB First. The value written must be less than or equal to 31-word width when configured for LSB First.
7-0 —	Reserved

44.4.2.19 SAI Receive Data Register (RDR0 - RDR3)

44.4.2.19.1 Offset

Register	Offset
RDR0	A0h
RDR1	A4h
RDR2	A8h
RDR3	ACh

44.4.2.19.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.19.3 Fields

Field	Function
31-0 RDR	Receive Data Register

Field	Function
	Reads from this register when the receive FIFO is not empty will return the data from the top of the receive FIFO. Reads from this register when the receive FIFO is empty are ignored.

44.4.2.20 SAI Receive FIFO Register (RFR0 - RFR3)

44.4.2.20.1 Offset

Register	Offset
RFR0	C0h
RFR1	C4h
RFR2	C8h
RFR3	CCh

44.4.2.20.2 Function

The MSB of the read and write pointers is used to distinguish between FIFO full and empty conditions. If the read and write pointers are identical, then the FIFO is empty. If the read and write pointers are identical except for the MSB, then the FIFO is full.

44.4.2.20.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											WFP				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCP	0										RFP				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.20.4 Fields

Field	Function
31-21 —	Reserved

Table continues on the next page...

Field	Function
20-16 WFP	Write FIFO Pointer FIFO write pointer for receive data channel.
15 RCP	Receive Channel Pointer When FIFO Combine mode is enabled for reads, indicates that this data channel is the next FIFO to be read. 0b - No effect. 1b - FIFO combine is enabled for FIFO reads and this FIFO will be read on the next FIFO read.
14-5 —	Reserved
4-0 RFP	Read FIFO Pointer FIFO read pointer for receive data channel.

44.4.2.21 SAI Receive Mask Register (RMR)

44.4.2.21.1 Offset

Register	Offset
RMR	E0h

44.4.2.21.2 Function

This register is double-buffered and updates:

1. When RCSR[RE] is first set
2. At the end of each frame

This allows the masked words in each frame to change from frame to frame.

44.4.2.21.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RWM															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

44.4.2.21.4 Fields

Field	Function
31-0 RWM	Receive Word Mask Configures whether the receive word is masked (received data ignored and not written to receive FIFO) for the corresponding word in the frame. 00000000000000000000000000000000b - Word N is enabled. 00000000000000000000000000000001b - Word N is masked.

44.5 Functional description

This section provides a complete functional description of the block.

44.5.1 SAI clocking

The SAI clocks include:

- The audio master clock
- The bit clock
- The bus clock

44.5.1.1 Audio master clock

The audio master clock is used to generate the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The transmitter and receiver can independently select between the bus clock and up to three audio master clocks to generate the bit clock.

The audio master clock generation and selection is chip-specific. Refer to chip-specific clocking information about how the audio master clocks are generated.

44.5.1.2 Bit clock

The SAI transmitter and receiver support asynchronous free-running bit clocks that can be generated internally from an audio master clock or supplied externally. There is also the option for synchronous bit clock and frame sync operation between the receiver and transmitter or between multiple SAI peripherals.

- If both transmitter and receiver are configured for asynchronous operation, then the transmitter and receiver will each use *their own* bit clock and frame sync.
- If the *transmitter* is configured for asynchronous mode and the receiver is configured for synchronous mode, then both transmitter and receiver will use the *transmitter* bit clock and frame sync.
- If the *receiver* is configured for asynchronous mode and the transmitter is configured for synchronous mode, then both transmitter and receiver will use the *receiver* bit clock and frame sync.

Note that the software configures synchronous or asynchronous mode, and that choice selects the bit clock/frame sync used.

Externally generated bit clocks must be:

- Enabled before the SAI transmitter or receiver is enabled
- Disabled after the SAI transmitter or receiver is disabled and completes its current frames

If the SAI transmitter or receiver is using an externally generated bit clock in asynchronous mode and that bit clock is generated by an SAI that is disabled in stop mode, then the transmitter or receiver should be disabled by software before entering stop mode. This issue does not apply when the transmitter or receiver is in a synchronous mode because all synchronous SAIs are enabled and disabled simultaneously.

44.5.1.3 Bus clock

The bus clock is used by the control and configuration registers and to generate synchronous interrupts and DMA requests.

NOTE

Although there is no specific minimum bus clock frequency specified, the bus clock frequency must be fast enough (relative to the bit clock frequency) to ensure that the FIFOs can be serviced, without generating either a transmitter FIFO underrun or receiver FIFO overflow condition.

44.5.2 SAI resets

The SAI is asynchronously reset on system reset. The SAI has a software reset and a FIFO reset.

44.5.2.1 Software reset

The SAI transmitter includes a software reset that resets all transmitter internal logic, including the bit clock generation, status flags, and FIFO pointers. It does not reset the configuration registers. The software reset remains asserted until cleared by software.

The SAI receiver includes a software reset that resets all receiver internal logic, including the bit clock generation, status flags and FIFO pointers. It does not reset the configuration registers. The software reset remains asserted until cleared by software.

44.5.2.2 FIFO reset

The SAI transmitter includes a FIFO reset that synchronizes the FIFO write pointer to the same value as the FIFO read pointer. This empties the FIFO contents and is to be used after TCSR[FEF] is set, and before the FIFO is re-initialized and TCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

The SAI transmitter can also reset the FIFO of individual data channels by setting the appropriate TCR3[CFR] bit. This should only be done when the corresponding TCR3[TCE] bit is clear.

The SAI receiver includes a FIFO reset that synchronizes the FIFO read pointer to the same value as the FIFO write pointer. This empties the FIFO contents and is to be used after the RCSR[FEF] is set and any remaining data has been read from the FIFO, and before the RCSR[FEF] is cleared. The FIFO reset is asserted for one cycle only.

The SAI receiver can also reset the FIFO of individual data channels by setting the appropriate RCR3[CFR] bit. This should only be done when the corresponding RCR3[RCE] bit is clear.

44.5.3 Synchronous modes

The SAI transmitter and receiver can operate synchronously to each other or synchronously to other SAI peripherals.

44.5.3.1 Synchronous mode

The SAI transmitter and receiver can be configured to operate with synchronous bit clock and frame sync.

If the transmitter bit clock and frame sync are to be used by both the transmitter and receiver:

- The transmitter must be configured for asynchronous operation and the receiver for synchronous operation.
- In synchronous mode, the receiver is enabled only when both the transmitter and receiver are enabled.
- It is recommended that the transmitter is the last enabled and the first disabled.

If the receiver bit clock and frame sync are to be used by both the transmitter and receiver:

- The receiver must be configured for asynchronous operation and the transmitter for synchronous operation.
- In synchronous mode, the transmitter is enabled only when both the receiver and transmitter are both enabled.
- It is recommended that the receiver is the last enabled and the first disabled.

When operating in synchronous mode, only the bit clock, frame sync, and transmitter/receiver enable are shared. The transmitter and receiver otherwise operate independently, although configuration registers must be configured consistently across both the transmitter and receiver.

44.5.3.2 Multiple SAI Synchronous mode

Synchronous operation between multiple SAI peripherals is not supported on all devices. This mode requires the source of the bit clock and frame sync to be configured for asynchronous operation and the remaining users of the bit clock and frame sync to be configured for synchronous operation.

Synchronous operation between multiple SAI transmitters or receivers also requires the source of the bit clock and frame sync to be enabled for any of the synchronous transmitters or receivers to also be enabled. It is recommended that the source of the bit clock and frame sync is the last enabled and the first disabled.

When operating in synchronous mode, only the bit clock, frame sync, and transmitter/receiver enable are shared. The separate SAI peripherals otherwise operate independently, although configuration registers must be configured consistently across both the transmitter and receiver.

44.5.4 Frame sync configuration

When enabled, the SAI continuously transmits and/or receives frames of data. Each frame consists of a fixed number of words and each word consists of a fixed number of bits. Within each frame, any given word can be masked causing the receiver to ignore that word and the transmitter to tri-state for the duration of that word.

The frame sync signal is used to indicate the start of each frame. A valid frame sync requires a rising edge (if active high) or falling edge (if active low) to be detected and the transmitter or receiver cannot be busy with a previous frame. A valid frame sync is also ignored (slave mode) or not generated (master mode) for the first four bit clock cycles after enabling the transmitter or receiver.

The transmitter and receiver frame sync can be configured independently with any of the following options:

- Externally generated or internally generated
- Active high or active low
- Assert with the first bit in frame or asserts one bit early
- Assert for a duration between 1 bit clock and the first word length
- Frame length from 1 to 32 words per frame
- Word length to support 8 to 32 bits per word
    - First word length and remaining word lengths can be configured separately
- Words can be configured to transmit/receive MSB first or LSB first

These configuration options cannot be changed after the SAI transmitter or receiver is enabled.

44.5.5 Data FIFO

Each transmit and receive channel includes a FIFO of size 16 x 32-bit. The FIFO data is accessed using the SAI Transmit/Receive Data Registers.

44.5.5.1 Data alignment

Data in the FIFO can be aligned anywhere within the 32-bit wide register through the use of the First Bit Shifted configuration field, which selects the bit index (between 31 and 0) of the first bit shifted.

Examples of supported data alignment and the required First Bit Shifted configuration are illustrated in Figure 44-2 for LSB First configurations and Figure 44-3 for MSB First configurations.

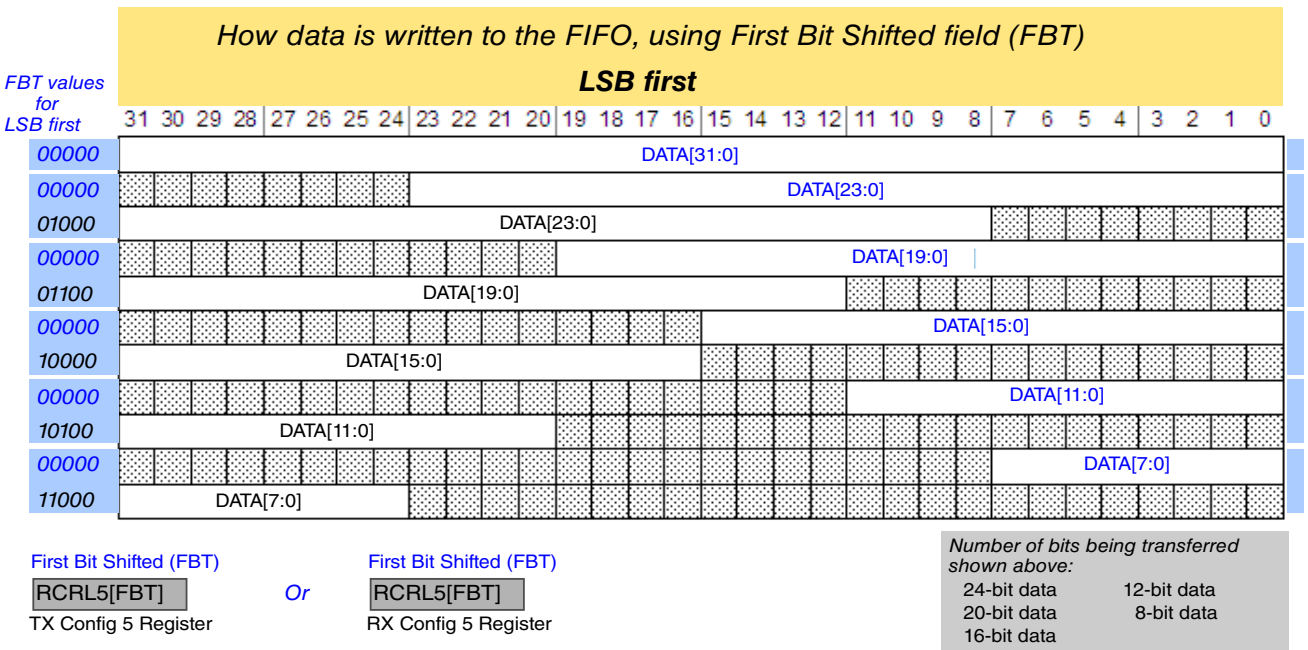


Figure 44-2. SAI first bit shifted, LSB first

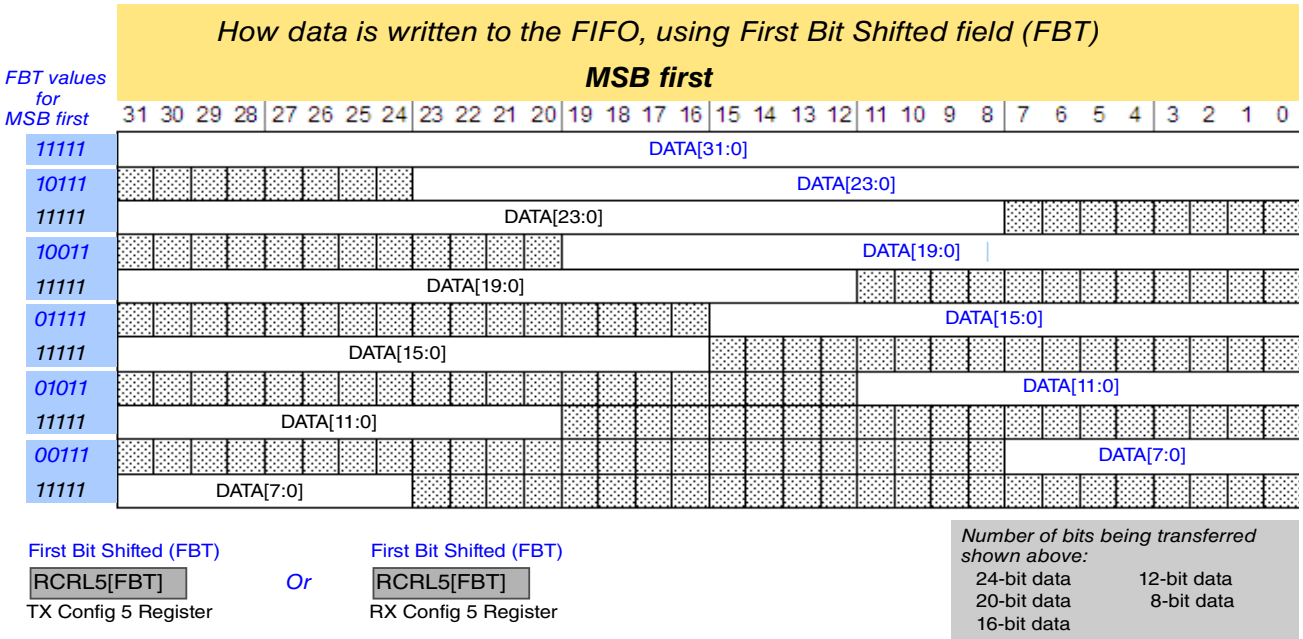


Figure 44-3. SAI first bit shifted, MSB first

44.5.5.2 FIFO pointers

When writing to a Transmit Data Register (TDR*n*), the Write FIFO Pointer (WFP) of the corresponding Transmit FIFO Register (TFR*n*) increments after each valid write. The SAI supports 8-bit, 16-bit and 32-bit writes to the Transmit Data Register and the FIFO pointer will increment after each individual write. Note that 8-bit writes should only be used when transmitting up to 8-bit data; 16-bit writes should only be used when transmitting up to 16-bit data.

- If the Transmit FIFO is full, then writes to a Transmit Data Register are ignored.
- If the Transmit FIFO is empty, then to avoid a FIFO underrun, the Transmit Data Register must be written at least 3 bit clocks before the start of the next unmasked word. Before enabling the transmitter, the Transmit FIFO should be initialized with data (since after the transmitter is enabled, the transmitter will start a new frame, and if no data is in the FIFO, then the transmitter will immediately give an error).

When reading a Receive Data Register (RDR*n*), the Read FIFO Pointer (RFP) of the corresponding Receive FIFO Register (RFR*n*) increments after each valid read. The SAI supports 8-bit, 16-bit and 32-bit reads from the RDR and the FIFO pointer will increment after each individual read. Note that 8-bit reads should only be used when receiving up to 8-bit data; 16-bit reads should only be used when receiving up to 16-bit data.

- If the Receive FIFO is empty, then reads from a Receive Data Register are ignored.
- If the Receive FIFO is full, then to avoid a FIFO overrun, the Receive Data Register must be read at least 3 bit clocks before the end of an unmasked word.

44.5.5.3 FIFO packing

FIFO packing supports storing multiple 8-bit or 16-bit data words in one 32-bit FIFO word for the transmitter and/or receiver. While this can be emulated by adjusting the number of bits per word and number of words per frame (for example, one 32-bit word per frame versus two 16-bit words per frame), FIFO packing does not require even multiples of words per frame and fully supports word masking. When FIFO packing is enabled, the FIFO pointers only increment when the full 32-bit FIFO word has been written (transmit) or read (receive) by software, supporting scenarios where different words within each frame are loaded/stored in different areas of memory.

When 16-bit FIFO packing is enabled for transmit, the transmit shift register is loaded at the start of each frame and after every second unmasked transmit word. The first word transmitted is taken from 16-bit word at byte offset $0 (first bit is selected by TCFG5[FBT] must be configured within this 16-bit word) and the second word transmitted is taken from the 16-bit word at byte offset $2 (first bit is selected by TCSR5[FBT][3:0]). The transmitter will transmit logic zero until the start of the next word once the 16-bit word has been transmitted.

When 16-bit FIFO packing is enabled for receive, the receive shift register is stored after every second unmasked received word, and at the end of each frame if there is an odd number of unmasked received words in each frame. The first word received is stored in the 16-bit word at byte offset $0 (first bit is selected by RCFG5[FBT] and must be configured within this 16-bit word) and the second word received is stored in the 16-bit word at byte offset $2 (first bit is selected by RCSR5[FBT][3:0]). The receiver will ignore received data until the start of the next word once the 16-bit word has been received.

The 8-bit FIFO packing is similar to 16-bit packing except four words are loaded or stored into each 32-bit FIFO word. The first word is loaded/stored in byte offset $0, second word in byte offset $1, third word in byte offset $2 and fourth word in byte offset $3. The TCFG5[FBT] and/or RCFG5[FBT] must be configured within byte offset $0.

44.5.5.4 FIFO Combine

FIFO combining mode allows the separate FIFOs for multiple data channels to be used as a single FIFO for either software accesses or a single data channel or both. Note that the enabled data channels must be contiguous and data channel 0 must be enabled when FIFO Combine mode is enabled.

Combining FIFOs for software access (writing transmit FIFO registers, reading receive FIFO registers) allows a DMA controller or software to read or write multiple FIFOs without incrementing the address that is accessed. Once enabled, the first software access to a FIFO register will access the first enabled channel FIFO, while the second access to a FIFO register will access the second enabled channel FIFO. This continues until software accesses the last enabled channel FIFO and the pointer resets back to the first enabled channel FIFO. To reset the pointer manually, software can reset the FIFOs or disable the FIFO combining on software accesses.

Combining FIFOs for transmit data channels allows one data channel to use the FIFOs of all enabled channel FIFOs, with identical data output on each enabled data channel. The transmit shift registers for all enabled data channels are loaded at the start of each frame and every N unmasked words (where N is the number of enabled data channels). The first word transmitted is loaded from the first enabled channel FIFO, while the second word transmitted is loaded from the second enabled channel FIFO, and so on until the end of the frame. Since the first word in each frame is always loaded from the first enabled data channel, it is recommended that the number of unmasked words in each frame is evenly divisible by the number of enabled data channels.

Combining FIFOs for receive data channels allows one data channel to use the FIFOs of all enabled channel FIFOs, with received data from channel 0 stored into each enabled data channel. The receive shift register for all enabled data channels are stored after every N unmasked words (where N is the number of enabled data channels). The first word received is stored to the first enabled channel FIFO, while the second word received is stored to the second enabled channel FIFO, and so on until the end of the frame. Since the first word in each frame is always stored the first enabled data channel, it is recommended that the number of unmasked words in each frame is evenly divisible by the number of enabled data channels.

Note that combining FIFOs for data channels will load or store each channel FIFO at the same time. This means that FIFO error conditions are only checked every N words (where N is the number of enabled data channels) and that the FIFO warning and request flags will assert if any of the enabled data channel meets the warning flag or request flag conditions.

44.5.6 Word mask register

The SAI transmitter and receiver each contain a word mask register, namely TMR and RMR, that can be used to mask any word in the frame. Because the word mask register is double buffered, software can update it before the end of each frame to mask a particular word in the next frame.

The TMR causes the Transmit Data pin to be tri-stated for the length of each selected word and the transmit FIFO is not read for masked words.

The RMR causes the received data for each selected word to be discarded and not written to the receive FIFO.

44.5.7 Interrupts and DMA requests

The SAI transmitter and receiver generate separate interrupts and separate DMA requests, but support the same status flags. Asynchronous versions of the transmitter and receiver interrupts are generated to wake up the CPU from stop mode.

44.5.7.1 FIFO request flag

The FIFO request flag is set based on the number of entries in the FIFO and the FIFO watermark configuration.

The transmit FIFO request flag is set when the number of entries in any of the enabled transmit FIFOs is less than or equal to the transmit FIFO watermark configuration and is cleared when the number of entries in each enabled transmit FIFO is greater than the transmit FIFO watermark configuration.

The receive FIFO request flag is set when the number of entries in any of the enabled receive FIFOs is greater than the receive FIFO watermark configuration and is cleared when the number of entries in each enabled receive FIFO is less than or equal to the receive FIFO watermark configuration.

The FIFO request flag can generate an interrupt or a DMA request.

44.5.7.2 FIFO warning flag

The FIFO warning flag is set based on the number of entries in the FIFO.

The transmit warning flag is set when the number of entries in any of the enabled transmit FIFOs is empty and is cleared when the number of entries in each enabled transmit FIFO is not empty.

The receive warning flag is set when the number of entries in any of the enabled receive FIFOs is full and is cleared when the number of entries in each enabled receive FIFO is not full.

The FIFO warning flag can generate an Interrupt or a DMA request.

44.5.7.3 FIFO error flag

The transmit FIFO error flag is set when the any of the enabled transmit FIFOs underflow. After it is set, all enabled transmit channels will transmit zero data before TCSR[FEF] is cleared.

When TCR4[FCONT] is set, the FIFO will continue transmitting data following an underflow without software intervention. To ensure that data is transmitted in the correct order, the transmitter will continue from the same word number in the frame that caused the FIFO to underflow, but only after new data has been written to the transmit FIFO. Software should still clear the TCSR[FEF] flag, but without reinitializing the transmit FIFOs.

RCSR[FEF] is set when the any of the enabled receive FIFOs overflow. After it is set, all enabled receive channels discard received data until RCSR[FEF] is cleared and the next next receive frame starts. All enabled receive FIFOs should be emptied before RCSR[FEF] is cleared.

When RCR4[FCONT] is set, the FIFO will continue receiving data following an overflow without software intervention. To ensure that data is received in the correct order, the receiver will continue from the same word number in the frame that caused the FIFO to overflow, but only after data has been read from the receive FIFO. Software should still clear the RCSR[FEF] flag, but without emptying the receive FIFOs.

The FIFO error flag can generate only an interrupt.

44.5.7.4 Sync error flag

The sync error flag, TCSR[SEF] or RCSR[SEF], is set when configured for an externally generated frame sync and the external frame sync asserts when the transmitter or receiver is busy with the previous frame. The external frame sync assertion is ignored and the sync error flag is set. When the sync error flag is set, the transmitter or receiver continues checking for frame sync assertion when idle or at the end of each frame.

The sync error flag can generate an interrupt only.

44.5.7.5 Word start flag

The word start flag is set at the start of the second bit clock for the selected word, as configured by the Word Flag register field.

The word start flag can generate an interrupt only.

Chapter 45
MIPI-DSI Controller

45.1 Chip-specific MIPI-DSI information

Table 45-1. Reference links to related information

Topic	Related module	Reference
Full description	MIPI-DSI	MIPI-DSI
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

45.1.1 Display serial interface (DSI) controller

The MIPI Display Serial Interface Controller (DSI Controller) is responsible for serializing display data.

Table 45-2. DSI Controller configuration

Parameter	Description
Name	DSI Controller
Supported standard version	MIPI DSI Specification v1.2r06, MIPI DCS Specification v1.2a, MIPI DPI Specification v2.00
Instances	1
Configurable features	• 2 data lanes, 1 clock lane • DPI interface to device
Interface speed	See the i.MX 7ULP Data Sheet (Document number: IMX7ULPCEC) for details.
External I/O pins	NA

45.1.2 SOPT1CFG register functionality in MIPI

The SIM_SOPT1CFG register allows access to the reset ports of the DSI Host Packet Interface (reset_esc, reset_byte), part of the DPI interface signals (such as reset_dpi, dpi_sd and dpi_cm) and also the DSI D-PHY PLL enable input port.

NOTE

The DPI-2 Interface is the DOTCLK (or RGB) interface mentioned in the LCDIF chapter.

45.1.3 Display Serial Interface PHY (DPHY)

The MIPI Display Serial Interface Physical Layer (DPHY) is a two-lane interface that supports up to 800 Mbps (precisely 792 Mbps or 396 MHz with 24 MHz crystal reference input) of data on each lane.

Table 45-3. DPHY configuration

Parameter	Description
Name	DPHY
Supported standard version	MIPI DPHY specification version 1.1
Instances	1
Configurable features	1 clock lane, 2 data lanes.
Interface speed	800 Mbps (precisely 792 Mbps or 396 MHz with 24 MHz crystal reference input)
External I/O pins	DSI_CLK_P, DSI_CLK_N, DSI_DATA0_P, DSI_DATA0_N, DSI_DATA1_P, DSI_DATA1_N. See the attached IOMUXC spreadsheet for pinout details

The MIPI DPHY generates the DSI byte clock (clk_byte) via the DPHY PLL. For details, see Figure 24-14

45.1.4 Features not supported on i.MX 7ULP

On this device, the DSI works only on MASTER mode (as a DSI HOST) and DSI Peripheral mode is not supported. Other features which are not supported on this device:

- DBI-2 interface (DBI-2/IF) is not supported including MPU, VSYNC, and Digital Video interface. Support to generic commands (short/long commands) as well as to DCS Commands will be handled through the Packet I/F via an APB2PacketIF Gasket.

- Status configuration via pins: Any status flags will not be pinned out. Any status shall be checked via DSI registers.
- DMA is also not supported. There is no requirement to use system DMA for display data transfer via DSI. The reason is that the LCDIF module serves as a bridge between the DSI controller and the system bus. As such, the DMA engine in the LCDIF will be the one used to transfer display data, instead. Moreover, the APB interface has FIFOs for the Packet Interface. This should provide sufficient storage to have the short/long commands for programming as well as receiving them over the slow APB interface.

45.1.5 DSI clock scheme

The DSI module along with the DPHY comprises 4 clock domains.

- DPI domain: logic in this domain is clocked by the pixel clock (*dpi_clk*) from the LCDIF module. This clock runs off a maximum frequency of 200 MHz, which is enough to support 1024x768 resolution. The DPI Bridge handles transferring video data received in the *dpi_clk* clock domain over to the DPHY domain (clk_byte)
- APB domain: logic in this domain is clocked by the APB clock (*pclk*). The Status & Control Register interface is at this domain.
- Escape Mode domain: logic in this domain is clocked by the Escape Mode logic, which includes the Low-Power Data Transmit (LPDT) state machines and low-power transmissions (*TxClkEsc*) as well as DPHY for reverse low-power reception (*RxClkEsc*). These clocks are synchronous between each other, while asynchronous with any other clocks. On i.MX 7ULP, these clocks are limited to the following range:
  - 60 MHz ≤ RxClkEsc ≤ 80 MHz
  - 15 MHz ≤ TxClkEsc ≤ 20 MHz
- DPHY domain: logic in this domain is clocked by the internal DPHY's PLL, which uses an external clock as the reference for the PLL (CLKREF). This clock is used by the DPHY to generate the High Speed MIPI clock and High Speed Data Lane signaling. This clock is also used to generate the MIPI Tx DPHY High Speed PPI interface (clk_byte), which is internal to the DSI TOP. This is a 1/8th the data rate of the DPHY Data Lane High Speed data rate

See Figure 45-3 for details.

45.1.6 DSI reset domains

The DSI has the following four reset domains:

- clk_byte reset domain (reset_byte_n): This is an active-low asynchronous reset that applies to all logic in the clk_byte clock domain.
- Escape reset domain (reset_esc_n): this is an active-low asynchronous reset that applies to all logic in the Escape Mode clock domain (TxClkEsc and RxClkEsc domains).
- APB reset domain (pclk_reset_n): this is an active-low asynchronous reset that applies to all logic in the APB clock domain (pclk)
- DPI reset domain (reset_dpi_n): this is an active-low asynchronous reset that applies to all logic in the DPI clock domain (dpi_pclk).

45.2 Overview

The Mobile Industry Processor Interface (MIPI) Display Serial Interface (DSI) controller is a flexible, high-performance, and easy-to-use digital core that implements all protocol functions defined in the MIPI DSI Specification. The MIPI DSI controller provides an interface that allows communication with MIPI DSI-compliant peripherals.

The MIPI DSI D-PHY is a high frequency, low power, low-cost, source-synchronous, physical layer supporting the MIPI Alliance standard for D-PHY.

45.3 Features

Key features of the MIPI DSI Controller Core include:

- Implements all three DSI Layers (Pixel to Byte packing, Low Level Protocol, Lane Management)
- Support for Command and Video Modes
- Host Version
- Scalable data lane support, 1 to 2 Data Lanes
  - Optional bidirectional support on lane 0
- Supports High Speed and Low Power operation
- Support for all DSI data types and formats
- Virtual Channel support
- Full Low-Level Protocol Error and Contention detection and reporting
- Supports continuous and non-continuous Clock Lane operation
- Supports multiple packets per transmission
- Support for all three Video Mode packet sequences
  - Non-Burst Mode with Sync Pulses

    - Non-Burst Mode with Sync Events
    - Burst mode
- Support for bus turnaround signaling
- Flexible packet based user interface
    - APB interface option (status and control)
    - Display Pixel Interface Core (DPI-2) option
- Supports PHY Protocol Interface (PPI) compatible MIPI D-PHYs
- MIPI Alliance Specification for Display Serial Interface Version 1.1 compliant

The following are the key features of the MIPI DSI D-PHY:

- 1 Clock lane
- 2 data lanes
- Supports MIPI Standard for D-PHY
- Supports both high speed and low power modes
- High Speed Serializers and Deserializers

45.4 DSI Host Controller Core

The figure below illustrates the DSI Host Controller Core structure. The DSI Host Controller Core operates on the host (transmit) side of a DSI link.

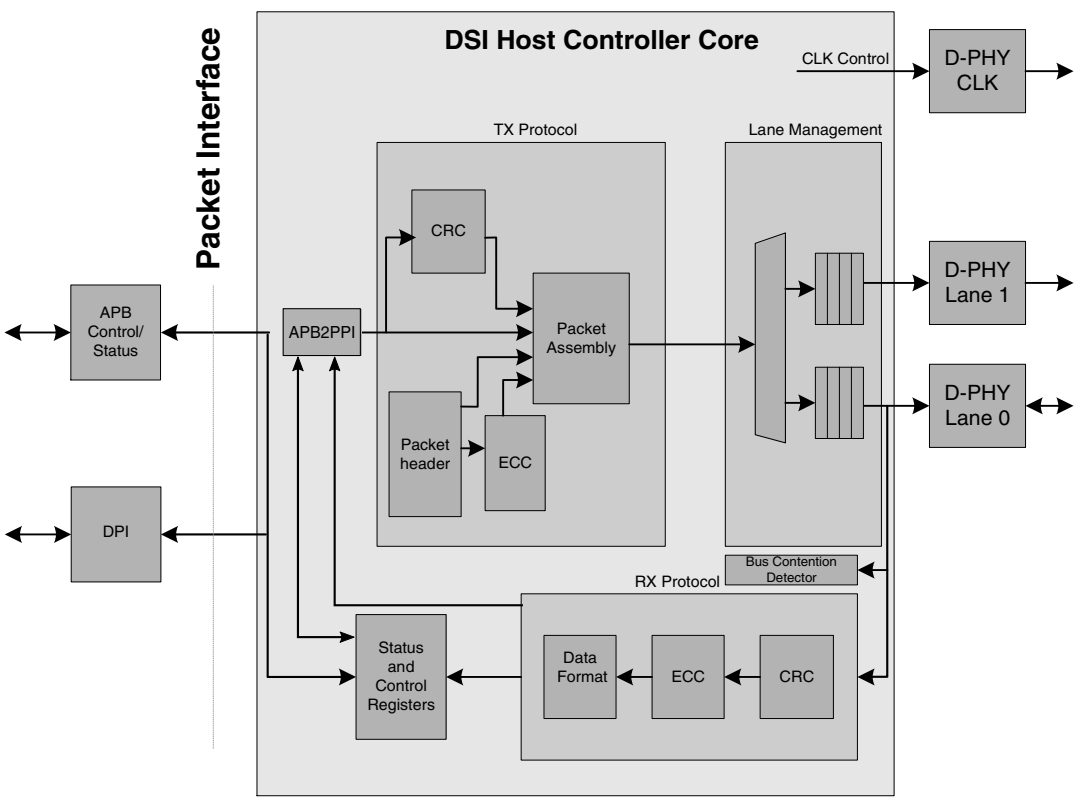


Figure 45-1. DSI Host Controller Core Block Diagram

The DSI Host Controller Core implements all three layers defined by the DSI Specification: Pixel to Byte Packing in the Application layer, Low Level Protocol, and Lane Management. The DSI Host Controller Core sends and receives DSI commands via the Packet Interface. The Packet Interface can be connected to a DPI translator or to an APB to PPI gasket.

The D-PHY interface of the DSI Host Controller Core supports up to four PHY Protocol Interface (PPI) compatible MIPI D-PHYs.

The Packet Interface is an easy-to-use data interface that accepts commands and data, and sends it over the DSI link. It supports 1 to 4 virtual channels, and the use of 1-2 D-PHY lanes. The DSI Host Controller Core takes care of all packet formatting details and transmission over the MIPI bus.

The DPI Translator connects to the DSI Host Controller Core via the Packet Interface. DPI masters may connect directly to the DPI Translator to send commands across the DSI link.

Sending commands and receiving information through the DSI link is also possible through the APB to PPI gasket. See MIPI DSI Host APB PKT IF Memory Map/Register section for details about the DSI registers which allow the APB to PPI setup and communication.

45.4.1 D-PHY Interface

The DSI Host Controller Core D-PHY Interface connects directly to MIPI PPI compliant D-PHYs. The DSI Host Controller Core uses this interface to transmit data, send Escape sequences, receive and transmit triggers, and detect and report D-PHY error conditions.

45.4.1.1 High-Speed Transmit Interface

The High-Speed Transmit interface is used to transmit High-Speed data across the MIPI interface. All signals are synchronous to the PPI TX byte clock, TxByteClkHS, and are PPI compliant.

45.4.1.2 Escape Mode Transmit Interface

The DSI Host Controller Core uses the Escape Mode Transmit Interface to generate Escape sequences on the MIPI Interface via the D-PHY. Escape sequences are used to change the DPHY transmit mode, go into low-power mode, or initiate a bus turnaround. The Controller generates Escape sequence requests when necessary to implement the DSI protocol. These signals are synchronous to the clk_esc.

45.4.2 Packet Interface

The DSI Host Controller Core Packet Interface consists of Transmit, Receive, and Control and Status sections. Through these interfaces, the user application can take complete control of the DSI interface, sending all video timing, sending DSI commands, receiving DSI reads, monitoring the status of the interface and responding to error reporting.

User application may have access to DSI Host Controller Core Packet Interface through DSI_HOST_APB_PKT_IF registers. For details refer to the section MIPI_DSI_HOST_APB_PKT_IF

45.4.2.1 Transmit Ports

The Transmit Packet Interface is the mechanism with which the user creates packets to send over the MIPI Interface.

For Long Packets, the user provides the Virtual Channel (VC) number, Data Type (DT), and Word Count (WC) to the controller. The Controller then creates a packet header and pulls the packet data from the Packet Interface and out to the D-PHY to transmit.

For Short Packets, the user provides the Virtual Channel (VC) number, Data Type (DT), and required parameters (if any) to the controller. The Controller then creates the short packet and sends it to the D-PHY to transmit. This interface enables the user application to transmit and receive any type of DSI packet.

The Packet Interface Transmit Interface ports are listed in the table below. Signals are synchronous to the Tx Byte clock.

NOTE

Only a subset of the Packet Interface Signals are accessible by user application through DSI_HOST_APB_PKT_IFx registers or chip-specific registers. For reset_byte_n and reset_esc_n signals, refer to chip-specific MIPI DSI information. For

tx_payload, tx_cmd_data_type, tx_cmd_vc, and tx_cmd_byte_count signals, refer to DSI_HOST_APB_PKT_IFx registers.

Table 45-4. DSI Host Controller Core Transmit Packet Interface

PORT	TYPE	DESCRIPTION
clk_byte	Input	Byte clock input. The D-PHY PPI interface, the tx_cmd and tx_payload interfaces are synchronous to clk_byte. This clock can be independent and unrelated to clk_esc.
clk_esc	Input	The low-speed D-PHY Escape Mode clock, which becomes TxClkEsc on the D-PHY interface. This clock can be independent and unrelated to clk_byte.
reset_byte_n	Input	Asynchronous reset, active low. This reset applies to all logic in the clk_byte clock domain.
reset_esc_n	Input	Asynchronous reset, active low. This reset applies to all logic in the clk_esc clock domain.
tx_payload[31:0]	Input	Packet data input.
tx_payload_en	Output	Packet data read enable. This active high signal indicates that the controller requires a valid packet during the next clk_byte period.
tx_payload_en_last	Output	Last packet read enable, active high signals last cycle of tx_payload_en.
tx_cmd_data_type[5:0]	Input	Transmit packet DSI data type. It is written into the command buffer when tx_cmd_ack is asserted high.
tx_cmd_vc[1:0]	Input	Transmit packet command virtual channel. It is written into the command buffer when tx_cmd_ack is asserted high.
tx_cmd_byte_count[15:0]	Input	Transmit packet payload byte count. It is written into the command buffer when tx_cmd_ack is asserted high. For DSI Long packet types, tx_cmd_byte_count defines the number of bytes of packet data to pull from the tx_payload port. For DSI Short packets, the format of tx_cmd_byte_count contains any optional parameters. If the SDI Short packet type does not have any parameters, it is recommended to set tx_cmd_byte_count to all 0s.
tx_cmd_req	Input	Transmit packet command request. This active high signal informs the controller that the packet command is valid. The packet command consists of the ports tx_cmd_data_type, tx_cmd_vc, and tx_cmd_byte_count. The controller will assert tx_cmd_ack when it accepts the command, after which, the user should either update port values for the next transmit packet command or deassert tx_cmd_req.
tx_cmd_ack	Output	Transmit packet command request acknowledge. This active high signal indicates that the controller has accepted the TX packet request and the user logic should either submit a new request or deassert tx_cmd_req on the next rising edge of clk_byte.

45.4.2.2 Receive Ports

The Receive Packet Interface returns data from the Peripheral to the user. The user is provided the Virtual Channel (VC) number, Data Type (DT), Word Count (WC), and Rx Payload Data.

This interface enables the user application to receive any type of DSI packet. These signals are synchronous to the clk_byte clock. The Packet Interface Receive Interface ports are listed in the table below.

NOTE

Only a subset of the Packet Interface signals are accessible by user application through DSI_HOST_APB_PKT_IFx registers and DSI_HOST device registers:

- rx_payload
- rx_cmd_vc
- rx_cmd_data_type
- rx_cmd_byte_count
- ecc_one_bit_error
- ecc_two_bit_error
- ecc_one_bit_error_pos[4:0]
- crc_err

See DSI_HOST_APB_PKT_IFx and DSI_HOST device sections for details.

Table 45-5. DSI Host Controller Core Receive Packet interface

PORT	TYPE	DESCRIPTION
rx_payload[31:0]	Output	Received packet data output. The Host Receive Packet Interface presents 4 bytes at a time. Bytes are valid in this interface according to the rx_cmd_byte_count signal, beginning with the lowest byte.
rx_payload_valid	Output	Packet data valid. This active high signal indicates that the controller is presenting valid packet during the next clk_byte period.
rx_payload_valid_last	Output	This data is the last of the packet, active high signals last cycle of rx_payload_valid.
rx_cmd_valid	Output	Packet header data is valid on the packet header ports below when this signal is asserted.
rx_cmd_vc[1:0]	Output	Packet virtual channel number, valid when rx_cmd_valid is asserted.
rx_cmd_data_type[5:0]	Output	Packet data type, valid when rx_cmd_valid is asserted. See the MIPI DSI-2 specification for a definition of possible values.
rx_cmd_byte_count[15:0]	Output	Packet Word Count (byte count). Contains the number of bytes of data in the received packet. Valid when rx_cmd_valid is asserted.

Table continues on the next page...

Table 45-5. DSI Host Controller Core Receive Packet interface (continued)

PORT	TYPE	DESCRIPTION
ecc_one_bit_error	Output	Single bit error in the packet header was detected and corrected. Active high. Valid when rx_cmd_valid is high.
ecc_two_bit_error	Output	Two packet header bit errors were detected and not corrected, active high. Valid when rx_cmd_valid is high.
ecc_one_bit_error_pos[4: 0]	Output	Position of the corrected single bit error in the packet header. Valid when ecc_one_bit_error is high.
crc_err	Output	Asserts high when the CRC calculated on the received data does not reach the end of the packet.

45.4.3 Status Interface Ports

The Status Interface Core provides for reading the Status of the DSI Host Controller Core via a simple interface. All signals in the Status Interface are synchronous to the clk_byte clock. The Status Interface ports are listed in the table below.

NOTE

Only a subset of the Packet Interface signals are accessible by user application through DSI_HOST_APB_PKT_IFx registers and DSI_HOST device registers:

- status_out
- hs_tx_timeout
- lp_rx_timeout

Refer to DSI_HOST_APB_PKT_IFx and DSI_HOST device sections for details.

Table 45-6. DSI Host Controller Status Interface Port

PORT	TYPE	DESCRIPTION
status_rd	Input	Initiates a read of the status register. Asserting this input high will result in the contents of the status register being presented on status_out[15:0] along with status_out_valid asserting high to indicate that status_out[15:0] contains a valid value.
status_addr[3:0]	Input	When status_rd is asserted, indicates which status register is to be returned. Address 0x0 is a valid address while 0x1-0xF are reserved.
status_out[31:0]	Output	Contains the contents of the addressed status register.
status_out_valid	Output	When asserted high, indicates that the data on status_out is valid.
tx_active	Output	tx_active asserts high when the Host Controller is actively transmitting data or when it has accepted a request from the user but has not yet started transmitting.

Table continues on the next page...

Table 45-6. DSI Host Controller Status Interface Port (continued)

PORT	TYPE	DESCRIPTION
hs_tx_timeout	Output	Asserts high for one clk_byte period to indicate that a High Speed transmit has timed out.
lp_rx_timeout	Output	Asserts high for one clk_byte period to indicate that a Low Power RX curred.

45.4.4 Control and Status Ports

The Control and Status Packet Interface allows the user to control and receive the status of the underlying link between the DSI Host and DSI Peripheral. It works on the same clk_byte clock as the Host Transmit Packet Interface. The Packet Interface Control and Status Interface ports are listed in the table below.

NOTE

Only a subset of the Packet Interface signals may be accessible by user application through DSI_HOST_APB_PKT_IFx registers and DSI_HOST device registers:

- dphy_direction
- tx_hs_mode
- dphy_turnaround

Please refer to DSI_HOST_APB_PKT_IFx and DSI_HOST device sections for details.

Table 45-7. DSI Host Controller Core Control And Status Packet Interface

PORT	TYPE	DESCRIPTION
tx_hs_mode	Input	Switches the DPHY into High Speed Data Transfer mode or Low Power Data Transfer mode. 1'b1 = request HS mode 1'b0 = request LP mode. The Packet interface will not acknowledge packet commands or data while switching modes.
dphy_turnaround	Input	Requests bus turnaround. 1'b1 = Request reverse direction LP mode, from Host TX to Host RX. 1'b0 = No effect. This signal is ignored if the bus is already in Reverse (Host RX) direction.
dphy_direction	Output	Reports the current bus direction. 1'b1 = Bus is in Reverse direction (Host RX). 1'b0 = Bus is in Forward direction (Host TX).

45.4.5 Interface Cores

There are several interface cores for the DSI Host Controller Core. These interface cores provide industry standard interfaces to the DSI Host Controller Core to simplify interfacing and integration the DSI Host Controller Core to your application. When selected, these interface cores provide additional ports to the DSI Host Controller core.

45.4.5.1 DPI-2 Interface Host Bridge Core

The DPI-2 Interface Host Bridge Core provides a DPI-2 compliant interface to the DSI Host controller. The DPI-2 Interface Host Bridge Core handles converting the DPI-2 interface into a packet format that is compatible with the DSI Host Controller Core's Packet Interface, and that adheres to the DSI specification. The DPI-2 Interface Core handles all pixel-to-byte packing for all DPI data types.

The DSI Host Controller DPI-2 Interface Core provides the following features:

- Support for Type 2,3, and 4 displays
- Support for 16-, 18- and 24- bit Pixel data and all alignment configurations
- Support for 16, 18, 24, and 18 bit loosely packed DSI data types
- Supports DSI video modes
  - Non-Burst Mode with Sync Pulses
  - Non-Burst Mode with Sync Events
  - Burst Mode
- Supports normal or inverted HSYNC and VSYNC signals
- Handles clock domain crossing from DPI Pixel clock to the Host controller TX Byte clock
- Interfaces directly to the Host Controller's DSI Packet Interface
- Comes already integrated with the DSI Host Controller

A block diagram of the DPI-2 Interface Host Bridge Core is shown below. The DPI-2 Host Bridge Core accepts a MIPI compliant DPI-2 set of signals (vsync, hsync, pixel data), creates DSI packets for the video timing events and video data and transmits those packets via the DSI Host Controller's Packet Interface.

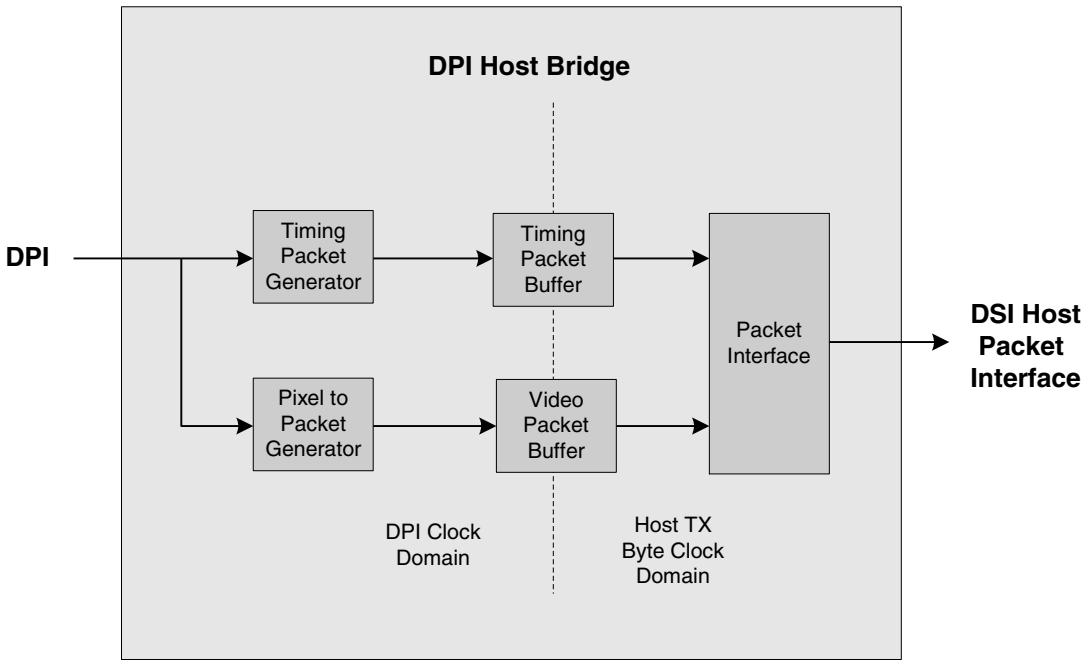


Figure 45-2. DPI-2 Host Bridge Core Block Diagram

45.4.5.1.1 DPI-2 Interface Ports

The DPI-2 Interface module provides a MIPI compliant DPI-2 interface to the DSI Host Controller Core. In addition to the standard DPI-2 interface signals (prefixed with a dpi_) there are several configuration and status ports that allow the user to adjust the behavior of the DPI-2 Interface module to better suit a particular application.

NOTE

The DPI configuration signals herein described are accessible to User Application through chip-specific registers (please refer to chip-specific MIPI-DSI information section for details) and DSI_HOST_DPI_INTFC device registers (please refer to MIPI DSI Host Memory Map/Register section for details).

Table 45-8. DSI Host Controller Core Receive Packet interface

PORT	TYPE	DESCRIPTION
dpi_sd	Input	Shut Down – Control to shutdown display (type 4 only) 1'b1= Send shutdown command. 1'b0= No effect
dpi_cm	Input	Color Mode control. 1'b0== Normal Mode 1'b1== Low-color Mode
cfg_dpi_pixel_payload_size [15:0]	Input	Maximum number of pixels that should send as one DSI packet. Recommended to be evenly divisible by the line size (in pixels).

Table continues on the next page...

Table 45-8. DSI Host Controller Core Receive Packet interface (continued)

PORT	TYPE	DESCRIPTION
cfg_dpi_pixel_fifo_send_ level[15:0]	Input	In order to optimize DSI utility, the DPI bridge buffers a certain number of DPI pixels before initiating a DSI packet. This configuration port controls the level at which the DPI Host bridge begins sending pixels.
cfg_dpi_interface_color_ coding[2:0]	Input	Sets the distribution of RGB bits within the 24-bit d bus, as specified by the DPI specification. 0= 16-bit Configuration 1 1= 16-bit Configuration 2 2= 16-bit Configuration 3 3= 18-bit Configuration 1 4= 18-bit Configuration 2 5= 24-bit
cfg_dpi_pixel_format[1:0]	Input	Sets the DSI packet type of the pixels. 0= 16-bit 1= 18-bit 2= 18-bit loosely packed, 3= 24-bit
dpi_host_underrun_err	Output	During DSI Host transmission of DPI data insufficient DPI data was received. This may indicate that DPI_CLK is too slow, or that the cfg_dpi_* parameters are incorrectly set.
cfg_dpi_vsync_polarity	Input	Sets polarity of dpi_vsync input, 0 – active low, 1 active high
cfg_dpi_hsync_polarity	Input	Sets Polarity of dpi_hsync input, 0 – active low, 1 – active high
cfg_host_dpi_video_mode[1:0]	Input	Select DSI video mode that the host DPI module should generate packets for. 2'b00 – Non-Burst mode with Sync Pulses 2'b01 – Non-Burst mode with Sync Events 2'b10 – Burst mode 2'b11 – Reserved, not valid
cfg_host_dpi_hfp[15:0]	Input	Sets the DSI packet payload size, in bytes, of the horizontal front porch blanking packet.
cfg_host_dpi_hbp[15:0]	Input	Sets the DSI packet payload size, in bytes, of the horizontal back porch blanking packet.
cfg_host_dpi_hsa[15:0]	Input	Sets the DSI packet payload size, in bytes, of the horizontal sync g packet.
cfg_enable_mult_packets	Input	Enable Multiple packets per video line. When enabled, cfg_dpi_pixel_payload_size must be set to exactly half the size of the video line. 0 – Video Line is sent in a single packet 1 – Video Line is sent in two packets
cfg_bllp_mode	Input	Optimize bllp periods to Low Power mode when possible 0 – blanking packets are sent during BLLP periods 1 – LP mode is used for BLLP periods

45.4.5.1.2 DPI-2 Video Timing Recreation over DSI

The DSI Host DPI-2 module is designed to preserve the receive video timing by properly spacing the DSI packets that carry the DPI video information and by inserting blanking packets into the packet stream. In order to achieve this goal, the DSI Host controller needs to have the cfg_host_dpi_hfp, cfg_host_dpi_hbp, and cfg_host_dpi_hsa ports properly set. The values of these ports reflect the size of the hfp, hbp, and has in terms of DSI packet bytes, values such that the number of DSI bytes when transmitted takes approximately the same amount of absolute time as the video event took on the DPI interface.

To relate pixel sizes on the DPI interface to DSI packet bytes, use the following relationship:

Time of DPI event - time to transmit x number of bytes on the DSI interface

dpi_event_size * dpi_pclk_period = number_of_dsi_bytes * 8 * dsi_hs_bit_period / cfg_num_lanes

Solving for number_of_dsi_bytes:

number_of_dsi_bytes = dpi_event_size * dpi_pclk_period * cfg_num_lanes / (8 * dsi_hs_bit_period)

number_of_dsi_bytes - value to set the cfg_host_dpi_hfp (hbp, or hsa) port to
dpi_event_size - size of the video event in pixels (hfp, hbp, hsa)
dpi_pclk_period - period of dpi_pclk
dsi_hs_bit_period - bit period of the mipi data lanes in High Speed mode
cfg_num_lanes - host controller setting for number of active lanes, 0 = 1 lane, 1 = 2 lanes, 2 = 3 lanes, 3 = 4 lanes)

45.4.6 RAM Usage

The DSI Host Controller Core requires on-chip RAM arrays. These RAM arrays and their corresponding sizes are listed in the table below.

All RAM arrays used in this core are configured as two-port, having the following attributes:

- Two address busses, one for the write port and one for read port
- Two data busses, one for write data on the write port and one for read data on the read port
- Two clock inputs, one for the write port and one for the read port. (Note: In some cases, the core logic will connect the same clock to both the read and write clock inputs of the RAM.)

Ram Purpose	Size (Depth x Width)
DPI Core Pixel Data FIFO	2048 x 32
APB to Packet Interface TX FIFO	64 x 32
APB to Packet Interface RX FIFO	64 x 32

45.4.7 DSI Host Clocking

The DSI Host Controller requires the following clocks:

- Reference clock (ref_clk) - used by the DPHY (DPHY PLL generates byte clock)

- Byte clock (clk_byte) - generated by the DPHY, used for packet generation and transfer to the DPHY
- TX escape mode clock (clk_tx_esc) - for internal LPDT state machines and low-power transmissions;
- Pixel clock (dpi_pclk)

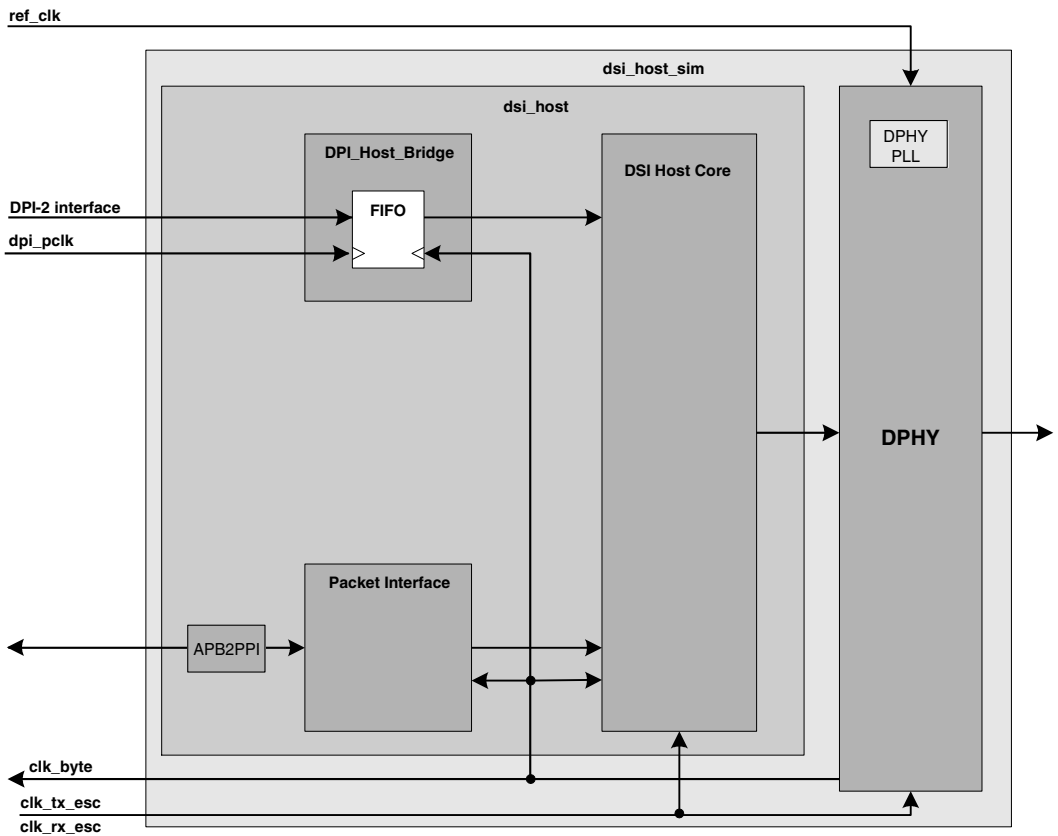


Figure 45-3. DSI Host Controller Clocking Block Diagram

45.4.7.1 ref_clk

The reference clock is used by the DPHY (DPHY PLL) to generate the High Speed MIPI clock and High Speed Data Lane signaling. The frequency of the resulting clock generated from DPHY PLL is typically equal to the high speed bit rate of the data lanes. The DPHY will also use this clock to generate the clk_byte clock output used by the Host DSI Controller and user application logic.

The MIPI DPHY PLL must be setup through the MIPI_DSI_HOST_FSL_IP1_DPHY INTFCx registers, prior to starting DSI Host Controller and DPHY operation. For details, please refer to MIPI DSI Host IP1 DPHY INTFC Memory section.

45.4.7.2 clk_byte

The clk_byte clock is generated by the DPHY (DPHY PLL) and the frequency is 1/8th the data rate of the DPHY Data Lane High Speed data rate. A DPHY configured to generate High Speed data at 1Gbps/lane would generate a 125MHz clk_byte clock.

The clk_byte clock is used by the DSI Host Controller to interface to the MIPI Tx DPHY High Speed PPI interface. The DSI Host Controller's Packet Interface is synchronous to the clk_byte clock.

45.4.7.3 dpi_pclk

The dpi_pclk clock is used on the Host DPI-2 interface. All of the Host DPI-2 signals are synchronous to this clock. The DSI Host Controller's DPI Bridge module handles transferring video data received in the dpi_pclk clock domain over to the clk_byte clock domain.

The dpi_pclk and clk_byte frequencies are related by the following formula:
clk_byte_freq >= dpi_pclk_freq * DPI_pixel_size / (8 * (cfg_num_lanes + 1))

cfg_num_lanes = the configuration port setting that selects the number of active MIPI DPHY data lanes clk_byte_freq = frequency of clk_byte which is 1/8th the High Speed data lane rate. dpi_pclk_freq = frequency of the dpi_pclk clock on the DPI-2 interface. DPI_pixel_size = size of pixels, in bits, on the DPI-2 interface

If the clk_byte frequency does not meet this requirement, the MIPI interface will not be able to keep up with the video stream on the DPI-2 interface resulting in dropped video lines.

45.4.7.4 clk_tx_esc

The clk_tx_esc clock is used by the MIPI Tx DPHY for state control and low power data transmission. The DSI Host Controller also uses clk_tx_esc for the portion of controller logic that interfaces to the MIPI Tx DPHY that are synchronous to clk_tx_esc. The frequency of clk_tx_esc is defined by the requirements of the MIPI interface and the MIPI Tx DPHY. For details about the clk_tx_esc source, please refer to the chip-specific MIPI-DSI clocking section.

45.4.8 Reset and Initialization

Reset and Initialization procedure for the DSI Host Controller is as follows:

1. Assert all resets through the chip-specific registers provided for this purpose:
   - reset_dpi_n
   - reset_byte_n
   - reset_esc_n
2. Setup DSI related clocks through chip-specific registers provided for this purpose (refer to SCG for details).
3. Setup MIPI DSI application by programming all DSI Host Controller and DSI DPHY required parameters (refer to MIPI DSI Host Memory Map/Register for details).
4. Deassert all remaining resets through the Chip-specific registers provided for this purpose:
   - reset_dpi_n
   - reset_byte_n
   - reset_esc_n
5. DSI Host Controller should be ready for use

45.5 DSI D-PHY

The figure below shows the block diagram of the MIPI DSI D-PHY. It consists of a clock lane module and two data lane modules. Each of these PHY Lane Modules communicates through a differential line to a complementary PHY at the other side of the lane interconnect.

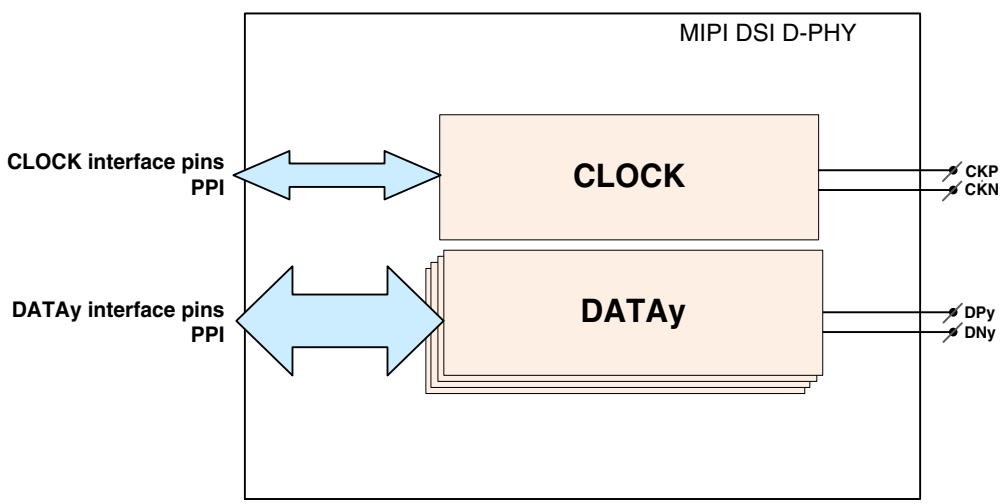


Figure 45-4. MIPI D-PHY Block Diagram

The figure below shows the D-PHY Lane components:

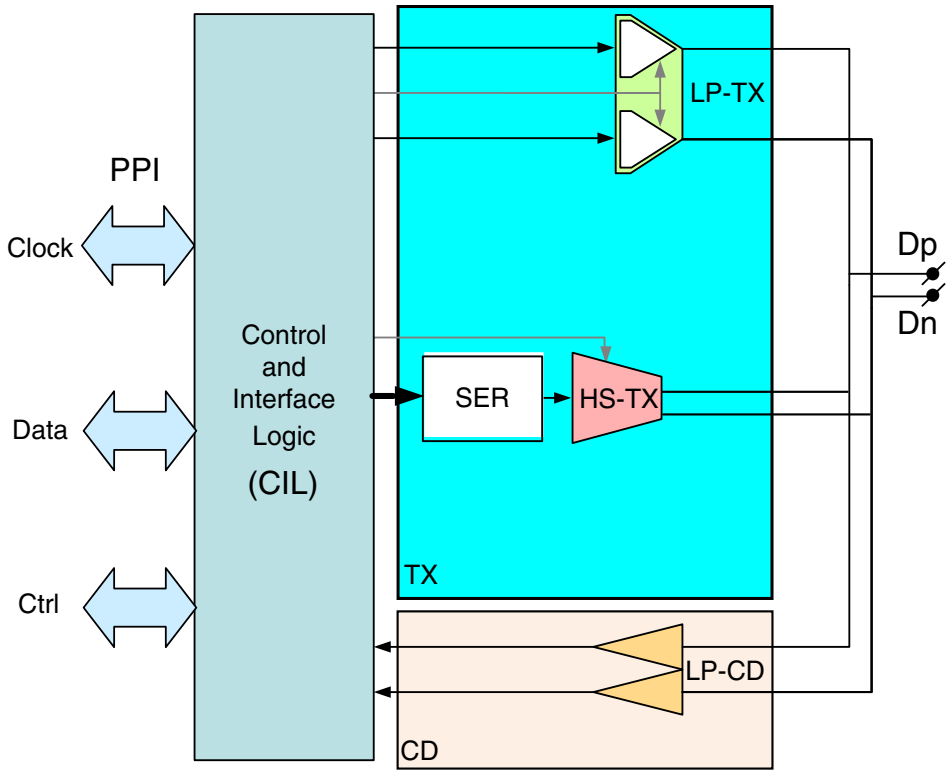


Figure 45-5. D-PHY Lane Diagram

The DATAy are bidirectional lanes and consist of High-Speed Transmitter (HS-TX), Low-Power Transmitter (LP-TX), Serializer (SER), and Low- Power Contention Detector (LP-CD).

The Clock lane is a bidirectional lane and consists of High-Speed Transmitter (HS-TX), and Low-Power Transmitter (LP-TX).

The Control and Interface Logic (CIL) module interfaces with the Protocol and determines the global operation of the Lane Module. The interface between the D-PHY and the protocol is called the PHY-Protocol Interface (PPI).

During normal operation, a Lane switches between Low-Power and High-Speed mode. Bidirectional Lanes can also switch communication direction. The change of operating mode or direction requires enabling and disabling of certain electrical functions. These enable and disable events do not cause glitches on the Lines that would result in a detection of an incorrect signal level. Therefore, all mode and direction changes are smooth to always ensure a proper detection of the Line signals.

45.5.1 High-Speed Transmitter (HS-TX)

The differential output voltage VOD is defined as the difference of the voltages V_{DP} and V_{DN} at the Dp and Dn pins, respectively.

VOD =VDP −VDN

The output voltages V_{DP} and V_{DN} at the Dp and Dn pins will not exceed the high-speed output high voltage V_{OHHS}. The common-mode voltage V_{CMTX} is defined as the arithmetic mean value of the voltages at the Dp and Dn pins:

VCMTX = (VDP + V DN)/2

V_{OD} and V_{CMTX} are graphically shown in Figure 45-6 for ideal HS signals. Figure 45-7 shows single ended HS signals with the possible kinds of distortion of the differential output and common mode voltages. V_{OD} and V_{CMTX} may be slightly different for driving a Differential-1 or a Differential-0 on the pins.

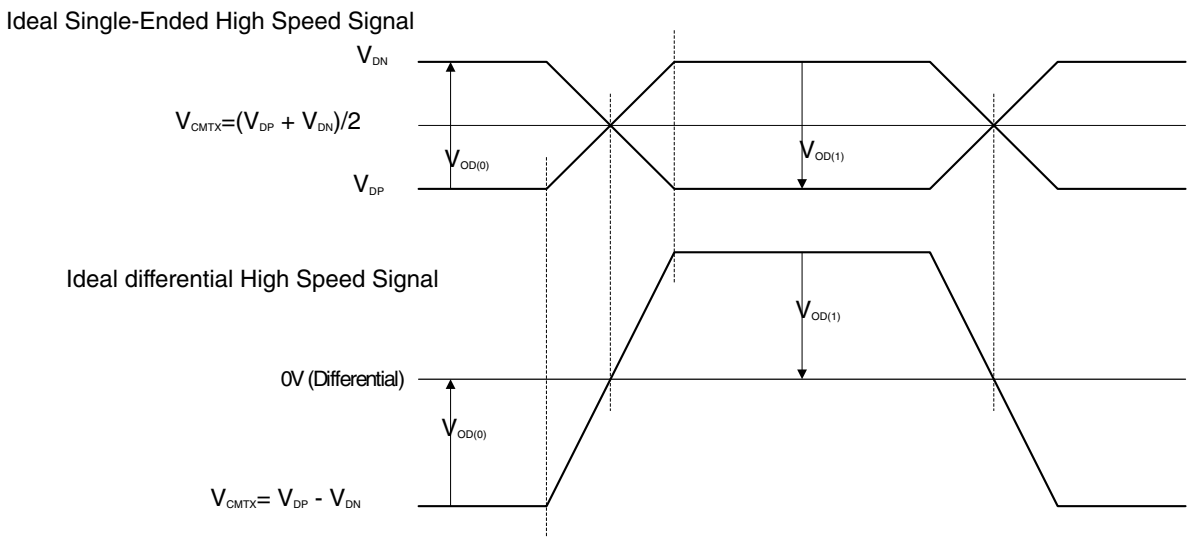


Figure 45-6. Ideal Single-ended and Resulting Differential HS Signals

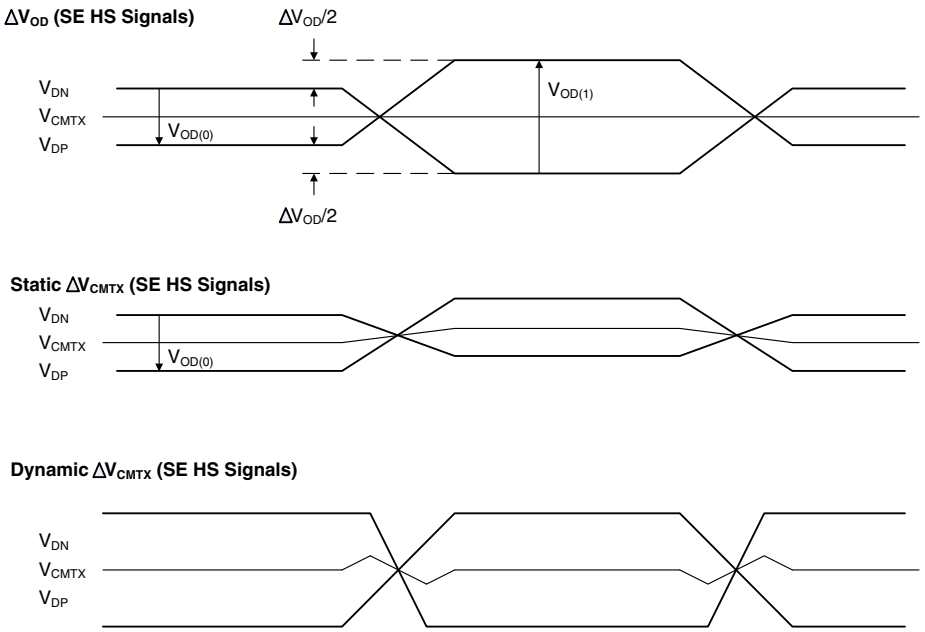


Figure 45-7. Possible ΔVcmtx and ΔVod Distortions of the Single-Ended HS Signals

The output differential voltage mismatch ΔVOD is defined as the difference of the absolute values of the differential output voltage in the Differential-1 state VOD(1) and the differential output voltage in the Differential-0 state VOD(0). This is expressed by:

$\Delta V = |V_{OD(1)}| - |V_{OD(0)}|$

If VCMTX(1) and VCMTX(0) are the common-mode voltages for static Differential-1 and Differential- 0 states respectively, then the common-mode reference voltage is defined by:

$V_{CMTX,\,REF} = (V_{CMTX(1)} + V_{CMTX(0)})/2$

The transient common-mode voltage variation is defined by:

$\Delta V_{CMTX\,(t)} = V_{CMTX\,(t)} - V_{CMTX,REF(t)}$

The static common-mode voltage mismatch between the Differential-1 and Differential-0 state is given by:

$\Delta V_{CMTX\,(1,0)} = (V_{CMTX\,(1)} - V_{CMTX\,(0)})/2$

A test circuit for the measurement of VOD and VCMTX is shown:

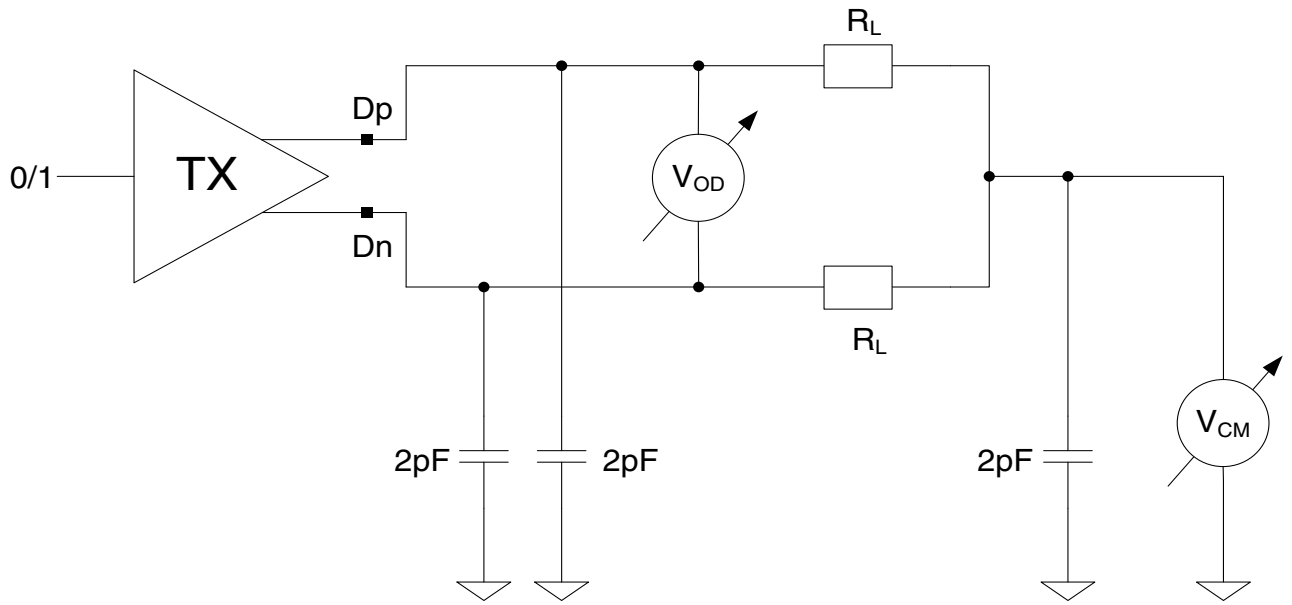


Figure 45-8. Test Circuit for Vcmtx and Vod measurement

The single-ended output impedance of the transmitter at both the Dp and Dn pins is denoted by Z_{OS}. ΔZ_{OS} is the mismatch of the single ended output impedances at the Dp and Dn pins, denoted by ZOSDP and ZOSDN respectively. This mismatch is defined as the ratio of the absolute value of the difference of Z_{OSDP} and Z_{OSDN} and the average of those impedances:

$$\Delta Z_{OS} = (2*|Z_{OSDP} - Z_{OSDN}|)/(Z_{OSDP} + Z_{OSDN})$$

The output impedance ZOS and the output impedance mismatch ΔZOS is compliant with High Speed Transmitter DC Specifications for both the Differential-0 and Differential-1 states for all allowed loading conditions. It is recommended the design keep the output impedance during state transitions as close as possible to the steady state value. The output impedance ZOS can be determined by injecting an AC current into the Dp and Dn pins and measuring the peak-to-peak voltage amplitude. The test circuit for such a measurement is shown in the following figure, where RL are load resistors and VTEST,PP is a test signal applied at the common-mode node via an AC coupling capacitor CAC. The frequency of the test signal is fTEST.

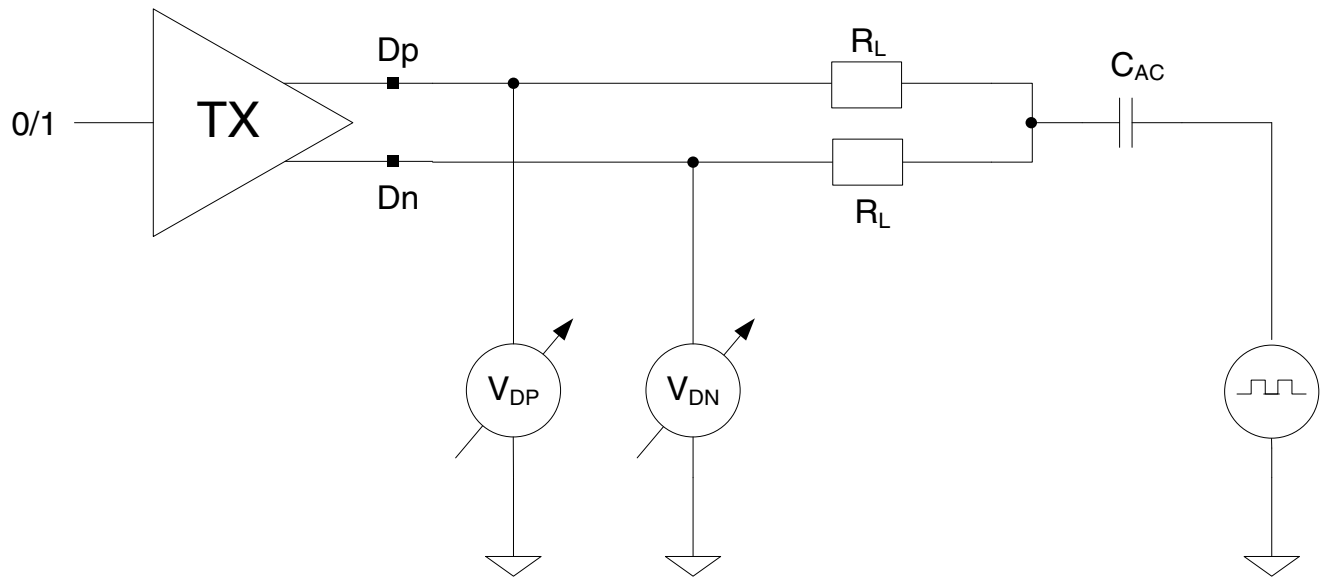


Figure 45-9. Test Circuit for Output Impedance measurements

Assuming negligible effect of the coupling capacitor, the example circuit gives the following relation between peak-to-peak pin voltage VPIN,PP and ZOS. This equation can be used for both Dn and Dp pins individually.

$Z_{OS} = (V_{PIN,PP} * RL)/(V_{TEST,PP} - Z_{PIN,PP})$

Where VPIN,PP is the peak-to-peak voltage at the pin Dp or Dn.

45.5.2 Low-Power Transmitter (LP-TX)

The Low-Power transmitter is a slew-rate controlled push-pull driver. It is used for driving the Lines in all Low-Power operating modes. The LP transmitter has been optimized to minimize static power consumption. The slew-rate of signal transitions is bounded in order to keep EMI low. The LP transmitter will not drive the pad pin potential statically beyond the maximum value of V_{OH}. V_{OH} is the thevenin output, high-level voltage in the high-level state, when the pad pin is not loaded. V_{OL} is the thevenin output, low-level voltage in the LP transmit mode. This is the voltage at an unloaded pad pin in the low-level state.

45.5.3 Low-Power Contention Detector (LP-CD)

Contention can be inferred from any of the following conditions:

- LP high fault is detected when the LP transmitter is driving high and the pin voltage is less than VIL.
- LP low fault is detected when the LP transmitter is driving low and the pad pin voltage is greater than VIHCD. An LP low fault will not be detected when the pin voltage is less than VILCD.

The general operation of a contention detector is similar to that of an LP receiver with lower threshold voltages. Although the DC specifications differ, the AC specifications of the LP-CD are defined to match those of the LP receiver. The LP-CD sufficiently filters the input signal to avoid false triggering on short events

The LP-CD threshold voltages (VILCD, VIHCD) are shown along with the normal signaling voltages:

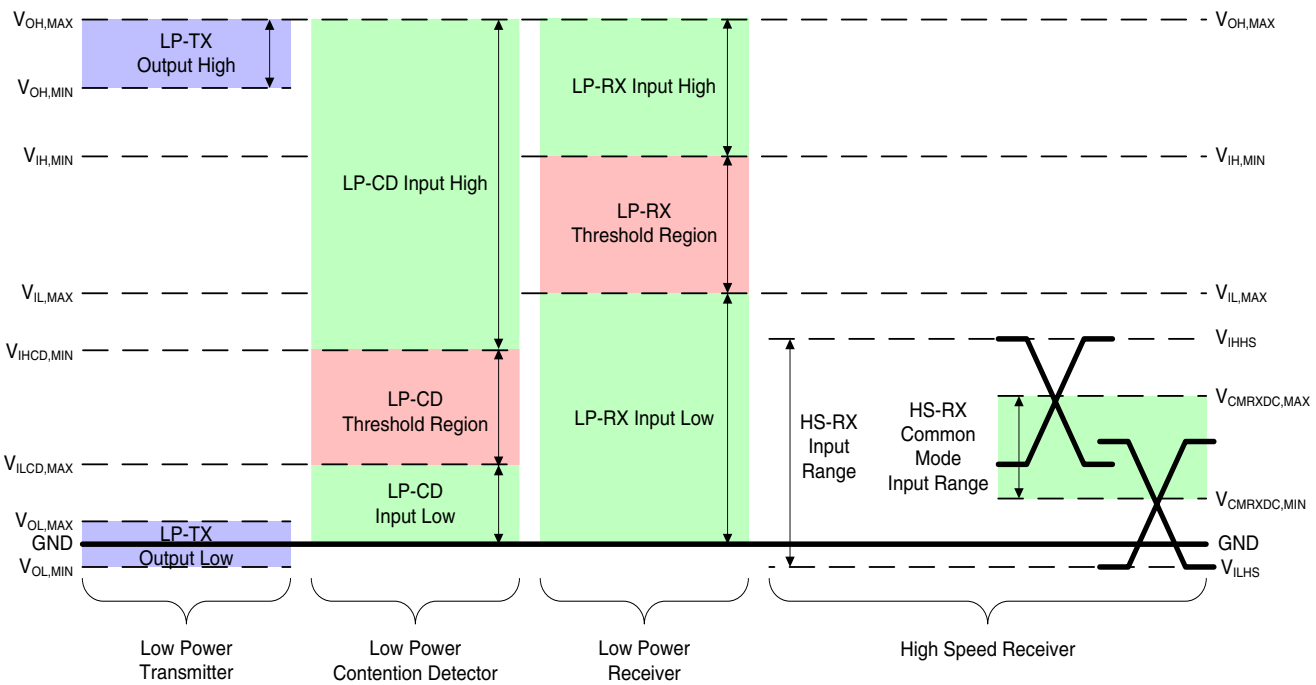


Figure 45-10. Signaling and MIPI Contention Voltage Levels

45.5.4 Input Characteristics

No structure within the PHY may be damaged when a DC signal that is within the signal voltage range VPIN is applied to a pad pin for an indefinite period of time. $V_{PIN(absmax)}$ is the maximum transient output voltage at the transmitter pin. The voltage on the transmitter's output pin will not excel $V_{PIN,MAX}$ for a period greater than $T_{VPIN(absmax)}$. When the PHY is in the low-power receive mode the pad pin leakage current will be I_{LEAK} when the pad signal voltage is within the signal voltage range of V_{PIN}. The

specification of ILEAK assures interoperability of any PHY in the LP mode by restricting the maximum load current of an LP transmitter. The test circuit for leakage current measurement is shown:

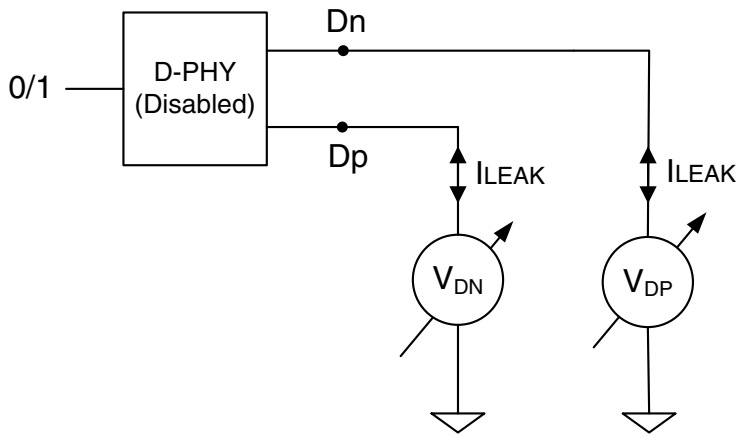


Figure 45-11. Pin-Leakage Measurement DC specification

45.5.5 High-Speed Serializer

The HS-Serializer converts 8-bits parallel data into 1 bit data stream to be transmitted by HSTX module. The CIL engine should clock out 8-bits parallel data, HSTX_DATA[7:0], on D0_HS_BYTE_CLKS rising edges. Then the Serializer will capture HSTX_DATA[7:0] on D0_HS_BYTE_CLKS rising edges and the signal connected to HSTX_DATA[0] is transmitted first. There is setup timing constraint and hold timing constraint designated to the rising edge of D0_HS_BYTE_CLKS as constrained in the liberty view provided.

The figure shows the relationship between HS_TXCLKP, HS_TXCLKN, HS_SER_LD, HS_BYTE_CLKS and HSTX_DATA[7:0]. It also shows how the 8-bits of parallel bits are mapped into the serial link:

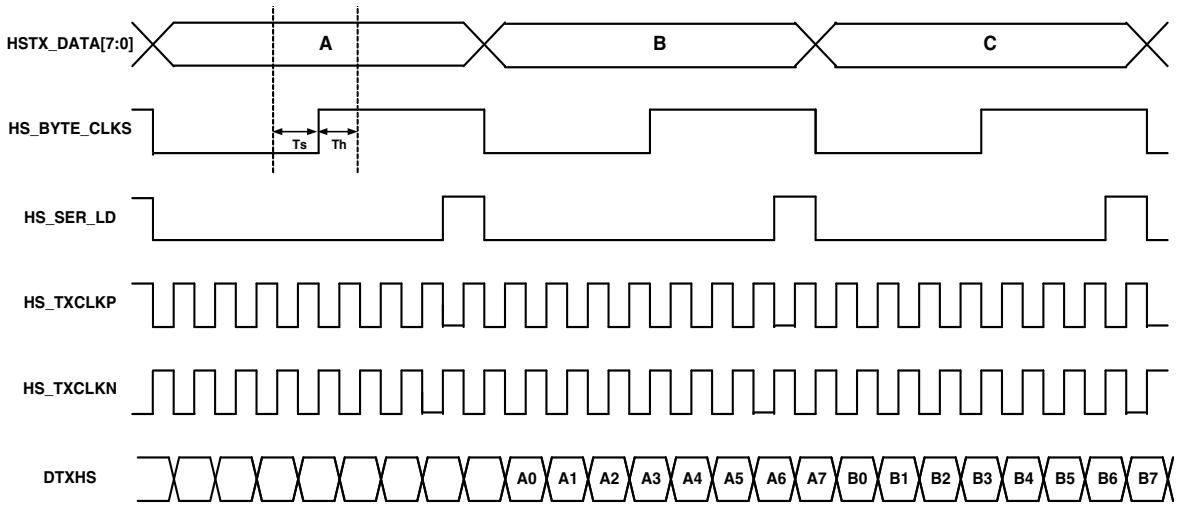


Figure 45-12. Serializer Timing Diagram

45.5.6 DSI D-PHY Signals

This section details the interface signals for MIPI DSI D-PHY. For detailed signal spec, please refer to the chip datasheet.

Table 45-9. DSI DPHY PPI Signals

Signal	Type	Description
TxClkEsc	Clock	Escape mode Transmit Clock. This clock is directly used to generate escape sequences. The period of this clock determines the symbol time for low power signals. It is therefore constrained by the normative part of the DPHY specification. **NOTE:** The max frequency of TxClkEsc is 20 MHz **NOTE:** The Min frequency of TxClkEsc is 12 MHz.
TxByteClkHS	Clock	High-Speed Transmit Byte Clock. This is used to synchronize PPI signals in the High-Speed transmit clock domain. It is recommended that all transmitting Data Lane Modules share on TxByteClkHS signal. The frequency of TxByteClkHS is exactly 1/8 the High-Speed bit rate.

45.5.7 Operating Modes

45.5.7.1 High-Speed Transmit from the Master Side

The figure shows an example of a High-Speed transmission on the Master side. While TxRequestHS is low, the Lane Module ignores the value of TxDataHS. To begin transmission, the protocol drives TxDataHS with the first byte of data and asserts TxRequestHS. This data byte is accepted by the PHY on the first rising edge of TxByteClkHS with TxReadyHS also asserted. At this point, the protocol logic drives the next data byte onto TxDataHS. After every rising clock cycle with TxReadyHS active, the protocol supplies a new valid data byte or ends the transmission. After the last data byte has been transferred to the Lane Module, TxRequestHS is driven low to cause the Lane Module to stop the transmission and enter Stop state. The minimum number of bytes transmitted could be as small as one.

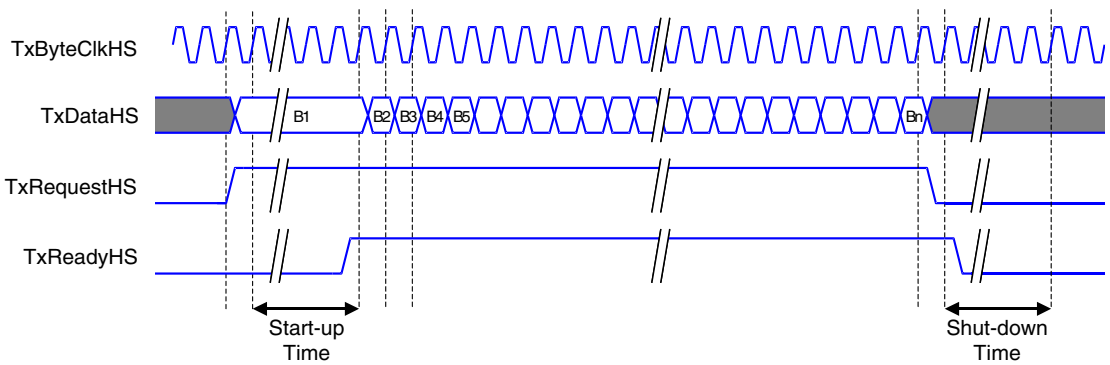


Figure 45-13. High-Speed Transmission from the Master Side

45.5.7.2 Low-Power Data Transmission

For Low-Power data transmission the TxClkEsc is used. The Protocol directs the Data Lane to enter Low-Power data transmission Escape mode by asserting TxRequestEsc with TxLpdtEsc high. The Low-Power transmit data is transferred on the TxDataEsc lines when TxValidEsc and TxReadyEsc are both active at a rising edge of TxClkEsc. The byte is transmitted in the time after the TxDataEsc is accepted by the Lane Module (TxValidEsc = TxReadyEsc = high). The Protocol knows the byte transmission is finished when TxReadyEsc is asserted. After the last byte has been transmitted, the protocol de-asserts TxRequestEsc to end the Low-Power data transmission. This causes TxReadyEsc to return low. Whenever TxRequestEsc transitions from high-to-low, it always remains in the low state for a minimum of two TxClkEsc clock cycles. The figure shows an example Low-Power data transmission operation.

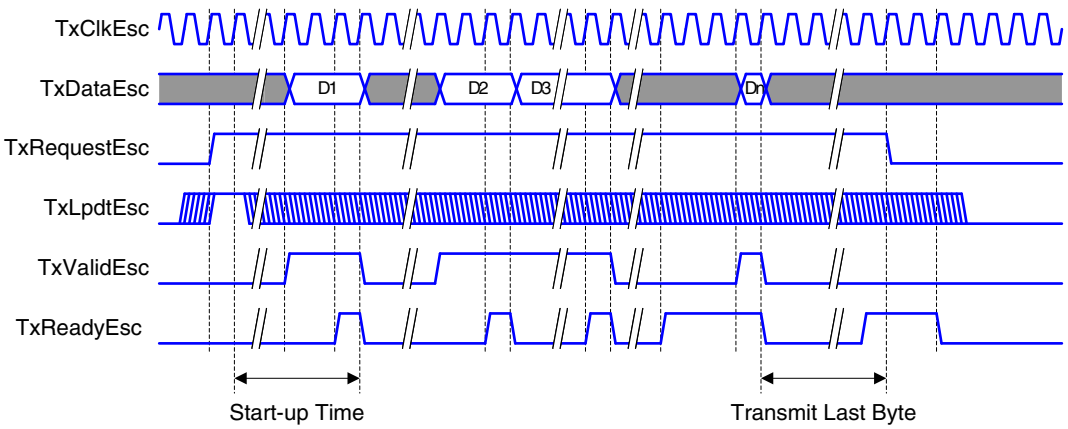


Figure 45-14. Low-Power Data Transmission

45.5.7.3 Turn-around

If the Master side and Slave side Lane Modules are both bi-directional, it is possible to turn around the Link for Escape mode signaling. Which side is allowed to transmit is determined by passing a “token” back and forth. That is, the side currently transmitting passes the token to the receiving side. If the receiving side acknowledges the turn-around request, as indicated by driving the appropriate line state, the direction is switched. The figure shows an example of two turn-around events. At the beginning, the local side is the transmitter, as shown by Direction=0. When the protocol on this side wishes to turn the Lane around (i.e. give the token to the other side), it asserts TurnRequest for at least one cycle of TxClkEsc. This initiates the turn around procedure. The remote side acknowledges the turn-around request by driving the appropriate states on the Lines. When this happens, the local Direction signal changes from transmit (0) to receive (1). The remote side initiates a turn-around request, passing the token back to the local side. When this happens, the local Direction signal changes back to transmit (0). The transmitter is in control of the Link direction and decides when to turn the Link around, passing control to the receiver. If the remote side does not acknowledge the turnaround request, the Direction signal does not change.

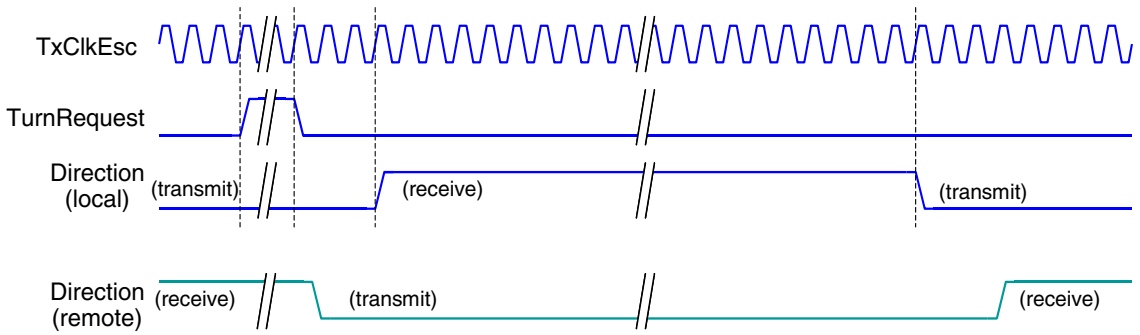


Figure 45-15. Turn-around Actions Transmit-to-Receive and Back to Transmit

45.6 DPHY PLL

The MIPI DSI DPHY PLL is a high performance PLL based frequency synthesizer that incorporates a lock detector, independent output divider, and supports power down modes. The PLL block diagram is shown below.

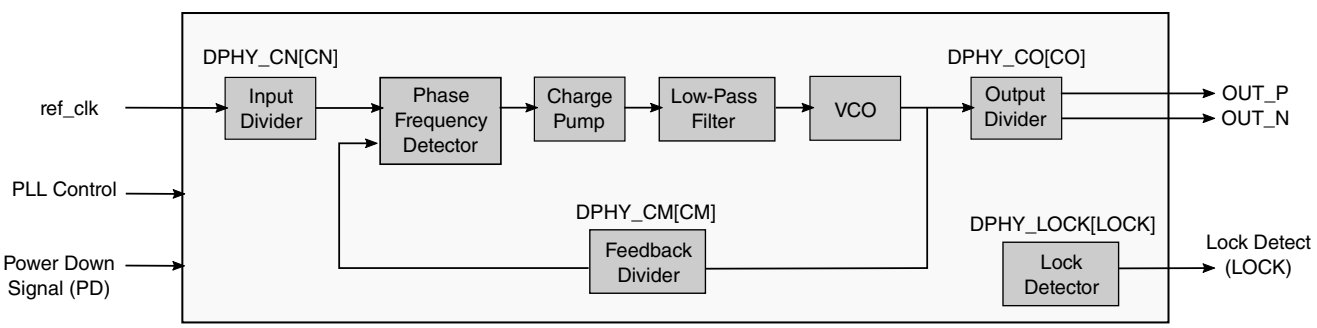


Figure 45-16. MIPI DSI DPHY PLL Block Diagram

The DPHY PLL input clock is ref_clk. The PLL output multiplies the input frequency by `(CM / (CN * CO))`.

NOTE

The input frequency ranges from 24 MHz till 200 MHz. The input divider has to be programmed such that the frequency after the input divider ranges from 24 MHz till 30 MHz.

NOTE

The VCO maximum output frequency is 800 MHz.

The power down signal (PD) is active-high and used to power down the PLL. Asserting PD will reset the PLL to it's initial state. To ensure proper PLL functionality, ref_clk needs to be stable before PLL power up. The expected power-on sequence for the DSI DPHY PLL is illustrated below.

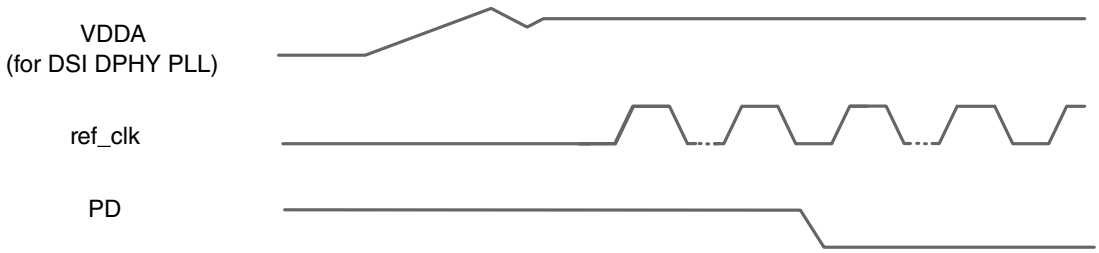


Figure 45-17. MIPI DSI DPHY PLL Power-on Sequence

45.7 DPHY Programming

The following sections detail the programming of the MIPI DSI DPHY high-speed timers.

45.7.1 High-Speed Prepare Timer ($T_{HS-PREPARE}$)

Calculate the value for clock lane high-speed prepare timer (mc_PRG_HS_PREPARE) using the formula below:

Assuming TxClkEsc = 20 MHz (T_{per} = 50 ns)
When mc_PRG_HS_PREPARE = 0, $T_{CLK-PREPARE}$= 1 * TxClkEsc Period = 50 ns
When mc_PRG_HS_PREPARE = 1, $T_{CLK-PREPARE}$ = 1.5 * TxClkEsc Period = 75 ns

Calculate the value for data lane high-speed prepare timer (m_PRG_HS_PREPARE) using the formula below:

Assuming TxClkEsc (TLPX) = 20 MHz (T_{per} = 50 ns)
When m_PRG_HS_PREPARE = 00, $T_{HS-PREPARE}$ = 1 * TxClkEsc Period = 50 ns
When m_PRG_HS_PREPARE = 01, $T_{HS-PREPARE}$ = 1.5 * TxClkEsc Period = 75 ns
When m_PRG_HS_PREPARE = 10, $T_{HS-PREPARE}$ = 2 * TxClkEsc Period = 100 ns
When m_PRG_HS_PREPARE = 11, $T_{HS-PREPARE}$ = 2.5 * TxClkEsc Period = 125 ns

45.7.2 High-Speed Zero Timer($T_{HS-ZERO}$)

Calculate the value for clock lane high-speed zero timer (mc_PRG_HS_ZERO) using the formula below:

$T_{CLK-ZERO}$= (mc_PRG_HS_ZERO + 3)*(TxByteClkHS Period)
TxByteClkHS = 1/8 Data Rate

When mc_PRG_HS_ZERO = 000110 @ 250 Mb/s, $T_{CLK-ZERO}$ = (6+3) * 32 ns = 288 ns
When mc_PRG_HS_ZERO = 100000 @ 1 Gb/s, $T_{CLK-ZERO}$ = (32+3) * 8 ns = 280 ns

Calculate the value for data lane high-speed zero timer (m_PRG_HS_ZERO) using the formula below:

$T_{HS-ZERO}$ = (m_PRG_HS_ZERO + 6)*(TxByteClkHS Period)
TxByteClkHS = 1/8 Data Rate
When m_PRG_HS_ZERO = 00001 @ 250 Mb/s, $T_{HS-ZERO}$ = (1+6) * 32 ns = 224 ns
When m_PRG_HS_ZERO = 01001 @ 1 Gb/s, $T_{HS-ZERO}$ = (9+6) * 8 ns = 120 ns

45.7.3 High-Speed Trail Timer ($T_{HS-TRAIL}$)

Calculate the value for the high-speed trail timer (PRG_HS_TRAIL) using the formula below:

$T_{HS-TRAIL}$= (PRG_HS_TRAIL)*(TxByteClkHS Period)
TxByteClkHS = 1/8 Data Rate
When PRG_HS_TRAIL = 0100 @ 250 Mb/s, >$T_{HS-TRAIL}$ = 4 * 32 ns = 128 ns
When PRG_HS_TRAIL = 1100 @ 1 Gb/s, >$T_{HS-TRAIL}$ = 12 * 8 ns = 96 ns

45.8 Packet Data and Pixel Formats

The DSI Host Controller Core and the external DSI Peripheral explicitly handle and exchanges 32-bit words encapsulated as packet data. The mapping between pixel formats and pixel data must be done on both sides by user application on both sides of the DSI link. The DSI spec gives many examples of common pixel formats. The DSI Host Controller Core and external DSI Peripheral have no preference or limitation on which pixel formats it will handle. The IP which provides video source packet data through DPI-2 interface (e.g. LCDIF) onto DSI should also be configured by the user application to match the pixel formats expected by the DSI Host Controller.

The DPI-2 interface core handles pixel to byte packing and byte to pixel unpacking according to the MIPI DSI specification.

45.9 MIPI DSI Host Memory Map/Register Definition

MIPI_DSI_HOST memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40A9_0000	MIPI_DSI_HOST0_DSI_HOST_CFG_NUM_LANES	32	R	0000_0000h	45.9.1/1886
40A9_0004	MIPI_DSI_HOST0_DSI_HOST_CFG_NONCONTINUOUS_CLK	32	R	0000_0000h	45.9.2/1887
40A9_0008	MIPI_DSI_HOST0_DSI_HOST_CFG_T_PRE	32	R	0000_0000h	45.9.3/1888
40A9_000C	MIPI_DSI_HOST0_DSI_HOST_CFG_T_POST	32	R	0000_0000h	45.9.4/1888
40A9_0010	MIPI_DSI_HOST0_DSI_HOST_CFG_TX_GAP	32	R	0000_0000h	45.9.5/1889
40A9_0014	MIPI_DSI_HOST0_DSI_HOST_CFG_AUTOINSERT_EOTP	32	R	0000_0000h	45.9.6/1890
40A9_0018	MIPI_DSI_HOST0_DSI_HOST_CFG_EXTRA_CMDS_AFTER_EOTP	32	R	0000_0000h	45.9.7/1890
40A9_001C	MIPI_DSI_HOST0_DSI_HOST_CFG_HTX_TO_COUNT	32	R	0000_0000h	45.9.8/1891
40A9_0020	MIPI_DSI_HOST0_DSI_HOST_CFG_LRX_H_TO_COUNT	32	R	0000_0000h	45.9.9/1891
40A9_0024	MIPI_DSI_HOST0_DSI_HOST_CFG_BTA_H_TO_COUNT	32	R	0000_0000h	45.9.10/ 1892
40A9_002C	MIPI_DSI_HOST0_DSI_HOST_CFG_STATUS_OUT	32	R	0000_0000h	45.9.11/ 1893
40A9_0030	MIPI_DSI_HOST0_DSI_HOST_RX_ERROR_STATUS	32	R	0000_0000h	45.9.12/ 1895

45.9.1 MIPI_DSI_HOSTx_DSI_HOST_CFG_NUM_LANES

Address: 40A9_0000h base + 0h offset = 40A9_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														dsi_host_cfg_num_lanes	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOSTx_DSI_HOST_CFG_NUM_LANES field descriptions

Field	Description
31–2 -	This field is reserved.

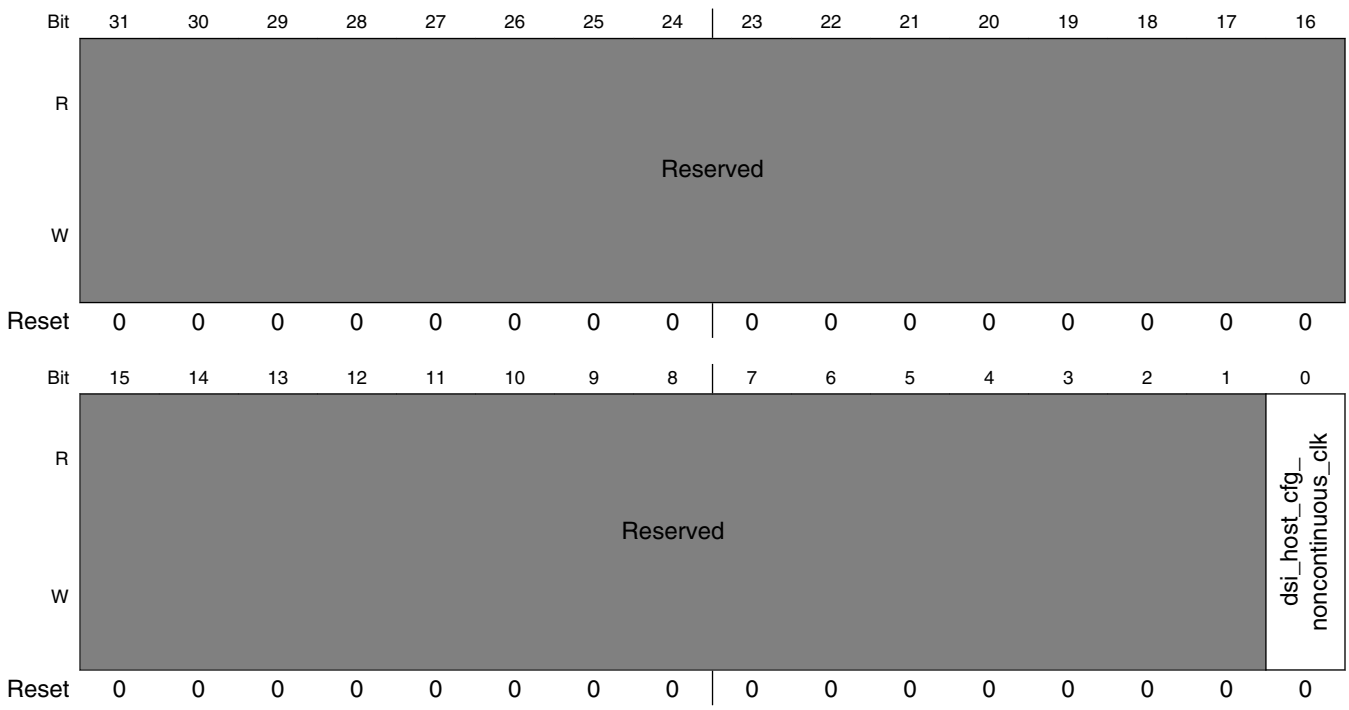
Table continues on the next page...

MIPI_DSI_HOSTx_DSI_HOST_CFG_NUM_LANES field descriptions (continued)

Field	Description
dsi_host_cfg_ num_lanes	Sets the number of active lanes that are to be used for transmitting data. • 2'b00 - 1 Lane • 2'b01 - 2 Lanes • 2'b10 - Reserved • 2'b11 - Reserved

45.9.2 MIPI_DSI_HOSTx_DSI_HOST_CFG_NONCONTINUOUS_CLK

Address: 40A9_0000h base + 4h offset = 40A9_0004h



MIPI_DSI_HOSTx_DSI_HOST_CFG_NONCONTINUOUS_CLK field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_cfg_ noncontinuous_clk	Sets the Host Controller into non-continuous MIPI clock mode. When in non-continuous clock mode, the high speed clock will transistion into low power mode between transmissions. • 1'b0 - Continuous high speed clock • 1'b1 - Non-Continuous high speed clock

45.9.3 MIPI_DSI_HOSTx_DSI_HOST_CFG_T_PRE

NOTE

Please refer to the MIPI Alliance Specification for D-PHY (Global Operation Timing Parameters) for details on calculating the parameter described below

Address: 40A9_0000h base + 8h offset = 40A9_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																									dsi_host_cfg_t_pre							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOSTx_DSI_HOST_CFG_T_PRE field descriptions

Field	Description
31–7 -	This field is reserved.
dsi_host_cfg_t_pre	Sets the number of byte clock periods ('clk_byte' input) that the controller will wait after enabling the clock lane for HS operation before enabling the data lanes for HS operation. This setting represents the TCLK-PRE parameter. The minimum value for this port is 1.

45.9.4 MIPI_DSI_HOSTx_DSI_HOST_CFG_T_POST

NOTE

Please refer to the MIPI Alliance Specification for D-PHY (Global Operation Timing Parameters) for details on calculating the parameter described below

Address: 40A9_0000h base + Ch offset = 40A9_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																									dsi_host_cfg_t_post							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOSTx_DSI_HOST_CFG_T_POST field descriptions

Field	Description
31–7 -	This field is reserved.

Table continues on the next page...

MIPI_DSI_HOSTx_DSI_HOST_CFG_T_POST field descriptions (continued)

Field	Description
dsi_host_cfg_t_post	Sets the number of byte clock periods ('clk_byte' input) to wait before putting the clock lane into LP mode after the data lanes have been detected to be in Stop State. This setting represents the DPHY timing parameters TLPX (TxClkEsc) + TCLK-PREPARE + TCLK-ZERO + TCLK-PRE requirement for the clock lane before the data lane is allowed to change from LP11 to start a high speed transmission. The minimum value for this port is 1.

45.9.5 MIPI_DSI_HOSTx_DSI_HOST_CFG_TX_GAP

NOTE

Please refer to the MIPI Alliance Specification for D-PHY (Global Operation Timing Parameters) for details on calculating the parameter described below

Address: 40A9_0000h base + 10h offset = 40A9_0010h

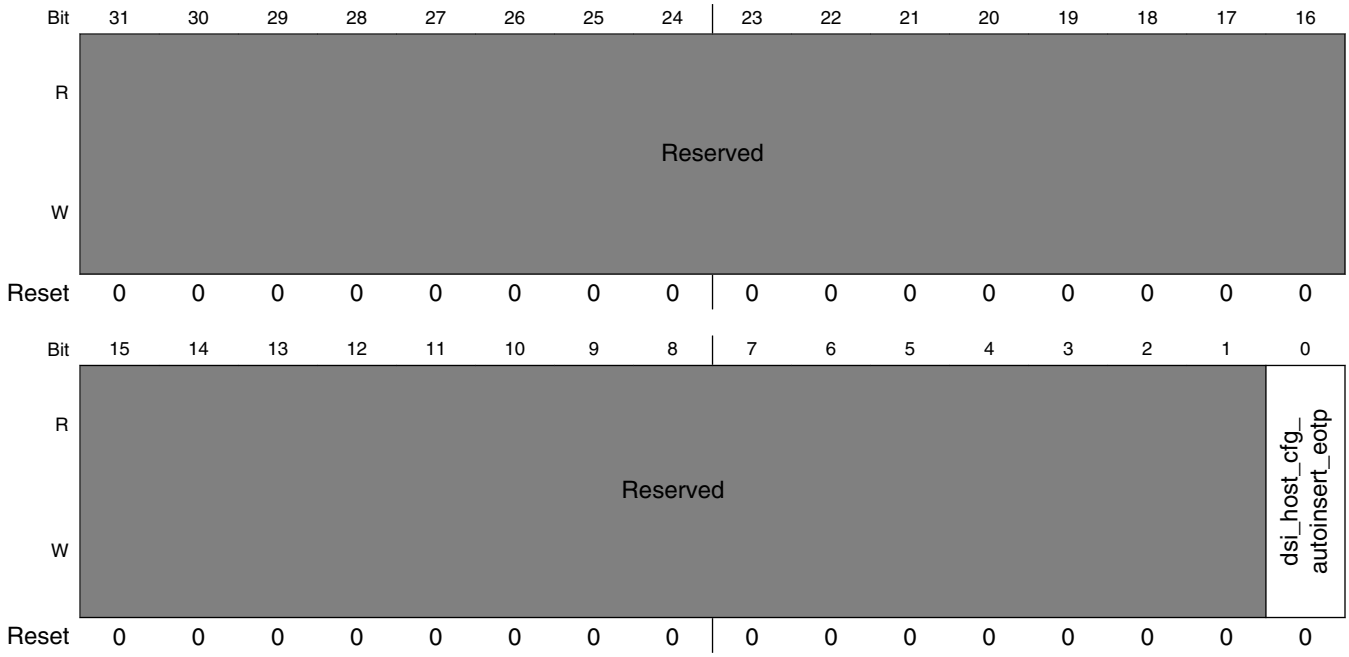
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																									dsi_host_cfg_tx_gap							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOSTx_DSI_HOST_CFG_TX_GAP field descriptions

Field	Description
31–7 -	This field is reserved.
dsi_host_cfg_tx_gap	Sets the number of byte clock periods ('clk_byte' input) that the controller will wait after the clock lane has been put into LP mode before enabling the clock lane for HS mode again. This setting represents the THS-EXIT parameter. The minimum value for this port is 1.

45.9.6 MIPI_DSI_HOSTx_DSI_HOST_CFG_AUTOINSERT_EOTP

Address: 40A9_0000h base + 14h offset = 40A9_0014h

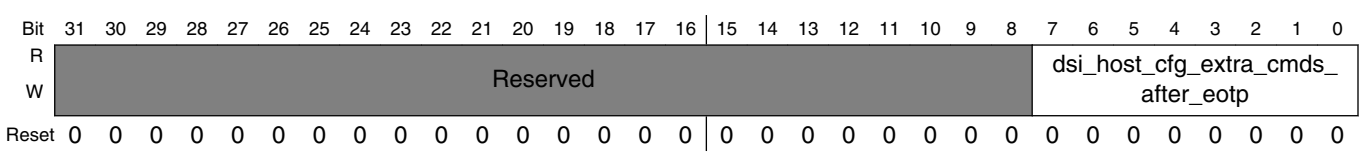


MIPI_DSI_HOSTx_DSI_HOST_CFG_AUTOINSERT_EOTP field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_cfg_ autoinsert_eotp	Enables the Host Controller to automatically insert an EoTp short packet when switching from HS to LP mode. • 1'b0 - EoTp is not automatically inserted • 1'b1 - EoTp is automatically inserted

45.9.7 MIPI_DSI_HOSTx_DSI_HOST_CFG_EXTRA_CMDS_AFTER_EOTP

Address: 40A9_0000h base + 18h offset = 40A9_0018h

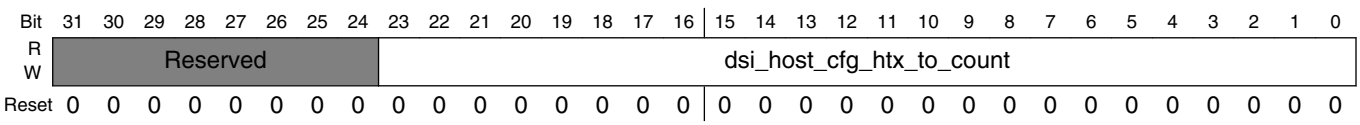


MIPI_DSI_HOSTx_DSI_HOST_CFG_EXTRA_CMDS_AFTER_EOTP field descriptions

Field	Description
31–8 -	This field is reserved.
dsi_host_cfg_ extra_cmds_ after_eotp	Configures the DSI Host Controller to send extra End Of Transmission Packets after the end of a packet. The value is the number of extra EOTP packets sent.

45.9.8 MIPI_DSI_HOSTx_DSI_HOST_CFG_HTX_TO_COUNT

Address: 40A9_0000h base + 1Ch offset = 40A9_001Ch

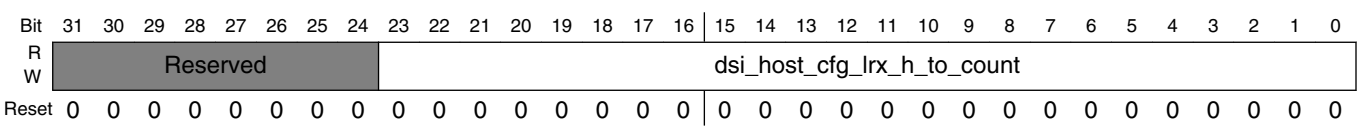


MIPI_DSI_HOSTx_DSI_HOST_CFG_HTX_TO_COUNT field descriptions

Field	Description
31–24 -	This field is reserved.
dsi_host_cfg_ htx_to_count	Sets the value of the DSI Host High Speed TX timeout count in clk_byte clock periods that once reached will initiate a timeout error and follow the recovery procedure documented in the DSI specification. This timeout parameter should be configured to represent the time taken to transmit the biggest HS data payload. If this timeout is reached the DSI byte count is cleared and the HS transmission is aborted. This timer can be also disabled, when set to 0.

45.9.9 MIPI_DSI_HOSTx_DSI_HOST_CFG_LRX_H_TO_COUNT

Address: 40A9_0000h base + 20h offset = 40A9_0020h



MIPI_DSI_HOSTx_DSI_HOST_CFG_LRX_H_TO_COUNT field descriptions

Field	Description
31–24 -	This field is reserved.
dsi_host_cfg_lrx_ h_to_count	Sets the value of the DSI Host low power RX timeout count in clk_byte clock periods that once reached will initiate a timeout error and follow the recovery procedure documented in the DSI specification. This timeout parameter should be configured to represent the time taken to receive the biggest LP (Escape mode) data payload. If this timeout is reached, the DSI byte count is cleared and the LP reception is aborted. This timer can be also disabled, when set to 0

45.9.10 MIPI_DSI_HOSTx_DSI_HOST_CFG_BTA_H_TO_COUNT

Address: 40A9_0000h base + 24h offset = 40A9_0024h

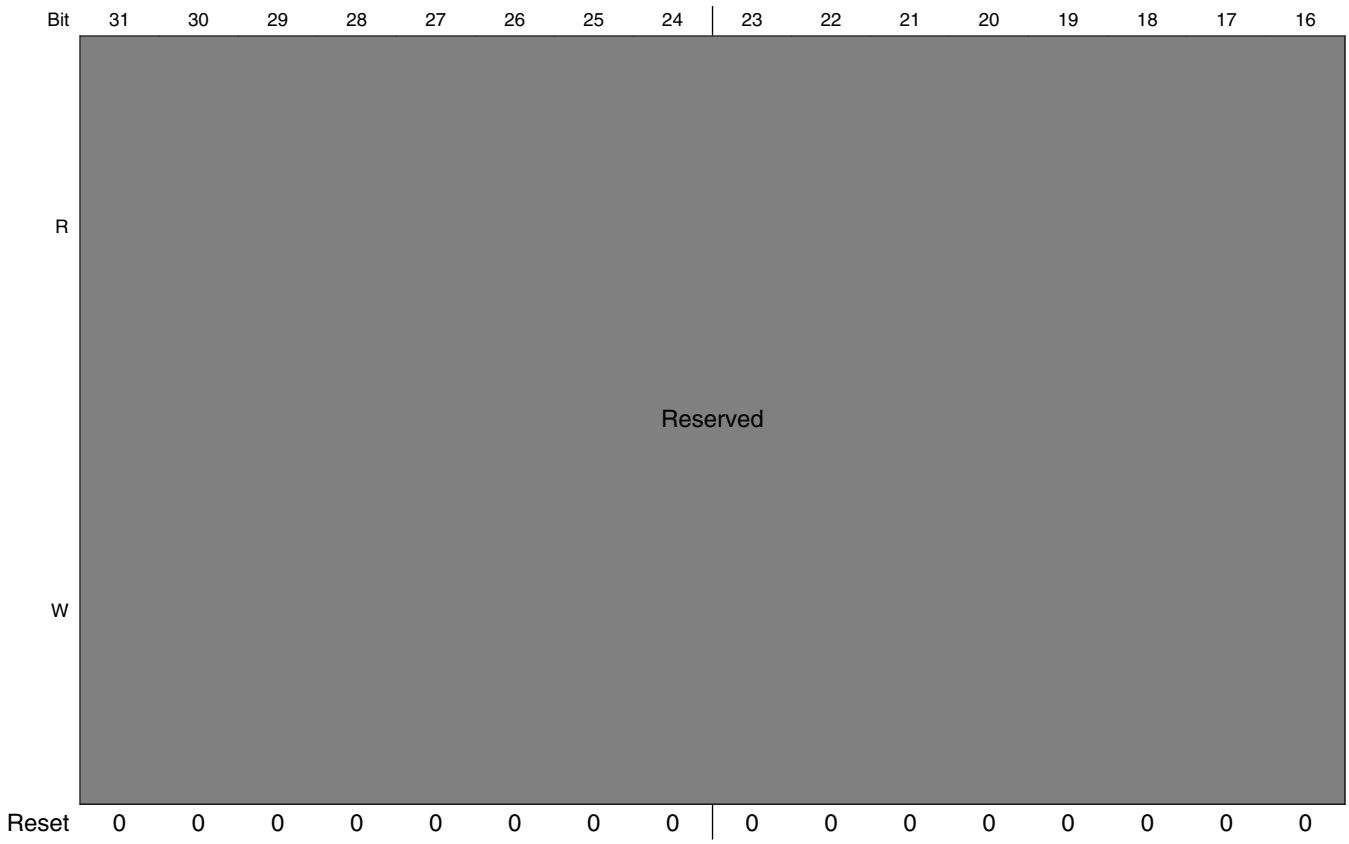
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								dsi_host_cfg_bta_h_to_count																									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

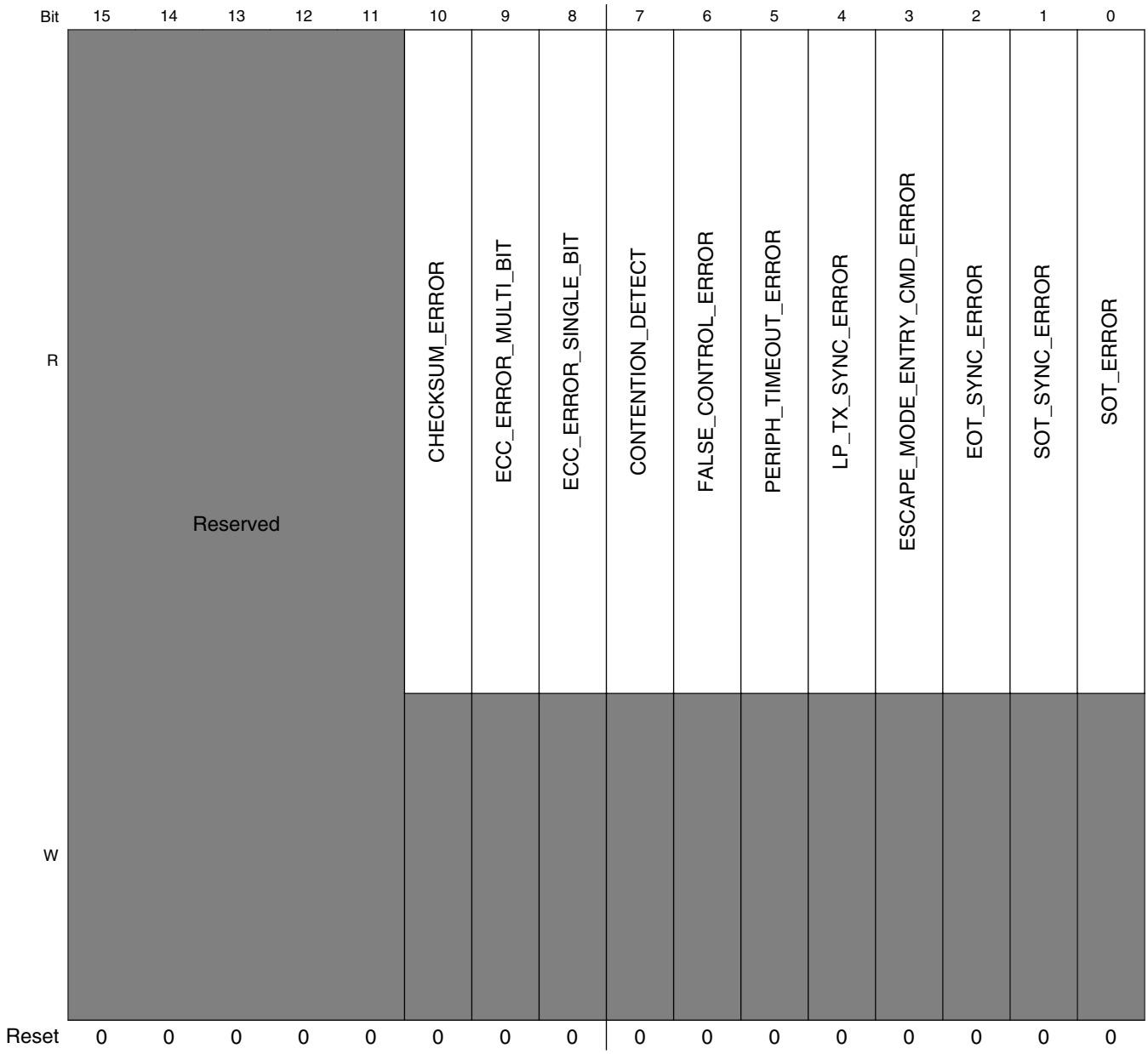
MIPI_DSI_HOSTx_DSI_HOST_CFG_BTA_H_TO_COUNT field descriptions

Field	Description
31–24 -	This field is reserved.
dsi_host_cfg_ bta_h_to_count	Sets the value of the DSI Host Bus Turn Around (BTA) timeout in clk_byte clock periods that once reached will initiate a timeout error.

45.9.11 MIPI_DSI_HOSTx_DSI_HOST_CFG_STATUS_OUT

Address: 40A9_0000h base + 2Ch offset = 40A9_002Ch





MIPI_DSI_HOSTx_DSI_HOST_CFG_STATUS_OUT field descriptions

Field	Description
31–11 -	This field is reserved. Reserved
10 CHECKSUM_ ERROR	Checksum Error (long packet only) – Checksum error from peripheral error report, cleared upon read
9 ECC_ERROR_ MULTI_BIT	ECC Error, multi-bit (detected, not corrected) – ECC multi-bit error from peripheral error report, cleared upon read

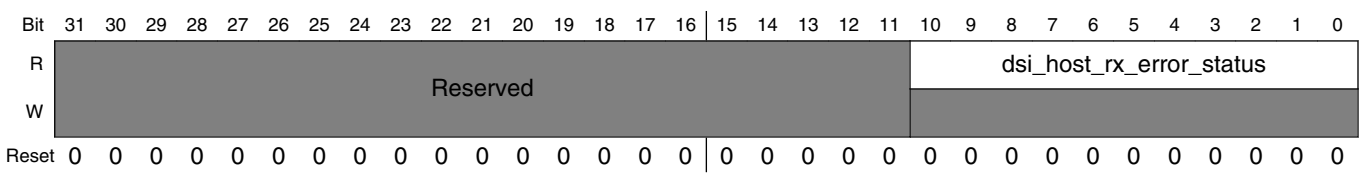
Table continues on the next page...

MIPI_DSI_HOSTx_DSI_HOST_CFG_STATUS_OUT field descriptions (continued)

Field	Description
8 ECC_ERROR_ SINGLE_BIT	ECC single bit error from peripheral error report, cleared upon read.
7 CONTENTION_ DETECT	Contention Detected – Contention Detection from peripheral error report, cleared upon read
6 FALSE_ CONTROL_ ERROR	False Control Error – False Control Error from peripheral error report, cleared upon read
5 PERIPH_ TIMEOUT_ ERROR	Peripheral Timeout Error – Peripheral Timeout error from peripheral error report, cleared upon read
4 LP_TX_SYNC_ ERROR	Low-Power Transmit Sync Error – Low Power Transmit Sync error from peripheral error report, cleared upon read.
3 ESCAPE_ MODE_ENTRY_ CMD_ERROR	Escape Mode Entry Command Error – Escape Mode Entry Command Error from peripheral error report, cleared upon read
2 EOT_SYNC_ ERROR	EoT Sync Error – End of Transmission (EoT) Sync Error from peripheral error report, cleared upon read
1 SOT_SYNC_ ERROR	SoT Sync Error – Start of Transmission (SoT) Sync Error from peripheral error report, cleared upon read
0 SOT_ERROR	SoT Error – Start of Transmission (SoT) Error from peripheral error report, cleared upon read.

45.9.12 MIPI_DSI_HOSTx_DSI_HOST_RX_ERROR_STATUS

Address: 40A9_0000h base + 30h offset = 40A9_0030h



MIPI_DSI_HOSTx_DSI_HOST_RX_ERROR_STATUS field descriptions

Field	Description
31–11 -	This field is reserved.
dsi_host_rx_ error_status	Status Register for Host receive error detection, ECC errors, CRC errors and for timeout indicators

Table continues on the next page...

MIPI_DSI_HOSTx_DSI_HOST_RX_ERROR_STATUS field descriptions (continued)

Field	Description
	• [0] ECC single bit error detected • [1] ECC multi bit error detected • [6:2] Errored bit position for single bit ECC error • [7] CRC error detected • [8] High Speed forward TX timeout detected • [9] Reverse Low power data receive timeout detected • [10] BTA timeout detected

45.10 MIPI DSI HOST DPI INTFC Memory Map/Register Definition

MIPI_DSI_HOST_DPI_INTFC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40A9_0200	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_PIXEL_PAYLOAD_SIZE	32	R	0000_0000h	45.10.1/ 1897
40A9_0204	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_PIXEL_FIFO_SEND_LEVEL	32	R	0000_0000h	45.10.2/ 1897
40A9_0208	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_INTERFACE_COLOR_CODING	32	R	0000_0000h	45.10.3/ 1898
40A9_020C	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_PIXEL_FORMAT	32	R	0000_0000h	45.10.4/ 1898
40A9_0210	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_VSYNC_POLARITY	32	R	0000_0000h	45.10.5/ 1899
40A9_0214	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_HSYNC_POLARITY	32	R	0000_0000h	45.10.6/ 1900
40A9_0218	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_VIDEO_MODE	32	R	0000_0000h	45.10.7/ 1901
40A9_021C	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_HFP	32	R	0000_0000h	45.10.8/ 1901
40A9_0220	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_HBP	32	R	0000_0000h	45.10.9/ 1901
40A9_0224	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_HSA	32	R	0000_0000h	45.10.10/ 1902
40A9_0228	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_ENABLE_MULT_PKTS	32	R	0000_0000h	45.10.11/ 1903
40A9_022C	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_VBP	32	R	0000_0000h	45.10.12/ 1903
40A9_0230	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_VFP	32	R	0000_0000h	45.10.13/ 1904
40A9_0234	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_BLLP_MODE	32	R	0000_0000h	45.10.14/ 1904

Table continues on the next page...

MIPI_DSI_HOST_DPI_INTFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40A9_0238	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_USE_NULL_PKT_BLLP	32	R	0000_0000h	45.10.15/ 1905
40A9_023C	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_VACTIVE	32	R	0000_0000h	45.10.16/ 1906
40A9_0240	MIPI_DSI_HOST_DPI_INTFC0_DSI_HOST_CFG_DPI_VC	32	R	0000_0000h	45.10.17/ 1906

45.10.1 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_PIXEL_PAYLOAD

Address: 40A9_0000h base + 200h offset = 40A9_0200h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																dsi_host_cfg_dpi_pixel_payload_size															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_PIXEL_PAYLOAD_SIZE field descriptions

Field	Description
31–16 -	This field is reserved.
dsi_host_cfg_dpi_pixel_payload_size	Maximum number of pixels that should be sent as one DSI packet. Recommended that the line size (in pixels) is evenly divisible by this parameter (packet payload size in pixels).

45.10.2 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_PIXEL_FIFO_SEN

Address: 40A9_0000h base + 204h offset = 40A9_0204h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																dsi_host_cfg_dpi_pixel_fifo_send_level															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_PIXEL_FIFO_SEND_LEVEL field descriptions

Field	Description
31–16 -	This field is reserved.
dsi_host_cfg_dpi_pixel_fifo_send_level	In order to optimize DSI utility, the DPI bridge buffers a cerntain number of DPI pixels before initiating a DSI packet. This configuration port controls the level at which the DPI Host bridge begins sending pixels.

45.10.3 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_INTERFACE_COL

Address: 40A9_0000h base + 208h offset = 40A9_0208h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													dsi_host_cfg_dpi_interface_color_coding		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_INTERFACE_COLOR_CODING field descriptions

Field	Description
31–3 -	This field is reserved.
dsi_host_cfg_dpi_interface_color_coding	Sets the distribution of RGB bits within the 24-bit d bus, as specified by the DPI specification. 0= 16-bit Configuration 1 1= 16-bit Configuration 2 2= 16-bit Configuration 3 3= 18-bit Configuration 1 4= 18-bit Configuration 2 5= 24-bit

45.10.4 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_PIXEL_FORMAT

Address: 40A9_0000h base + 20Ch offset = 40A9_020Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

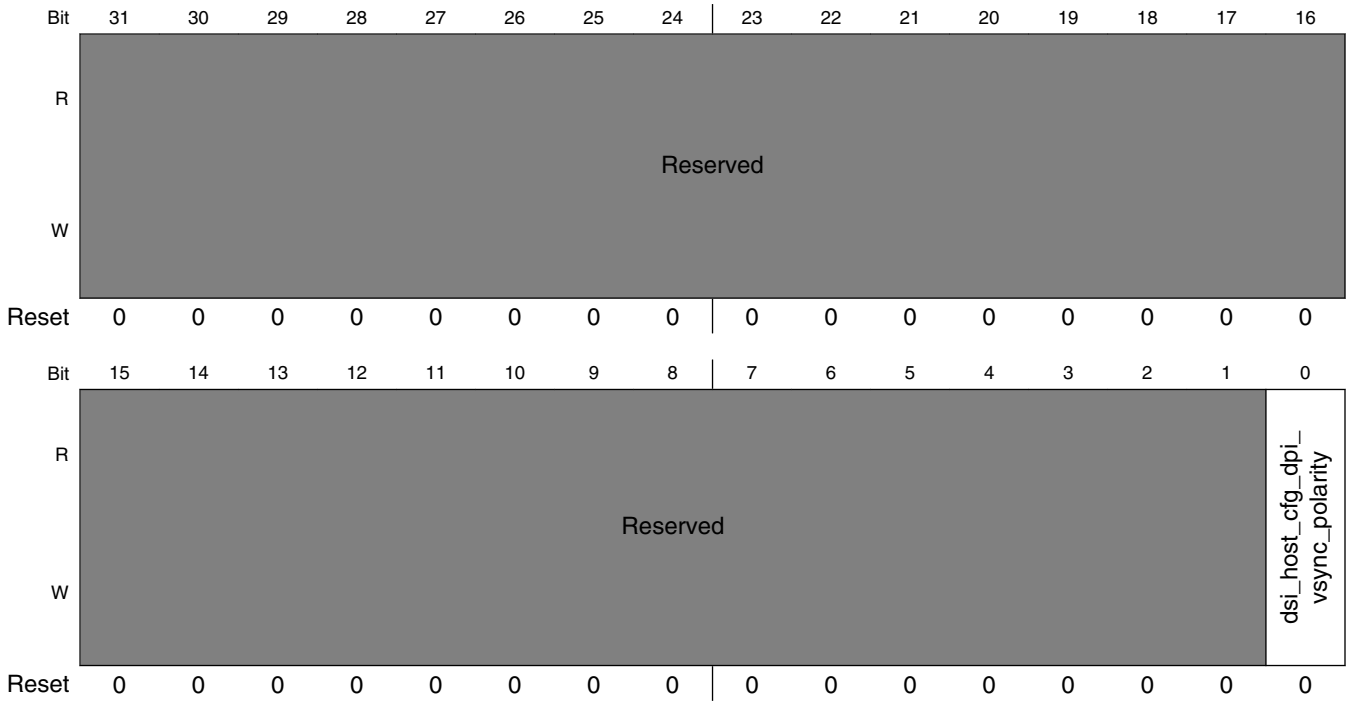
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														dsi_host_cfg_dpi_pixel_format	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_PIXEL_FORMAT field descriptions

Field	Description
31–2 -	This field is reserved.
dsi_host_cfg_dpi_pixel_format	Sets the DSI packet type of the pixels. 0 - 16 bit 1 - 18 bit 2 - 18 bit loosely packed 3 - 24 bit

45.10.5 MIPI_DSI_HOST_DPI_INTFC*x*_DSI_HOST_CFG_DPI_VSYNC_POLARIT

Address: 40A9_0000h base + 210h offset = 40A9_0210h

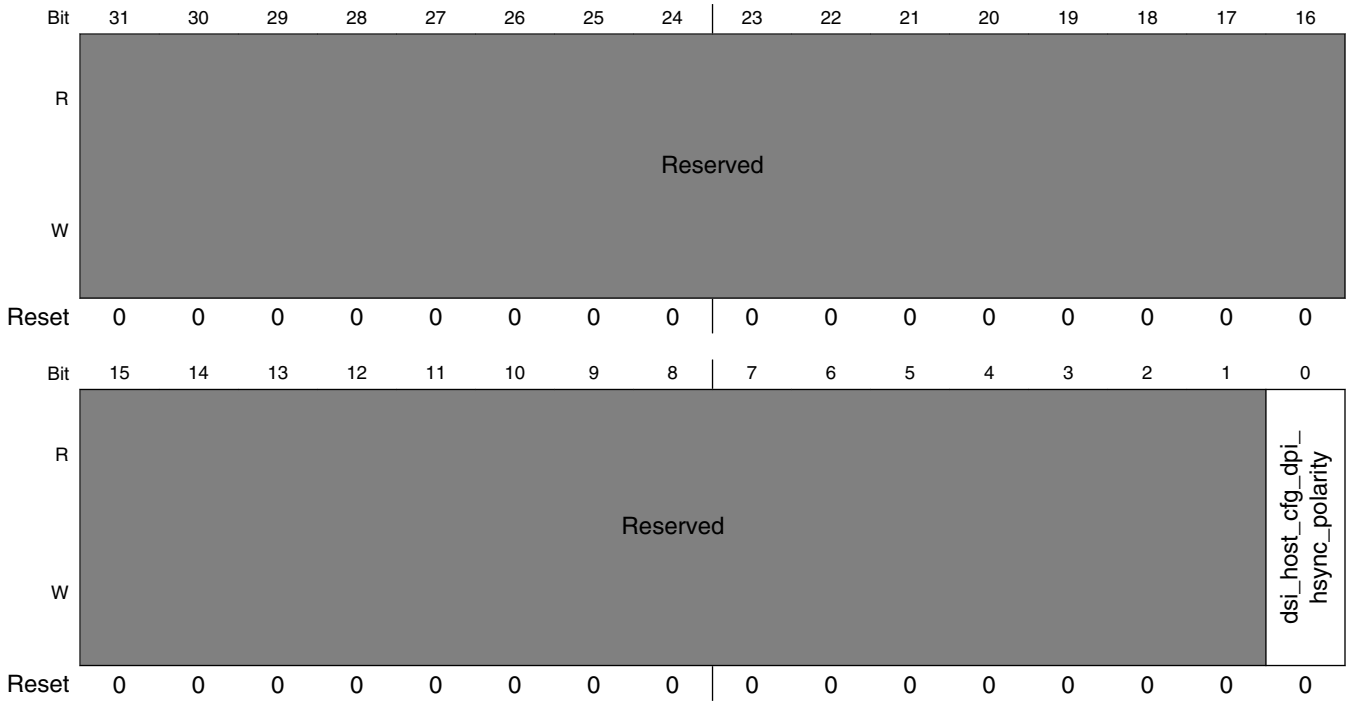


MIPI_DSI_HOST_DPI_INTFC*x*_DSI_HOST_CFG_DPI_VSYNC_POLARITY field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_cfg_dpi_vsync_polarity	Sets polarity of dpi_vsync_input 0 - active low 1 - active high

45.10.6 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HSYNC_POLARIT

Address: 40A9_0000h base + 214h offset = 40A9_0214h



MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HSYNC_POLARITY field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_cfg_dpi_hsync_polarity	Sets polarity of dpi_hsync_input 0 - active low 1 - active high

45.10.7 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VIDEO_MODE

Address: 40A9_0000h base + 218h offset = 40A9_0218h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														dsi_host_cfg_dpi_video_mode	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VIDEO_MODE field descriptions

Field	Description
31–2 -	This field is reserved.
dsi_host_cfg_dpi_video_mode	Select DSI video mode that the host DPI module should generate packets for. 2'b00 - Non-Burst mode with Sync Pulses 2'b01 - Non-Burst mode with Sync Events 2'b10 - Burst mode 2'b11 - Reserved, not valid

45.10.8 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HFP

Address: 40A9_0000h base + 21Ch offset = 40A9_021Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																dsi_host_cfg_dpi_hfp															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HFP field descriptions

Field	Description
31–16 -	This field is reserved.
dsi_host_cfg_dpi_hfp	Sets the DSI packet payload size, in bytes, of the horizontal front porch blanking packet.

45.10.9 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HBP

Address: 40A9_0000h base + 220h offset = 40A9_0220h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																dsi_host_cfg_dpi_hbp															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HBP field descriptions

Field	Description
31–16 -	This field is reserved.
dsi_host_cfg_ dpi_hbp	Sets the DSI packet payload size, in bytes, of the horizontal back porch blanking packet.

45.10.10 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HSA

Address: 40A9_0000h base + 224h offset = 40A9_0224h

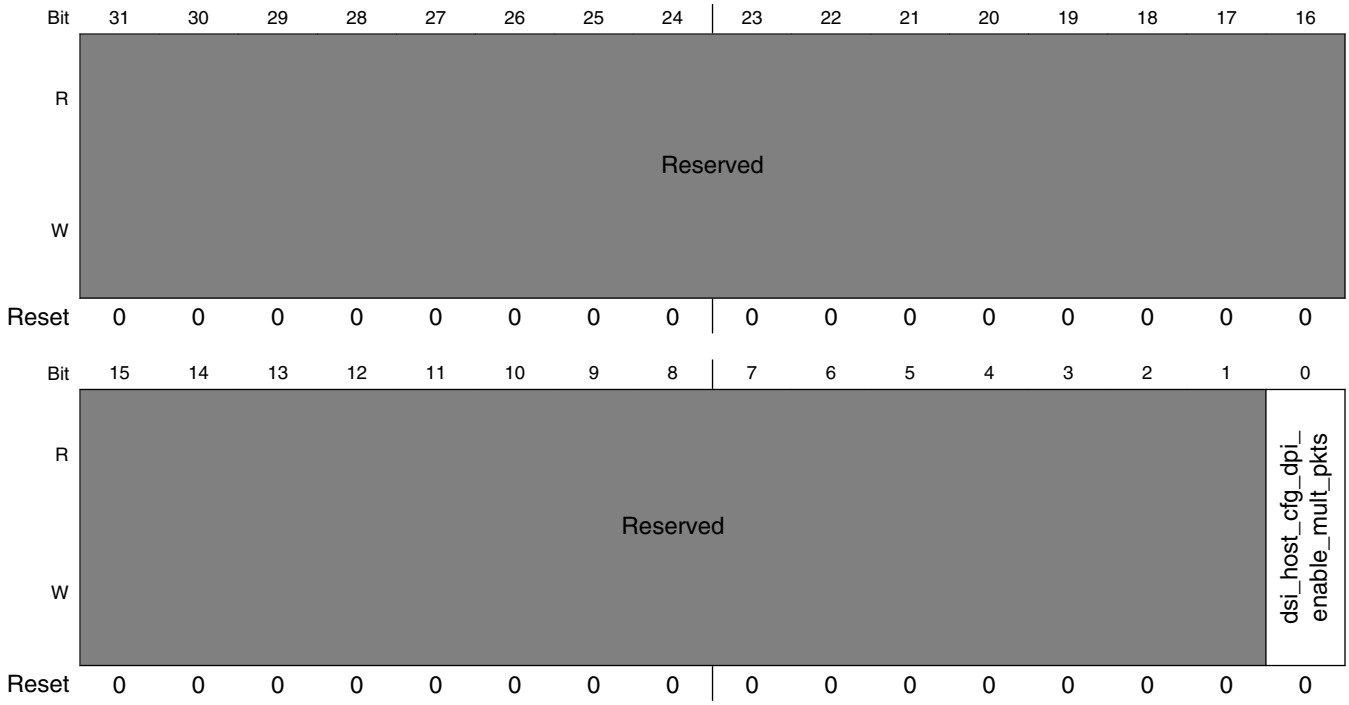
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																dsi_host_cfg_dpi_hsa															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_HSA field descriptions

Field	Description
31–16 -	This field is reserved.
dsi_host_cfg_ dpi_hsa	Sets the DSI packet payload size, in bytes, of the horizontal sync width filler blanking packet.

45.10.11 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_ENABLE_MULT

Address: 40A9_0000h base + 228h offset = 40A9_0228h

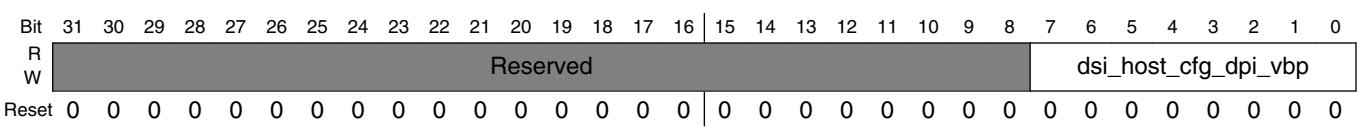


MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_ENABLE_MULT_PKTS field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_cfg_dpi_enable_mult_pkts	Enable Multiple packets per video line. When enabled, cfg_dpi_pixel_payload_size must be set to exactly half the size of the video line. 0 - Video Line is sent in a single packet 1 - Video Line is sent in two packets

45.10.12 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VBP

Address: 40A9_0000h base + 22Ch offset = 40A9_022Ch

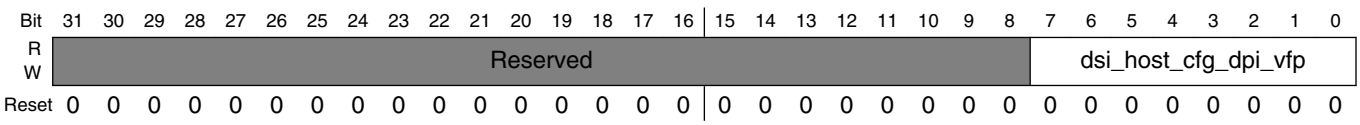


MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VBP field descriptions

Field	Description
31–8 -	This field is reserved.
dsi_host_cfg_ dpi_vbp	Sets the number of lines in the vertical back porch.

45.10.13 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VFP

Address: 40A9_0000h base + 230h offset = 40A9_0230h

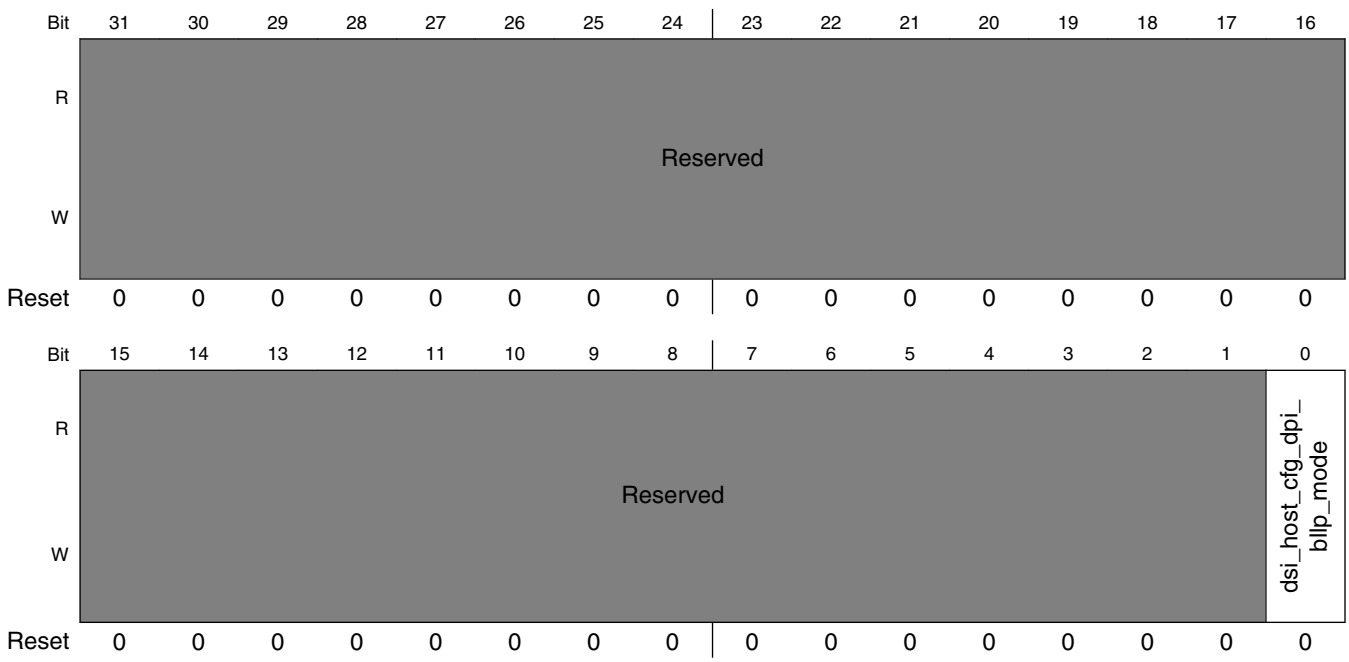


MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VFP field descriptions

Field	Description
31–8 -	This field is reserved.
dsi_host_cfg_ dpi_vfp	Sets the number of lines in the vertical front porch.

45.10.14 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_BLLP_MODE

Address: 40A9_0000h base + 234h offset = 40A9_0234h

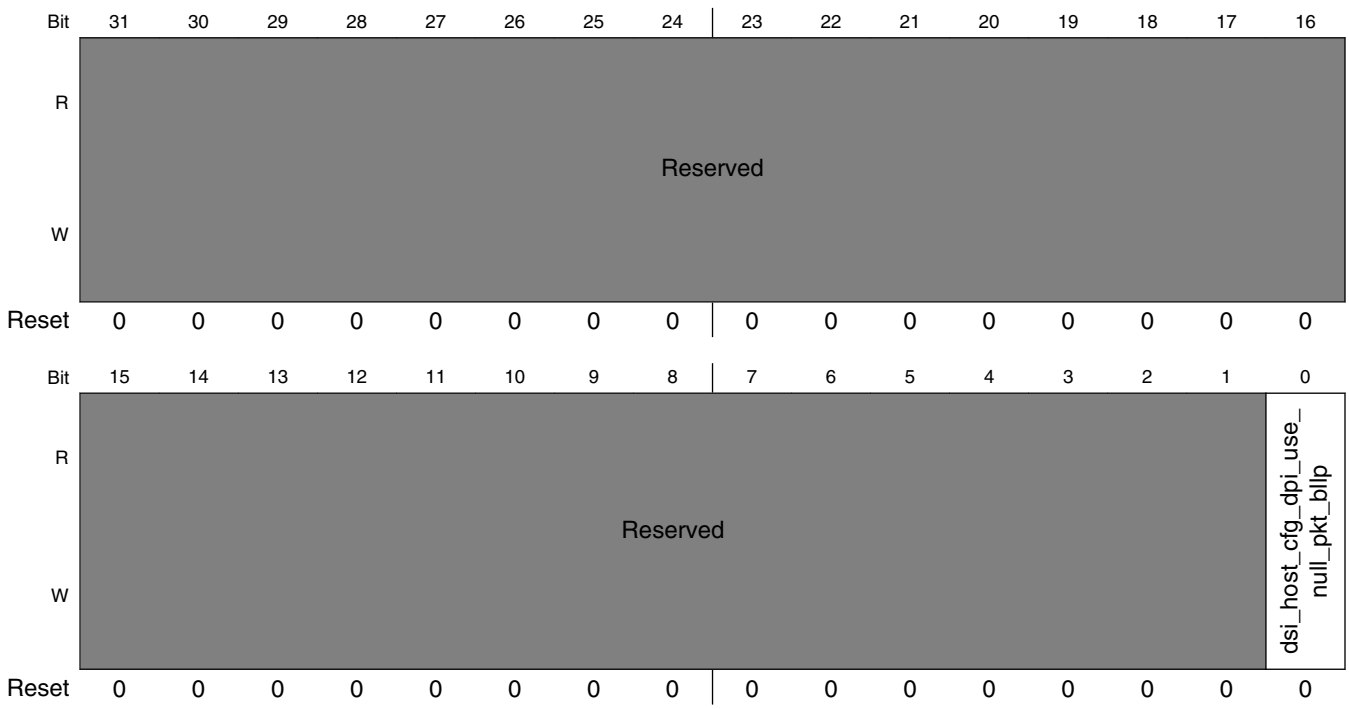


MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_BLLP_MODE field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_cfg_ dpi_bllp_mode	Optimize bllp periods to Low Power mode when possible 0 - blanking packets are sent during BLLP periods 1 - LP mode is used for BLLP periods

45.10.15 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_USE_NULL_PKT

Address: 40A9_0000h base + 238h offset = 40A9_0238h

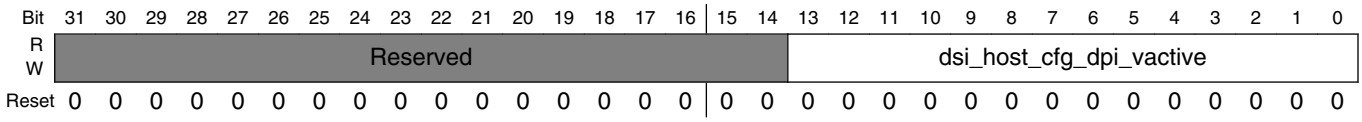


MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_USE_NULL_PKT_BLLP field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_cfg_ dpi_use_null_ pkt_bllp	Selects type of blanking packet to be sent during bllp region 0 - Blanking packet used in bllp region 1 - Null packet used in bllp region

45.10.16 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VACTIVE

Address: 40A9_0000h base + 23Ch offset = 40A9_023Ch

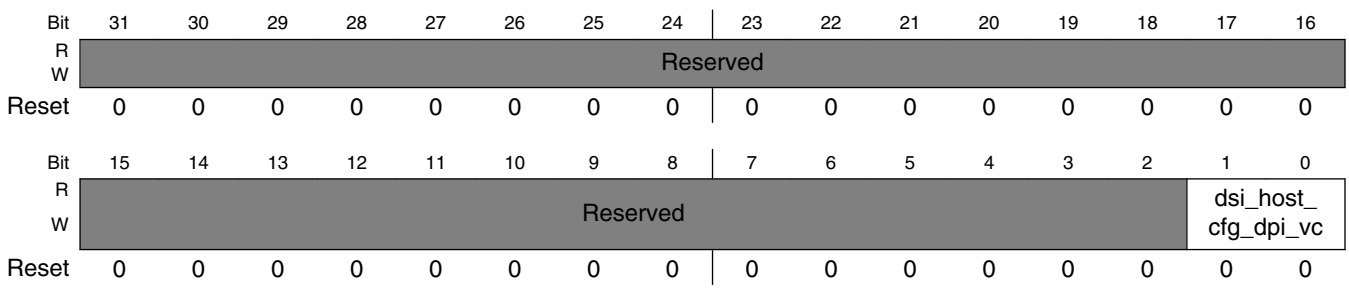


MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VACTIVE field descriptions

Field	Description
31–14 -	This field is reserved.
dsi_host_cfg_ dpi_vactive	Sets the number of lines in the vertical active area. This field is equivalent to (real vertical size) - 1. For example, for an image of size 640x480, the bit field should be set as 479.

45.10.17 MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VC

Address: 40A9_0000h base + 240h offset = 40A9_0240h



MIPI_DSI_HOST_DPI_INTFCx_DSI_HOST_CFG_DPI_VC field descriptions

Field	Description
31–2 -	This field is reserved.
dsi_host_cfg_ dpi_vc	Sets the Virtual Channel (VC) of packets that will be sent to the receive packet interface. Packets with VC not equal to this value are discarded and the "DSI VC ID Invalid" bit (bit 12) in the DSI error report is set.

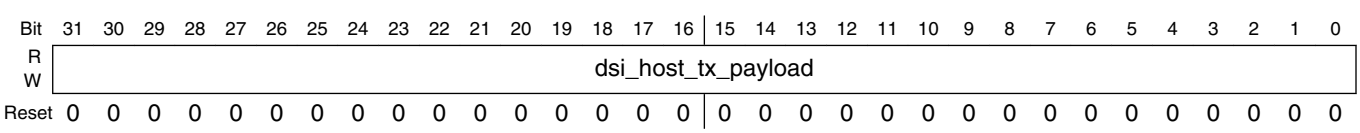
45.11 MIPI DSI Host APB PKT IF Memory Map/Register Definition

MIPI_DSI_HOST_APB_PKT_IF memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40A9_0280	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_TX_PAYLOAD	32	R/W	0000_0000h	45.11.1/ 1907
40A9_0284	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_PKT_CONTROL	32	R/W	0000_0000h	45.11.2/ 1908
40A9_0288	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_SEND_PACKET	32	R/W	0000_0000h	45.11.3/ 1908
40A9_028C	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_PKT_STATUS	32	R	0000_0000h	45.11.4/ 1909
40A9_0290	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_PKT_FIFO_WR_LEVEL	32	R	0000_0000h	45.11.5/ 1909
40A9_0294	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_PKT_FIFO_RD_LEVEL	32	R	0000_0000h	45.11.6/ 1910
40A9_0298	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_PKT_RX_PAYLOAD	32	R	0000_0000h	45.11.7/ 1910
40A9_029C	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_PKT_RX_PKT_HEADER	32	R	0000_0000h	45.11.8/ 1911
40A9_02A0	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_IRQ_STATUS	32	R	0000_0000h	45.11.9/ 1911
40A9_02A4	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_IRQ_STATUS2	32	R	0000_0000h	45.11.10/ 1912
40A9_02A8	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_IRQ_MASK	32	R/W	0000_0000h	45.11.11/ 1912
40A9_02AC	MIPI_DSI_HOST_APB_PKT_IF0_DSI_HOST_IRQ_MASK2	32	R/W	0000_0000h	45.11.12/ 1913

45.11.1 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_TX_PAYLOAD

Address: 40A9_0000h base + 280h offset = 40A9_0280h

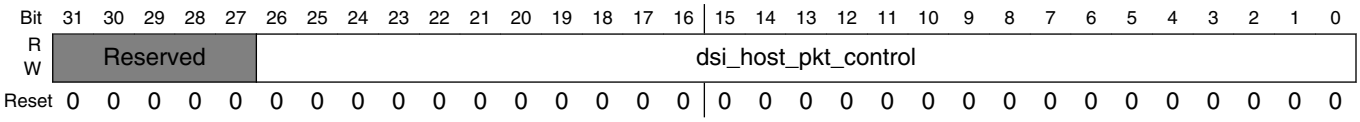


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_TX_PAYLOAD field descriptions

Field	Description
dsi_host_tx_payload	Tx Payload data write register. Writes to this registers load the payload fifo with 32 bit values.

45.11.2 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_CONTROL

Address: 40A9_0000h base + 284h offset = 40A9_0284h

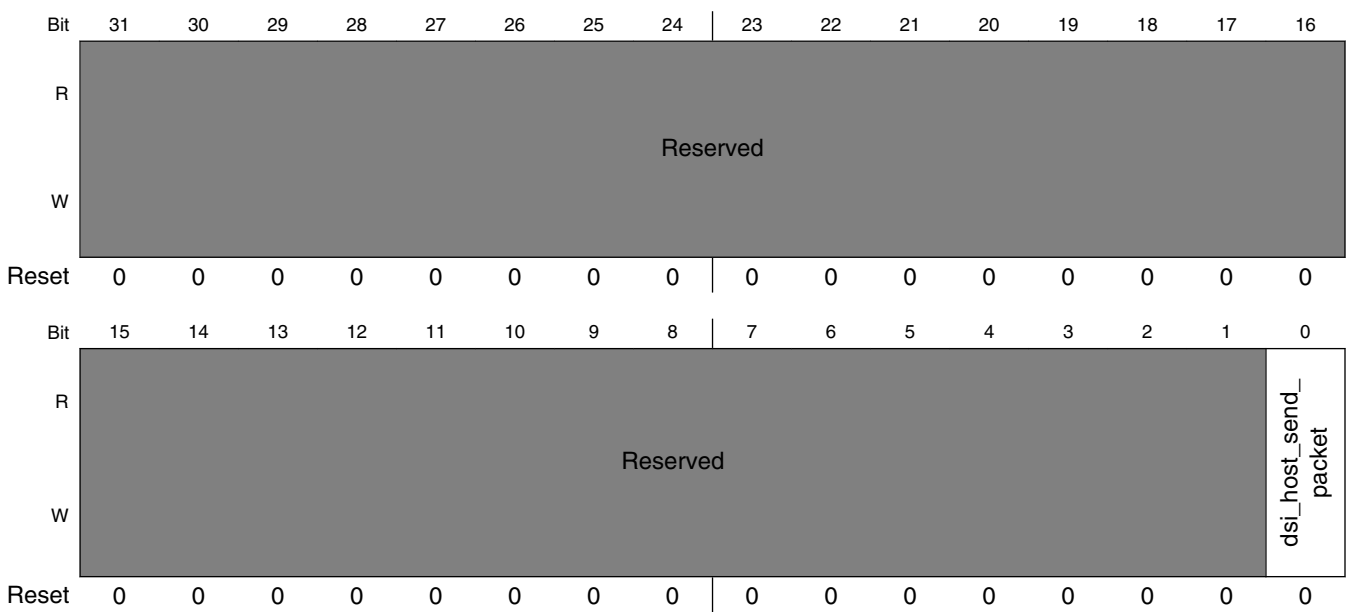


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_CONTROL field descriptions

Field	Description
31–27 -	This field is reserved.
dsi_host_pkt_ control	Tx packet control register. • [15:0] - Packet word count • [17:16] - Packet Virtual Channel • [23:18] - Packet Header DSI Data Type • [24] - Lp or HS select. 0 - LP mode, 1 - HS mode • [25] - perform BTA after packet is sent • [26] - perform BTA only, no packet tx

45.11.3 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_SEND_PACKET

Address: 40A9_0000h base + 288h offset = 40A9_0288h

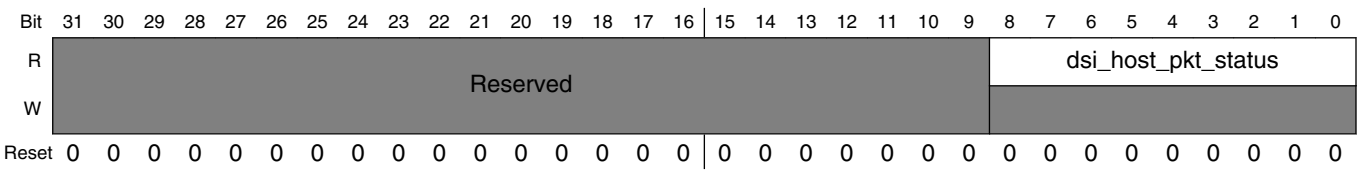


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_SEND_PACKET field descriptions

Field	Description
31–1 -	This field is reserved.
0 dsi_host_send_packet	Tx send packet. Writing to this register causes the packet described in dsi_host_pkt_control to be sent.

45.11.4 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_STATUS

Address: 40A9_0000h base + 28Ch offset = 40A9_028Ch

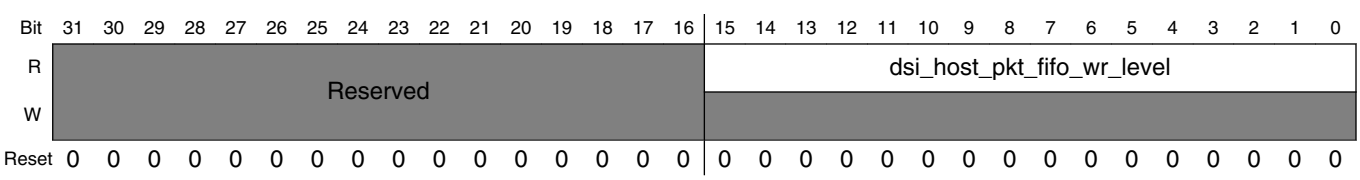


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_STATUS field descriptions

Field	Description
31–9 -	This field is reserved.
dsi_host_pkt_status	Status of APB to packet interface • [0] - state machine not idle • [1] - Tx packet done • [2] - dphy direction 0 - tx had control, 1 - rx has control • [3] - tx fifo overflow • [4] - tx fifo underflow • [5] - rx fifo overflow • [6] - rx fifo underflow • [7] - rx packet header has been received • [8] - all rx packet payload data has been receive d

45.11.5 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_FIFO_WR_LEVEL

Address: 40A9_0000h base + 290h offset = 40A9_0290h

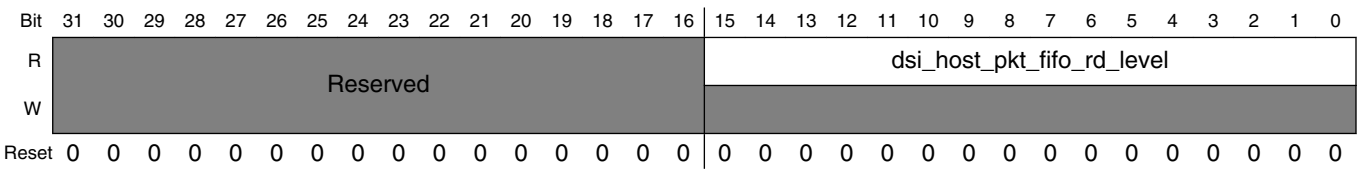


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_FIFO_WR_LEVEL field descriptions

Field	Description
31–16 -	This field is reserved.
dsi_host_pkt_ fifo_wr_level	Write level of APB to pkt interface fifo

45.11.6 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_FIFO_RD_LEVEL

Address: 40A9_0000h base + 294h offset = 40A9_0294h

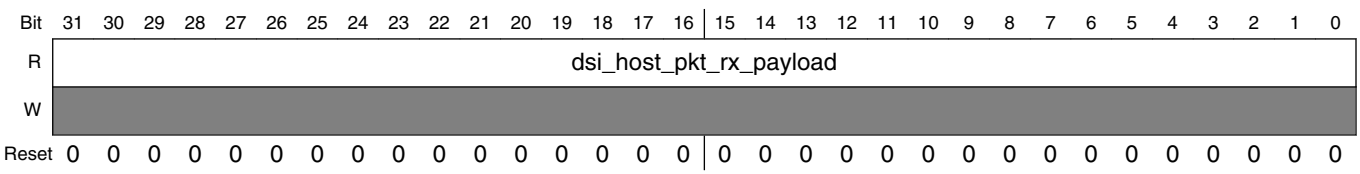


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_FIFO_RD_LEVEL field descriptions

Field	Description
31–16 -	This field is reserved.
dsi_host_pkt_ fifo_rd_level	Read level of APB to pkt interface fifo

45.11.7 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_RX_PAYLOAD

Address: 40A9_0000h base + 298h offset = 40A9_0298h

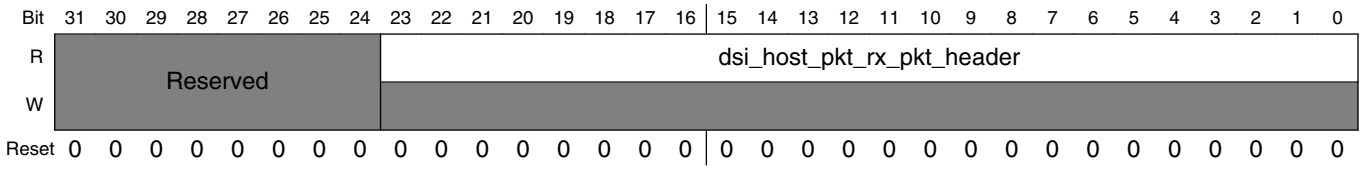


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_RX_PAYLOAD field descriptions

Field	Description
dsi_host_pkt_rx_ payload	APB to pkt interface rx payload read

45.11.8 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_RX_PKT_HEADER

Address: 40A9_0000h base + 29Ch offset = 40A9_029Ch

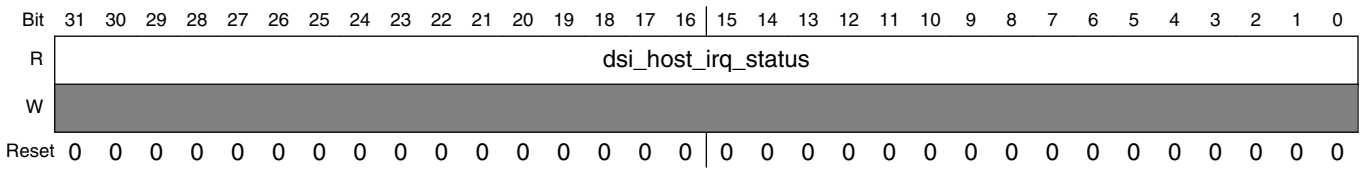


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_PKT_RX_PKT_HEADER field descriptions

Field	Description
31–24 -	This field is reserved.
dsi_host_pkt_rx_ pkt_header	APB to pkt interface rx packet header • [15:0] word count • [21:16] data type • [23:22] Virtual Channel

45.11.9 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_STATUS

Address: 40A9_0000h base + 2A0h offset = 40A9_02A0h

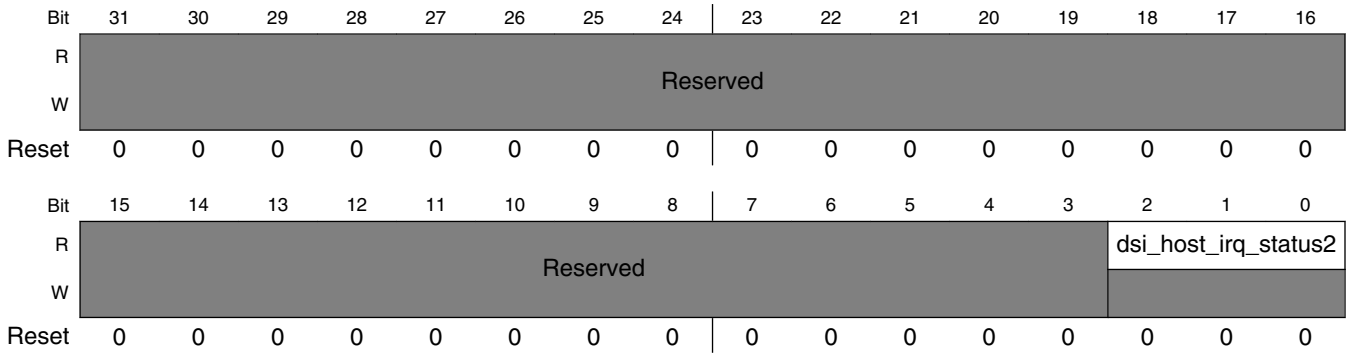


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_STATUS field descriptions

Field	Description
dsi_host_irq_ status	Status of APB to packet interface • [0] - state machine not idle • [1] - Tx packet done • [2] - dphy direction 0 - tx had control, 1 - rx has control • [3] - tx fifo overflow • [4] - tx fifo underflow • [5] - rx fifo overflow • [6] - rx fifo underflow • [7] - rx packet header has been received • [8] - all rx packet payload data has been received • [28:9] - map directory to dsi host controller status_out port bit descriptions • [29] - host bta timeout, host controller host_bta_timeout port • [30] - low power rx timeout, host controller lp_rx_timeout port • [31] - high speed tx timeout, host controller hs_tx_timeout port

45.11.10 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_STATUS2

Address: 40A9_0000h base + 2A4h offset = 40A9_02A4h

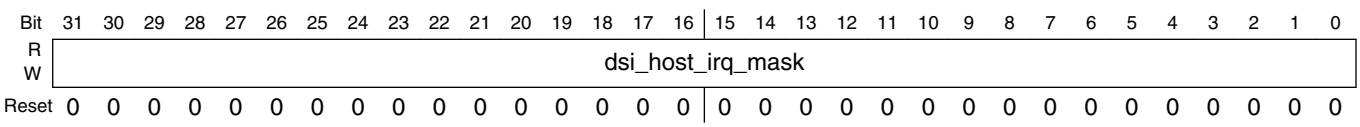


MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_STATUS2 field descriptions

Field	Description
31–3 -	This field is reserved.
dsi_host_irq_ status2	Status of APB to packet interface part 2. Read part 2 first then dsi_host_irq_status. Reading dsi_host_irq_status will clear both status and status 2. • [0] - single bit ecc error • [1] - multi bit ecc error • [2] - crc error

45.11.11 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_MASK

Address: 40A9_0000h base + 2A8h offset = 40A9_02A8h



MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_MASK field descriptions

Field	Description
dsi_host_irq_ mask	irq mask • [0] - state machine not idle • [1] - Tx packet done • [2] - dphy direction 0 - tx had control, 1 - rx has control • [3] - tx fifo overflow • [4] - tx fifo underflow • [5] - rx fifo overflow • [6] - rx fifo underflow • [7] - rx packet header has been received • [8] - all rx packet payload data has been received • [28:9] - map directory to dsi host controller status_out port bit descriptions

MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_MASK field descriptions (continued)

Field	Description
	• [29] - host bta timeout, host controller host_bta_timeout port • [30] - low power rx timeout, host controller lp_rx_timeout port • [31] - high speed tx timeout, host controller hs_tx_timeout port

45.11.12 MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_MASK2

Address: 40A9_0000h base + 2ACh offset = 40A9_02ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																													dsi_host_irq_mask2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_APB_PKT_IFx_DSI_HOST_IRQ_MASK2 field descriptions

Field	Description
31–3 -	This field is reserved.
dsi_host_irq_mask2	irq mask 2 • [0] - single bit ecc error • [1] - multi bit ecc error • [2] - crc error

45.12 MIPI DSI Host IP1 DPHY INTFC Memory Map/Register Definition

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40A9_0300	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_PD_DPHY	32	R	0000_0001h	45.12.1/ 1915
40A9_0304	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_M_PRG_HS_PREPARE	32	R	0000_0000h	45.12.2/ 1915
40A9_0308	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_MC_PRG_HS_PREPARE	32	R	0000_0000h	45.12.3/ 1916
40A9_030C	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_M_PRG_HS_ZERO	32	R	0000_0000h	45.12.4/ 1917
40A9_0310	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_MC_PRG_HS_ZERO	32	R	0000_0000h	45.12.5/ 1918

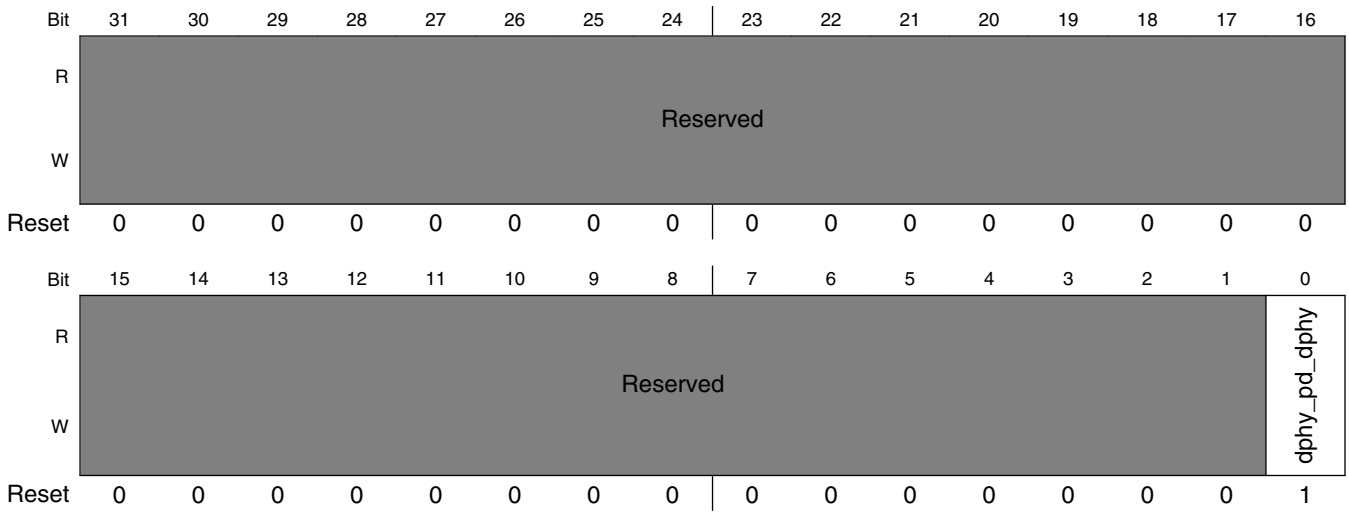
Table continues on the next page...

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40A9_0314	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_M_PRG_HS_TRAIL	32	R	0000_0000h	45.12.6/ 1919
40A9_0318	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_MC_PRG_HS_TRAIL	32	R	0000_0000h	45.12.7/ 1919
40A9_031C	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_PD_PLL	32	R	0000_0001h	45.12.8/ 1920
40A9_0320	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_TST	32	R	0000_0025h	45.12.9/ 1921
40A9_0324	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_CN	32	R	0000_0000h	45.12.10/ 1921
40A9_0328	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_CM	32	R	0000_0000h	45.12.11/ 1922
40A9_032C	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_CO	32	R	0000_0000h	45.12.12/ 1923
40A9_0330	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_LOCK	32	R	0000_0000h	45.12.13/ 1924
40A9_0334	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_LOCK_BYP	32	R	0000_0000h	45.12.14/ 1925
40A9_0338	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_RTERM_SEL	32	R	0000_0000h	45.12.15/ 1925
40A9_033C	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_AUTO_PD_EN	32	R	0000_0000h	45.12.16/ 1926
40A9_0340	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_RXLPRP	32	R	0000_0000h	45.12.17/ 1927
40A9_0344	MIPI_DSI_HOST_FSL_IP1_DPHY_INTFC0_DPHY_RXCDRP	32	R	0000_0000h	45.12.18/ 1927

45.12.1 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_PD_DPHY

Address: 40A9_0000h base + 300h offset = 40A9_0300h

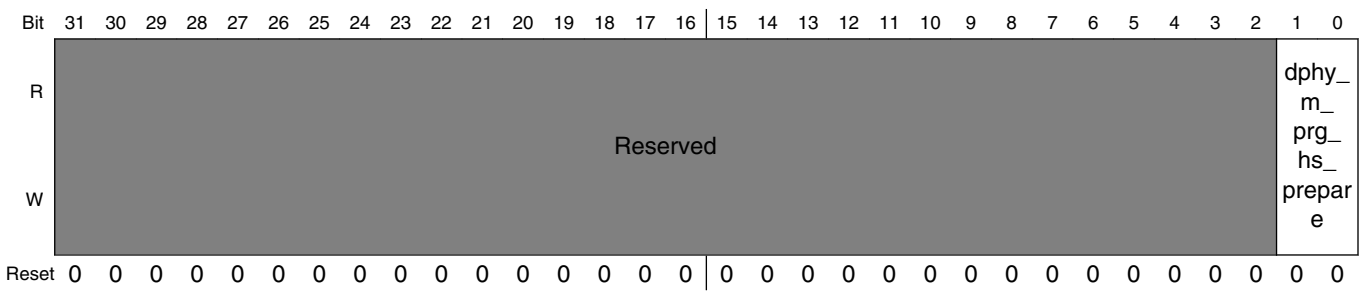


MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_PD_DPHY field descriptions

Field	Description
31–1 -	This field is reserved.
0 dphy_pd_dphy	DPHY PD_DPHY input control. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.2 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_M_PRG_HS_PREPA

Address: 40A9_0000h base + 304h offset = 40A9_0304h



MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_M_PRG_HS_PREPARE field descriptions

Field	Description
31–2 -	This field is reserved.

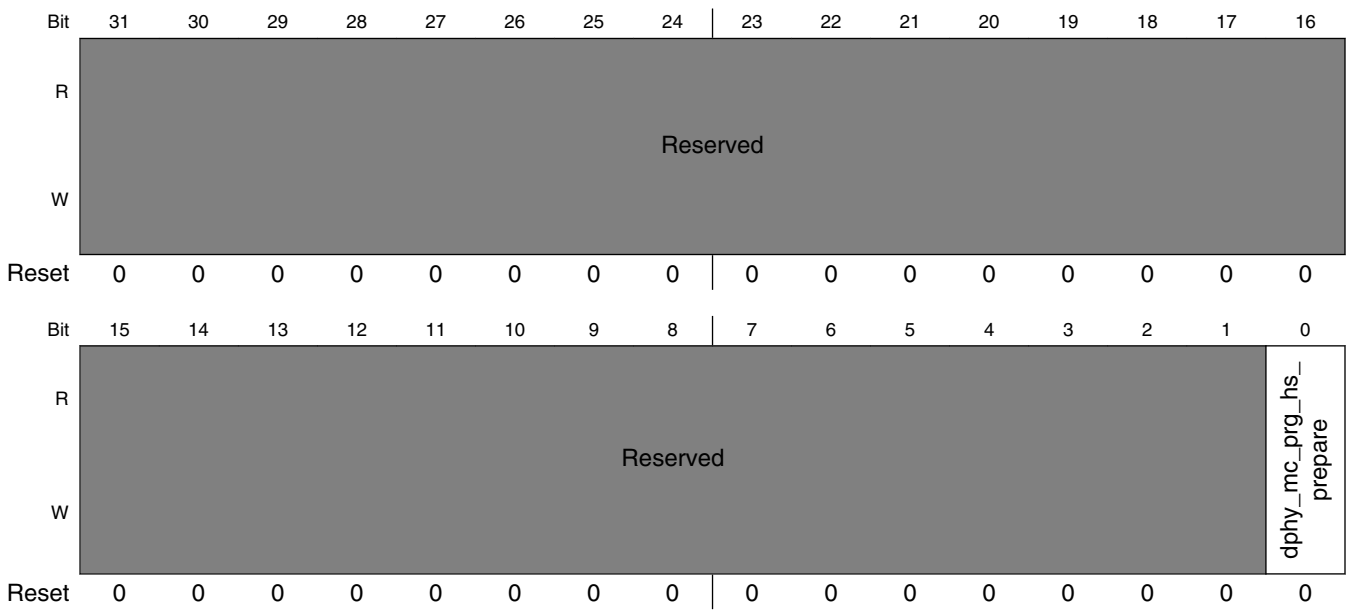
Table continues on the next page...

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_M_PRG_HS_PREPARE field descriptions (continued)

Field	Description
dphy_m_prg_hs_prepare	DPHY m_PRG_HS_PREPARE input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 00 1 01 1.5 10 2 11 2.5

45.12.3 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_MC_PRG_HS_PREP

Address: 40A9_0000h base + 308h offset = 40A9_0308h



MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_MC_PRG_HS_PREPARE field descriptions

Field	Description
31–1 -	This field is reserved.
0 dphy_mc_prg_hs_prepare	DPHY mc_PRG_HS_PREPARE input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 0 1 1 1.5

45.12.4 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_M_PRG_HS_ZERO

Address: 40A9_0000h base + 30Ch offset = 40A9_030Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																											dphy_m_prg_hs_zero					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_M_PRG_HS_ZERO field descriptions

Field	Description
31–5 -	This field is reserved.
dphy_m_prg_hs_zero	DPHY m_PRG_HS_ZERO input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 00000 0 00001 1 00010 2 00011 3 00100 4 00101 5 00110 6 00111 7 01000 8 01001 9 01010 10 01011 11 01100 12 01101 13 01110 14 01111 15 10000 16 10001 17 10010 18 10011 19 10100 20 10101 21 10110 22 10111 23 11000 24 11001 25 11010 26 11011 27 11100 28 11101 29 11110 30 11111 31

45.12.5 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_MC_PRG_HS_ZERO

Address: 40A9_0000h base + 310h offset = 40A9_0310h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																										dphy_mc_prg_hs_zero					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_MC_PRG_HS_ZERO field descriptions

Field	Description
31–6 -	This field is reserved.
dphy_mc_prg_hs_zero	DPHY mc_PRG_HS_ZERO input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 100000 32 100001 33 100010 34 100011 35 100100 36 100101 37 100110 38 100111 39 101000 40 101001 41 101010 42 101011 43 101100 44 101101 45 101110 46 101111 47 110000 48 110001 49 110010 50 110011 51 110100 52 110101 53 110110 54 110111 55 111000 56 111001 57 111010 58 111011 59 111100 60 111101 61 111110 62 111111 63

45.12.6 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_M_PRG_HS_TRAIL

Address: 40A9_0000h base + 314h offset = 40A9_0314h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																												dphy_m_prg_hs_trail			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_M_PRG_HS_TRAIL field descriptions

Field	Description
31–4 -	This field is reserved.
dphy_m_prg_hs_trail	DPHY m_PRG_HS_TRAIL input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 0000 0 0001 1 0010 2 0011 3 0100 4 0101 5 0110 6 0111 7 1000 8 1001 9 1010 10 1011 11 1100 12 1101 13 1110 14 1111 15

45.12.7 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_MC_PRG_HS_TRAIL

Address: 40A9_0000h base + 318h offset = 40A9_0318h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																												dphy_mc_prg_hs_trail			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_MC_PRG_HS_TRAIL field descriptions

Field	Description
31–4 -	This field is reserved.
dphy_mc_prg_hs_trail	DPHY mc_PRG_HS_TRAIL input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 0000 0 0001 1 0010 2 0011 3 0100 4 0101 5 0110 6 0111 7 1000 8 1001 9 1010 10 1011 11 1100 12 1101 13 1110 14 1111 15

45.12.8 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_PD_PLL

Address: 40A9_0000h base + 31Ch offset = 40A9_031Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															PD
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_PD_PLL field descriptions

Field	Description
31–1 -	This field is reserved.
0 PD	DPHY PD_PLL input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.9 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_TST

Address: 40A9_0000h base + 320h offset = 40A9_0320h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																										TST					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_TST field descriptions

Field	Description
31–6 -	This field is reserved.
TST	DPHY TST input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.10 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CN

Address: 40A9_0000h base + 324h offset = 40A9_0324h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																											CN				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CN field descriptions

Field	Description
31–5 -	This field is reserved.
CN	DPHY PLL Input Divider. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 11111 Divide by 1 00000 Divide by 2 10000 Divide by 3 11000 Divide by 4 11100 Divide by 5 01110 Divide by 6 00111 Divide by 7 10011 Divide by 8 01001 Divide by 9 00100 Divide by 10 00010 Divide by 11 10001 Divide by 12 01000 Divide by 13 10100 Divide by 14

Table continues on the next page...

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CN field descriptions (continued)

Field	Description
	01010 Divide by 15 10101 Divide by 16 11010 Divide by 17 11101 Divide by 18 11110 Divide by 19 01111 Divide by 20 10111 Divide by 21 11011 Divide by 22 01101 Divide by 23 10110 Divide by 24 01011 Divide by 25 00101 Divide by 26 10010 Divide by 27 11001 Divide by 28 01100 Divide by 29 00110 Divide by 30 00011 Divide by 31 00001 Divide by 32

45.12.11 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CM

Address: 40A9_0000h base + 328h offset = 40A9_0328h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																								CM							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CM field descriptions

Field	Description
31–8 -	This field is reserved.
CM	DPHY PLL Feedback Divider. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 111x0000 Divide by 16 : : 111x1111 Divide by 31 11000000 Divide by 32 : : 11011111 Divide by 63 10000000 Divide by 64 : : 10111111 Divide by 127 00000000 Divide by 128

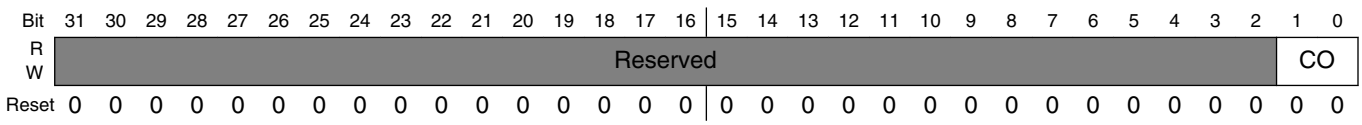
Table continues on the next page...

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CM field descriptions (continued)

Field	Description
	: : 01111111 Divide by 255

45.12.12 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CO

Address: 40A9_0000h base + 32Ch offset = 40A9_032Ch

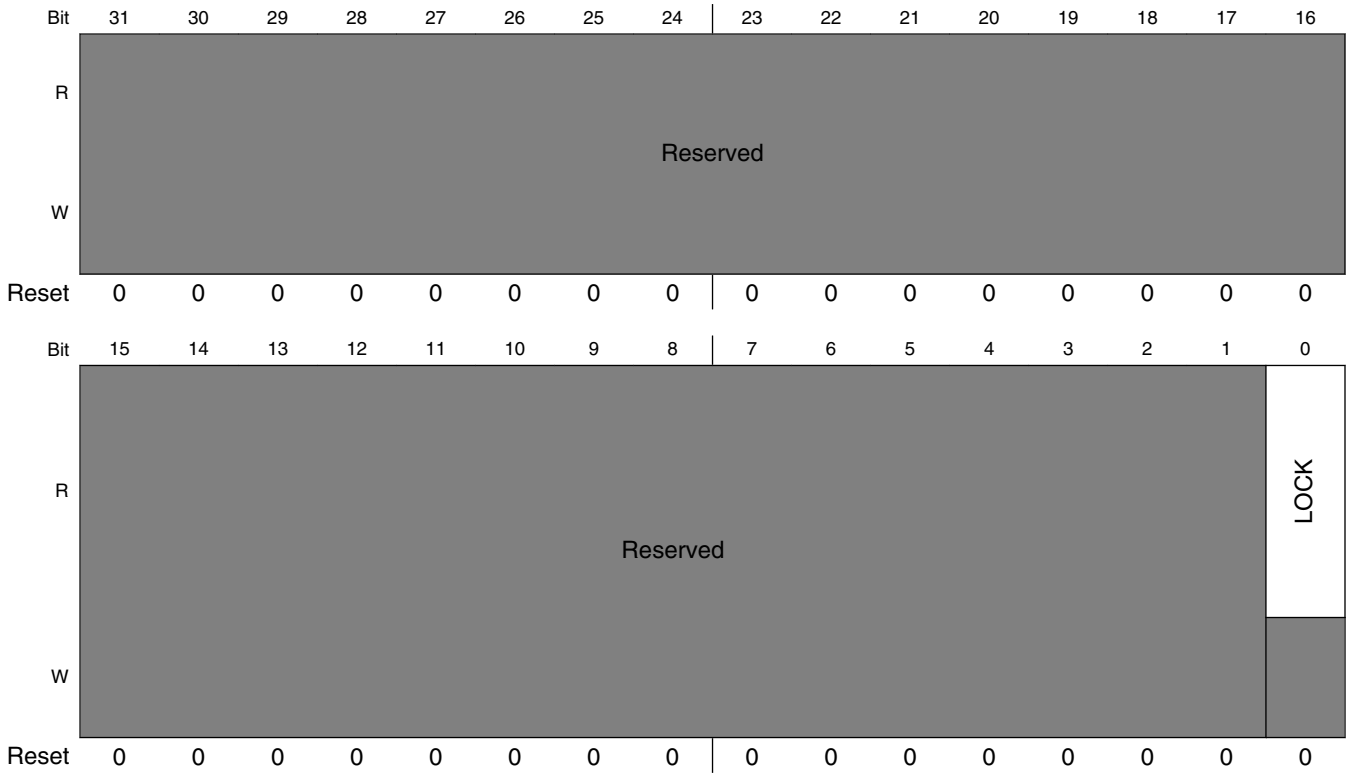


MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_CO field descriptions

Field	Description
31–2 -	This field is reserved.
CO	DPHY PLL Output Divider. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section. 00 Divide by 1 01 Divide by 2 10 Divide by 4 11 Divide by 8

45.12.13 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_LOCK

Address: 40A9_0000h base + 330h offset = 40A9_0330h

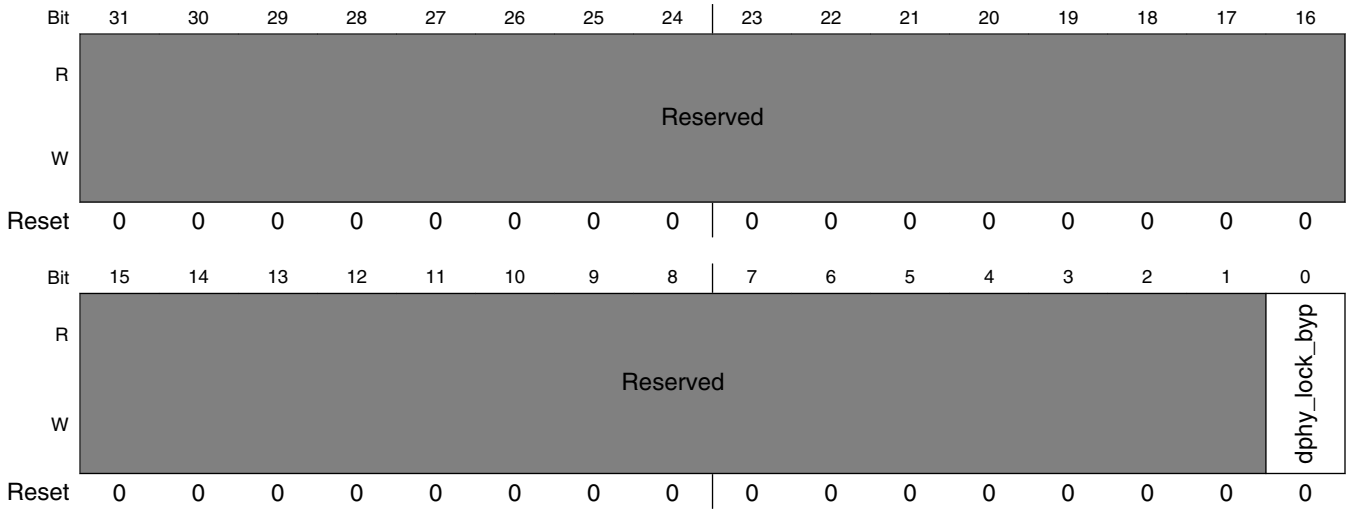


MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_LOCK field descriptions

Field	Description
31–1 -	This field is reserved.
0 LOCK	DPHY PLL LOCK output. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.14 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_LOCK_BYP

Address: 40A9_0000h base + 334h offset = 40A9_0334h

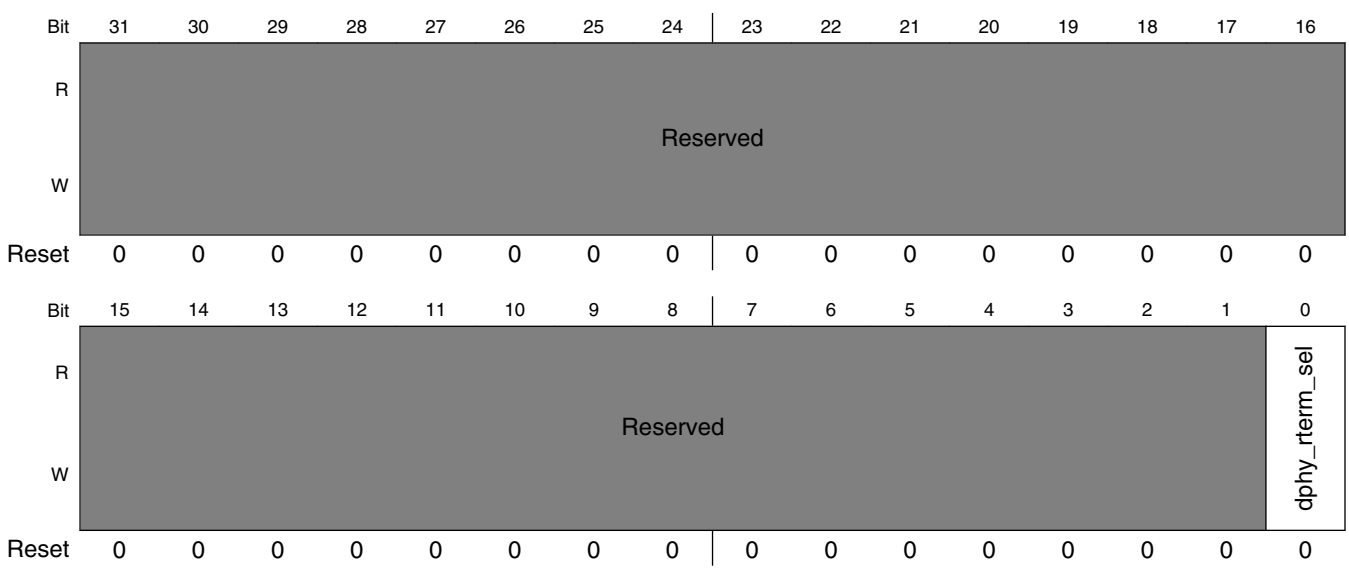


MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_LOCK_BYP field descriptions

Field	Description
31–1 -	This field is reserved.
0 dphy_lock_byp	DPHY LOCK_BYP input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.15 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_RTERM_SEL

Address: 40A9_0000h base + 338h offset = 40A9_0338h

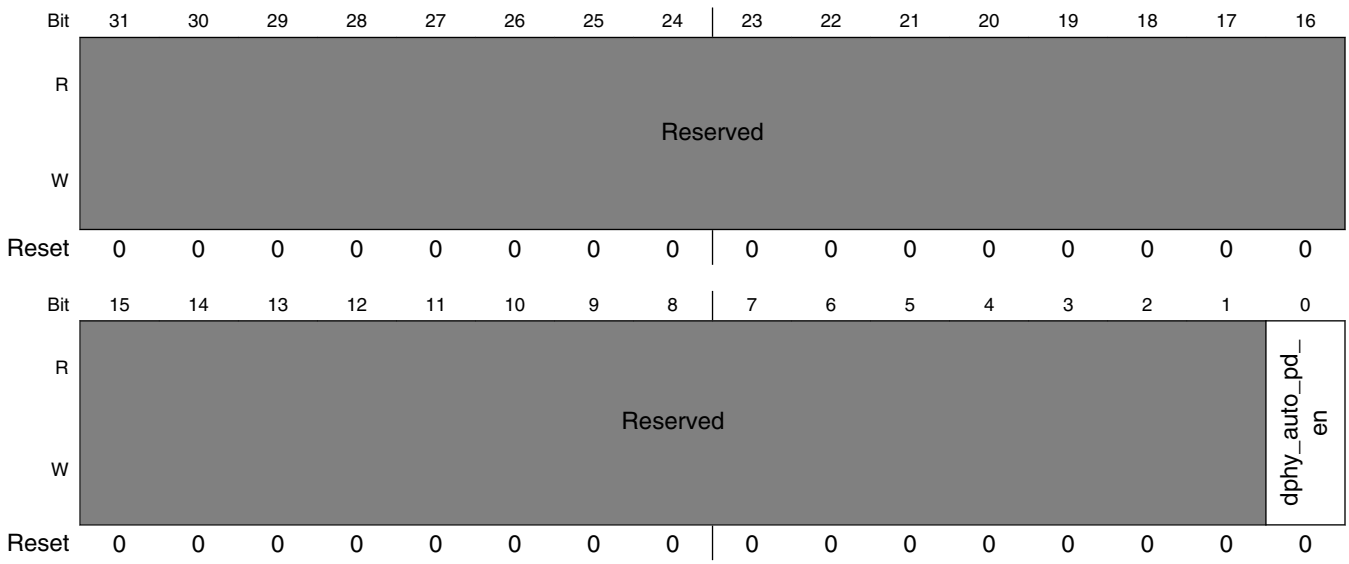


MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_RTERM_SEL field descriptions

Field	Description
31–1 -	This field is reserved.
0 dphy_rterm_sel	DPHY RTERM_SEL input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.16 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_AUTO_PD_EN

Address: 40A9_0000h base + 33Ch offset = 40A9_033Ch



MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_AUTO_PD_EN field descriptions

Field	Description
31–1 -	This field is reserved.
0 dphy_auto_pd_en	DPHY AUTO_PD_EN input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.17 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_RXLPRP

Address: 40A9_0000h base + 340h offset = 40A9_0340h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														dphy_rxlprp	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_RXLPRP field descriptions

Field	Description
31–2 -	This field is reserved.
dphy_rxlprp	DPHY RXLPRP input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

45.12.18 MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_RXCDRP

Address: 40A9_0000h base + 344h offset = 40A9_0344h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														dphy_rxcdrp	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MIPI_DSI_HOST_FSL_IP1_DPHY_INTFCx_DPHY_RXCDRP field descriptions

Field	Description
31–2 -	This field is reserved.
dphy_rxcdrp	DPHY RXCDRP input. Detailed information about this parameter programming is available in the MIPI-DSI DPHY section.

Chapter 46
2D Graphics Processing Unit (GPU2D)

46.1 Chip-specific GPU2D information

Table 46-1. Reference links to related information

Topic	Related module	Reference
Full description	GPU2D	GPU2D
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

46.1.1 2D Graphic processing unit (GPU-2D)

i.MX 7ULP integrates GPU for 2D graphic composition processing. The GC320 GPU-2D supports user interface rendering and performs functions like blending, filtering, rotation, overlay, resizing, transparency, and other dynamic effects.

The module uses 128-bit AXI master bus interface and 32-bit AHB slave bus interface

Table 46-2. GPU-2D configuration

Parameter	Description
Name	GPU-2D
Supported standard version	GC320
Instances	1
Configurable features	NA
Interface speed	NA
External I/O pins	NA

46.2 Overview

The 2D Graphics Processing Unit (GPU2D) is a high-performance multi-pipe 2D graphics core that accelerates the 2D graphics display on a variety of devices. The GPU2D supports a wide-range of addressable screen sizes and resolutions.

The R2D GPU Hardware acceleration is brought to numerous 2D applications including graphical user interfaces (GUI), menu displays, flash animation, and gaming.

46.3 GPU2D Block Diagram

46.3.1 R2D GPU

The R2D Graphics Processing Unit (GPU) defines a high-performance 2D raster graphics core that accelerates the 2D graphics display.

R2D GPU supports acceleration of the following graphics APIs:

- DirectFB (Linux / Linux Embedded)
- GDI / DirectDraw (Windows Embedded Compact 7 / Embedded CE 6)
- Android

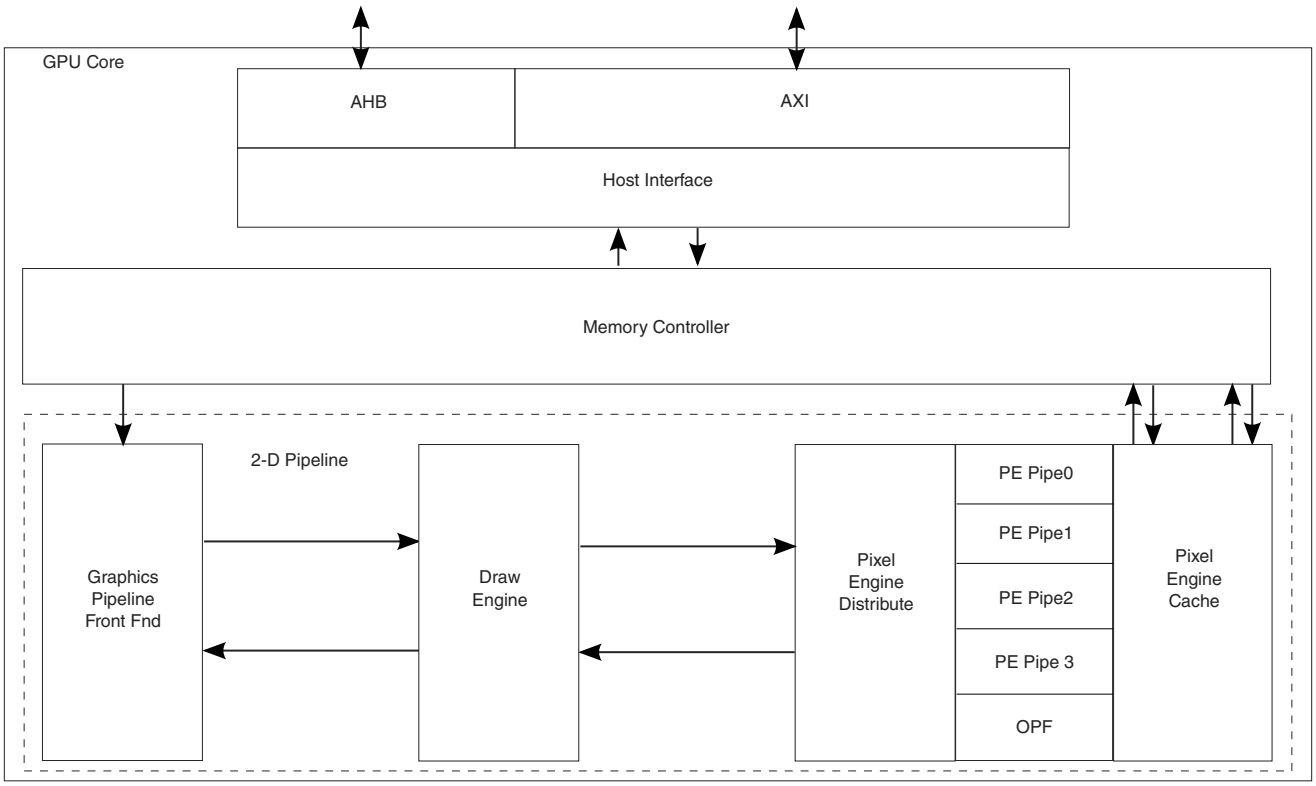


Figure 46-1. R2D GPU Block Diagram

46.4 GPU2D Features

The following sections describe the functional features of the R2D GPU.

46.4.1 Full Featured R2D GPU Pipeline

- Bit BLT
- Stretch BLT
- Rectangle fill and clear
- Line drawing
- Filter BLT
- Mono expansion for text rendering
- ROP2, ROP3, and ROP4
- Alpha blending, including Java 2 Porter-Duff compositing blending rules
- 32K x 32K coordinate system
- 90 / 180 / 270 degree rotation
- Transparency by monochrome mask, chroma key, or pattern mask
- Support 2x2 in 4x4 tile format

- A8 output with rotation in filter BLT and bit BLT
- SrC/Dest color key full bypass support

46.5 GPU2D OPERATIONS

46.5.1 R2D GPU Operations

Information detailing the R2D GPU operations can be found in this section.

46.5.1.1 Line

The LINE operation draws a line. Coordinates for two points are given: start point and end point. The end point is not drawn. Lines are rendered using the Bresenham algorithm. The Bresenham algorithm has the advantage of using integer arithmetic and has no accumulation of rounding errors.

In the case of line, only ROP2 and ROP4 are supported. It operates on pattern and destination. The pattern should have a transparency mask in order to use ROP4.

Clipping is supported for lines on a per pixel basis.

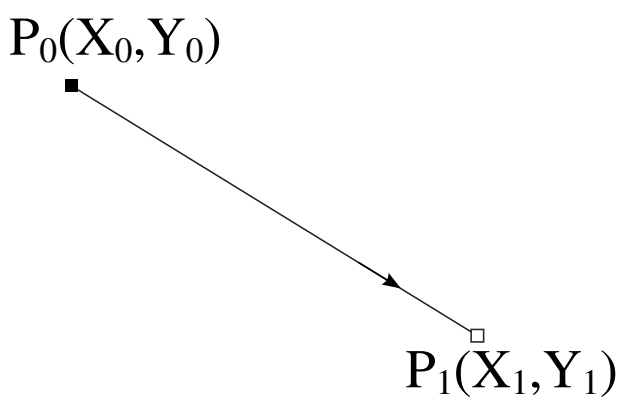


Figure 46-2. Line

46.5.1.2 Rectangle Fill and Clear

Rectangle fill creates a rectangle area with a given color or a pattern fill. Essentially rectangle fill is a pattern fill, where an 8x8 pattern is initialized with the specified color. It supports ROP2 and ROP4 with the pattern and destination as its inputs. If ROP4 is used, the pattern should have a transparency mask.

Clear is similar to Rectangle Fill except that it does not use a pattern. A 32-bit clear value with 4-bit byte mask is used to fill the entire rectangle area.

Both Rectangle Fill and Clear support clipping, which is performed on a per primitive basis.

46.5.1.3 BitBLT

Bit blit (BitBLT) transfers data from one area of a memory (source) to another area of the memory (destination).

The source and destination can be from the same or different memory locations. Both source and destination must be described by a rectangular area. The source and destination rectangles can be of the same size (most bit blits are of this nature) or of different sizes, in which case, the operation becomes a stretch or shrink blit.

BitBLT supports ROP2, ROP3, and ROP4 which includes source, destination and pattern, and an optional transparency color. Clipping can be performed on a primitive basis.

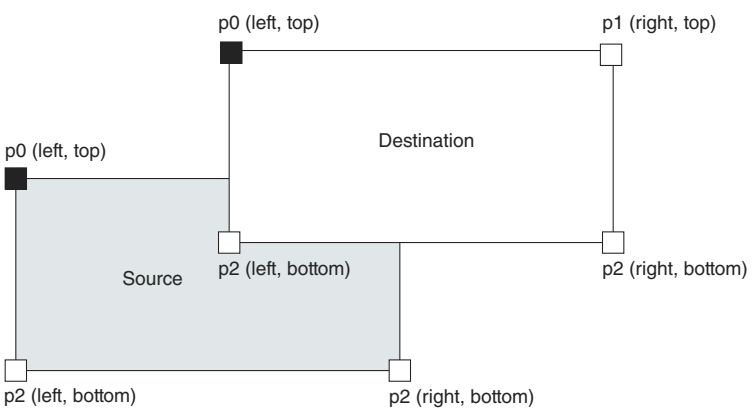


Figure 46-3. BitBLT

The graphics engine supports the following source and destination formats for data bit blits and filter blits. In addition to these source and destination RGB formats, their swizzle formats (ARGB, RGBA, ABGR, BGRA) are also supported. For YUV formats, the GPU supports their U/V swap formats.

Table 46-3. BitBLT Formats

Formats	Source Image	Destination Image
A1R5G5B5	Yes	Yes
A4R4G4B4	Yes	Yes
X1R5G5B5	Yes	Yes
X4R4G4B4	Yes	Yes
R5G6B5	Yes	Yes
A8R8G8B8	Yes	Yes
X8R8G8B8	Yes	Yes
A8	Yes	Yes
1-bit monochrome	Yes	No
8-bit color index	No	No
YUY2 (packed YUV422)	Yes	Yes
UYVY (packed YUV422)	Yes	Yes
YV12 (planar YUV420)	Yes	MD
NV12 (semi-planar YUV420)	Yes	MD
NV16 (semi-planar YUV422)	Yes	MD

MD = Multi-Destination support

46.5.1.4 Stretch BLT

The stretch blit (Stretch BLT) primitive performs a bit blt operation with stretch or shrink. The modified Bresenham algorithm is used to generate corresponding coordinates for fast stretching. The stretch factor is specified in a 15.0 fixed-point format. Stretch blit is not allowed to overlap, therefore no part of source and destination can share any piece of memory. Non-stretch blits can overlap. For stretch blit, clipping is performed on a per pixel basis.

46.5.1.5 Monochrome Expansion and Mask BLT

Monochrome expansion and mask blit are different operations, although both use the bit stream from command buffer and both can be the source for ROP4 source selection. This means that each output pixel can be a combination of source, pattern, monochrome mask (for masked blits) and destination.

46.5.1.5.1 Monochrome expansion

For monochrome expansion, the bit from the stream is used to switch on/off a solid color that is defined in a register. This mechanism enables the use of just one bit per pixel to represent colors. In effect, the MONO EXPANSION primitive increases color representation from one bit per pixel to multiple bits per pixel. A typical application for mono color is font drawing.

Monochrome expansion does not support overlapping of the source and destination. It is the responsibility of the driver to make sure that the command will never be executed with overlapping source and destination.

46.5.1.5.2 Mask BLT

For Mask BLT, the bit from the stream is used to toggle on/off a color in the source frame buffer. Mask BLT takes its color source from memory and its monochrome mask from the command stream. Clipping is supported and is performed on a per pixel basis.

46.5.1.6 Filter BLT

Filter blit performs high quality scaling, up or down, using an FIR re-sampling filter with up to 9 taps. The sub-pixel coordinates (locations between the pixel grids) are generated by the drawing engine. The filter block in the drawing engine uses the sub-pixel information to select the appropriate filter kernel. R2D GPU processes 1 pixel every cycle when performing filter blit.

A stretch- or shrink-factor of 15.16 fixed-point format is supported. To generate a single destination pixel requires 9 source pixels. An image is scaled in two passes, one for X-dimension (HOR_FILTER_BLT) and the other for Y-dimension (VER_FILTER_BLT). The software sets up the filter kernel/coefficient table and the kernel size, as well as a temporary buffer for storing intermediate results. After the first pass is completed, intermediate results are sent back to memory, and then the second pass starts to scale the first-pass image. Because of this two-step procedure, the throughput of FILTER BLT is lower than that of STRETCH BLT. Also the Filter Kernel Table may need to be reloaded, and some cycles are consumed in calculating the stepping parameters.

When the stretch or shrink factor is 1, the filterBlit works as a bitBlit copy. It can be used as format converter in that case, for instance, YUV to RGB converter. To use as a format converter, only one pass (HOR_FILTER_BLT or VER_FILTER_BLT) is needed. To optimize the memory bandwidth, when using filterBlit to do YUV to RGB filtering, the temporary target buffer format can be specified as YUY2 to process Y-dimension

filtering (VER_FILTER_BLT). This is to avoid converting YUV to A8R8G8B8 in the 1st vertical pass to reduce the memory bandwidth and increase the pixel processing rate. This is the only special case that GPU may use YUY2 as target format.

When the stretch or shrink factor (scale ratio) is not 1:1, filter blit requires both a vertical and a horizontal pass to do the scaling. Shrink performance will be less than 1 pixel per cycle for each vertical and horizontal pass. Stretch performance will be near to the performance for the 1:1 scale ratio.

The Filter BLT primitive supports the following source and destination image formats:

Table 46-4. Filter BLT Formats

Formats		Source Image	Destination Image
A1R5G5B5		Yes	Yes
A4R4G4B4		Yes	Yes
A8R8G8B8		Yes	Yes
R5G6B5		Yes	Yes
X1R5G5B5		Yes	Yes
X4R4G4B4		Yes	Yes
X8R8G8B8		Yes	Yes
YUV	NV12 (4:2:0, semi-planar)	Yes	No
	NV16 (4:2:2, semi-planar)	Yes	No
	UYVY (4:2:2, packed)	Yes	No
	YUY2 (4:2:2, packed)	Yes	Yv[1]
	YV12 (4:2:0)	Yes	No
	8-bit color index	Yes	No
	A8	Yes	Yes

1. Yv is possible only when vertical filter blit is done on YUV input.

46.5.1.7 R2D Performance of different operations

Table 46-5. Performance of different operations with/without rotation

Primitive	Peak Performance	Source/destination overlap	Clipping
Line	1 pixel / cycle	N/A	Done on pixel basis
Rectangle	2 pixel / cycle	N/A	Done on primitive basis
Clear	2 pixel / cycle	N/A	Done on primitive basis
Blit	2 pixel / cycle	Overlap is allowed	Done on primitive basis
Stretch blit	1~2 pixel / cycle	No overlap allowed	Done on pixel basis
Monochrome expansion	1 pixel / cycle	No overlap allowed	Done on pixel basis
Filter blit	1 pixel / cycle	N/A	Done on primitive basis

NOTE

The performance decreases for filter blit operations which require two passes.

46.5.1.8 Rotation

- 90° / 180° / 270° / Mirror rotation is supported for all primitives.
- Independent source rotation

46.5.1.9 Transparency Mode

For monochrome expansion:

- Opaque
- Conditional transparency: Transparent if the current pixel matches the specified value.

For blits:

- Opaque
- Masked transparency: Transparent if the mask for the current pixel or pattern is zero.
- Source Conditional transparency: Transparent if the source pixel is within the specified value range.
- Destination Conditional transparency: Transparent if the destination pixel is not within the specified value range.

46.5.1.10 Clipping

One clipping rectangle is supported for all bit blit primitives.

46.5.1.11 R2D GPU Data Formats

The graphics engine supports the following source and destination formats for data bit blits and filter blits. In addition to these source and destination RGB formats, their swizzle formats (ARGB, RGBA, ABGR, BGRA) are also supported. For YUV formats, the GPU supports their U/V swap formats.

Table 46-6. Data Formats

Format	Bit Blit Input	Bit Blit Output	Stretch Blit Input	Stretch Blit Dst	Filter Blit Input	Filter Blit Dst	OPF Input	OPF Dst	Multi-Source Input	Multi-Source Dst
A1R5G5B5	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
A4R4G4B4	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
A8R8G8B8	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
X1R5G5B5	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
X4R4G4B4	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
X8R8G8B8	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
R5G6B5	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
A8	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
YUY2 (packed YUV422)	Y	Y	Y	N	Y	Yv	Y	Y	Y	Y
UYVY (packed YUV422)	Y	Y	Y	N	Y	N	Y	Y	Y	Y
YV12 (planar YUV420)	Y	MD	Y	N	Y	N	Y	N	Y	N
NV12 (semi-planar YUV420)	Y	MD	Y	N	Y	N	Y	N	Y	N
NV16 (semi-planar YUV422)	Y	MD	Y	N	Y	N	Y	N	Y	N
8-bit color index	N	N	Y	N	Y	N	N	N	Y	N
1-bit monochrome	Y	N	N	N	N	N	N	N	N	N

Yv: Only possible when vertical filter blit is done on YUV input.
MD: Multi-Destination support

YUV output is available for packed or planar YUV input formats per the following table.

Table 46-7. YUV Output Support for Packed or Planar YUV Source per 2D Operation

2D Operation	Destination
Bit blit	Packed YUV
One Pass Filter (OPF)	Packed YUV
Multi-source	Packed YUV (no scaling)
Multi-destination	Planar YUV
Stretch blit	No
Two Pass Filter	No

46.5.1.12 ARGB Data Conversion of R2D GPU

The pixels read from source or destination will be expanded into A8R8G8B8 format to maintain lossless pixel operations. The resulting pixels will be converted into the destination format.

46.5.1.13 YUV to RGB Conversion of R2D GPU

YUV data can be converted into 8-bit per component RGB format at the output of the cache only. Once converted, there is no way back to YUV format. GPU supports BT.601 and BT.709 YUV to RGB color conversion standards.

In BT.601, the YUV to RGB conversion is done using the following approximation:

16 <= Y <= 235
16 <= U <= 240
16 <= V <= 240

A = Y - 16
B = U - 128
C = V - 128

R = clip((298*A + 410*C + 128) >> 8)
G = clip((298*A - 101*B - 209*C + 128) >> 8)
B = clip((298*A + 519*B + 128) >> 8)

The Y, U and V components are clamped prior to the conversion.
Y is clamped between 16 and 235, inclusively.
U and V are clamped between 16 and 240, inclusively.

In BT.709, the R, G, B equations are slightly changed to

R = clip((298*A + 461*C + 128) >> 8)
G = clip((298*A - 55*B - 137*C + 128) >> 8)
B = clip((298*A + 543*B + 128) >> 8)

46.5.1.14 Source/Destination Pre-multiply and De-Multiply Support

GPU supports source pre-multiply source alpha or global alpha, or source global color for global colorizing. On destination, the GPU supports destination pre-multiply destination alpha, destination de-multiply alpha.

46.5.1.15 Alpha Blending

The GPU supports alpha blending together with ROP. The alpha blending function is performed on ROP function result source.

The general alpha blending equations are:

Cd = Fs * Cs' + Fd * Cd'

Ad = Fs * As'' + Fd * Ad''

Where

- Cs' is the source color component (adjusted for NPM if necessary)
- Cd' is the destination color component (adjusted for NPM if necessary)
- As'' is the modified source alpha component
- Ad'' is the modified destination alpha component
- Fs is fraction of the source that contributes to the final value
- Fd is fraction of the destination that contributes to the final value

The blending is done in 5 logical stages (not real implementation stages):

1. Transparent/opaque conversion
   - In this stage, the incoming alpha (source or destination independently) can be inverted if needed to match the internal alpha rule. Internally, an alpha of 0 means transparent, while an alpha of "0xFF" means opaque. External content might follow the opposite rule. The output of the block is either As (Ad for destination) or 1-As (1-Ad for destination).
2. Global value substitution
   - A global alpha value from a register can be used to substitute or scale the incoming alpha. An incoming alpha As can pass-through, be directly substituted by Ags (global alpha) or scaled by the global alpha value (As * Ags). The source and destination have distinct global alpha values.
3. Blending factor generation
   - At this stage, the blending factors are generated (refer to table below). Each alpha can take the values 0, 1, A or 1-A depending on the blending mode.
4. Final blending
   - This is the final stage which implements blending equations.

   The fractions take the values described in the following table, depending on the blending mode.

Table 46-8. Blending Modes Fractions Description

Blending Mode	Fs	Fd
Clear	0	0
SRC	1	0
DST	0	1
SRC_OVER	1	1 - As"
DST_OVER	1 - Ad"	1
SRC_IN	Ad"	0
DST_IN	0	As"
SRC_OUT	1 - Ad"	0
DST_OUT	0	1 - As"
SRC_ATOP	Ad"	1 - As"
DST_ATOP	1 - Ad"	As"
XOR	1 - Ad"	1 - As"

To control the blending modes, the following register fields are used:

- 1 bit for transparent/opaque conversion for source alpha
- 1 bit for transparent/opaque conversion for destination alpha
- 2 bits for source alpha modifications, to specify the 3 cases (As, Ags, As*Ags)
- 2 bits for destination alpha modifications, to specify the 3 cases (Ad, Agd, Ad*Agd)
- 4 bits to select between the 12 blending modes
- 8 bits for global source alpha
- 8 bits for global destination alpha

Alpha blending is supported on bit blit and filter blit primitives.

46.5.1.16 GPU Multi-Source Blending

- Up to eight sources are supported.
- Programmable block size guarantees cache efficiency so each source is read once and each destination is written once.

- It supports 90, 180, 270 degree destination and source rotation and mirror with different block size to have high cache efficiency.
- It also supports ROP2, ROP3, and ROP4 which includes source, destination and pattern, and an optional transparency color. Alpha blending between every source is supported.

46.5.1.17 GPU Cache Management

The software cache flush is supported to flush the GPU cache to memory. Auto-flush of GPU cache is also supported. The software sets up the auto-flush interval, and hardware will do the cache flush automatically at programmable intervals.

Chapter 47
3D Graphics Processing Unit (GPU3D)

47.1 Chip-specific GPU3D information

Table 47-1. Reference links to related information

Topic	Related module	Reference
Full description	GPU	GPU
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

47.1.1 3D Graphic processing unit (GPU3D)

i.MX 7ULP integrates the GC7000 Nano Ultra GPU supporting OpenGL ES2.0/1.1, Desktop OpenGL 2.1, OpenVG1.1, and GLSL shading language support.

GPU3D uses the 256-bit AXI master bus interface and 32-bit AHB slave bus interface

Table 47-2. GPU3D configuration

Parameter	Description
Name	GPU
Supported standard version	GC7000 Nano Ultra
Instances	1
Configurable features	NA
Interface speed	NA
External I/O pins	NA

47.2 Overview

The GPU3D is a high-performance core that delivers hardware acceleration for 3D graphics display.

Addressable screen sizes range from the smallest cell phones to HD 720p displays. It provides high performance, high quality graphics, low power consumption, and the smallest silicon footprint.

GPU3D accelerates numerous 3D graphics applications, including graphical user interfaces (GUI), menu displays, flash animation, and gaming. This module supports the following graphics APIs:

- OpenGL ES 2.0
- OpenGL ES 1.1
- OpenVG 1.1
- DirectX 11 (9_3)
- OpenGL 2.1
- EGL 1.4

47.3 GPU3D Block Diagram

The main functional units of the GPU3D are shown in the figure below and their description is as follows:

- Host Interface: Allows GPU3D to communicate with external memory and the CPU through AXI or AHB bus. In this block data crosses clock domain boundaries.
- Memory Controller: Internal memory management unit that controls the block-to-host memory request interface.
- Graphic Pipeline Front End: Inserts high level primitives and commands into the graphics pipeline.
- Ultra-threaded Unified Shader: SIMD processor that performs as both vertex shader and fragment shader. When used as a vertex shader it performs geometry transformations and lighting computations. When used as a fragment shader, it applies texture data and computes color values for each pixel. GPU3D has one such shader.
- 3D Rendering Engine: Converts triangles and lines into pixels. Computes slopes of color attributes and texture coordinates. Performs clipping.

- Texture Engine: Retrieves texture information from memory upon request by the fragment shader. Performs interpolation and filtering, and transfers the computed value to the fragment shader or the vertex shader.
- Pixel Engine/Resolve: Pixel engine does alpha blending and visible surface determination. Resolve does tiling and de-tiling.

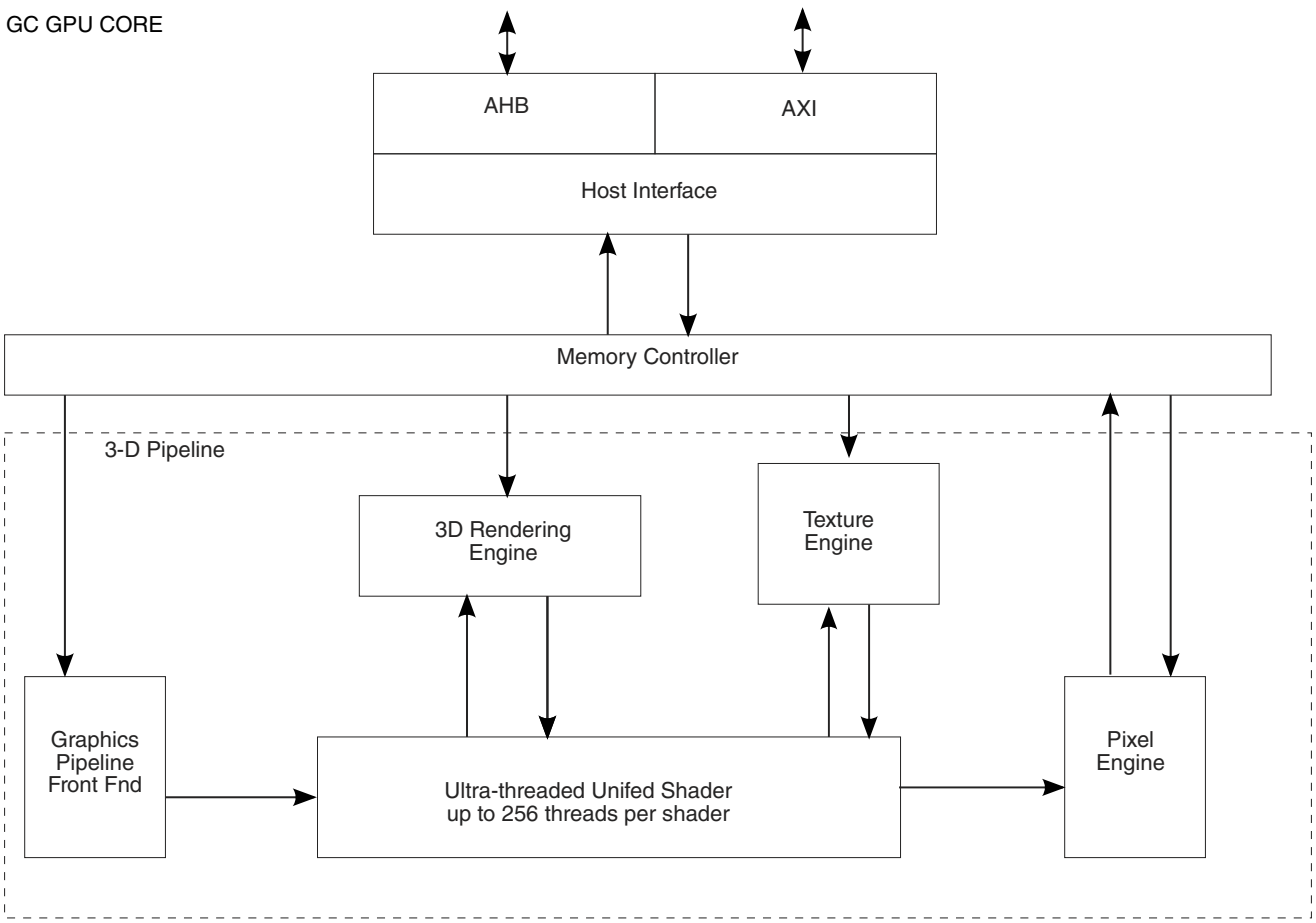


Figure 47-1. GPU3D Top Level Block Diagram

The core clock (core_clk) is the main clock for the processing control. The core clock configuration for the GPU is determined by the configuration of PFD on SCG and clock mux/dividers on PCC.

The GPU3D provides architectural support for the following features:

Table 47-3. GPU3D Architectural Support

Feature	GPU Support
Primary API	OpenGL ES1.1 and 2.0
Additional API's	OpenVG1.1
	DirectX 11 (9_3)

Table continues on the next page...

Table 47-3. GPU3D Architectural Support (continued)

Feature	GPU Support
	OpenGL 2.1
Other Graphics Support	EGL1.4
Drivers	OpenGL ES 2.0 / 1.1
	OpenVG1.1
	EGL1.4
	DirectX 11 (9_3)
	OpenGL 2.1
Operating Systems	Windows Embedded Compact7, Embedded CE6
	Linux Embedded
	Android 4.x / 3.x / 2.x
Z (depth)	Early Z support included
Stencil	Early stencil support included
GLSL ES Shader languages	1.0
Shader types and execution units	One programmable Scalable Ultra-threaded Unified Shaders (SIMD4:ctl-flow,tx-load) one instruction issue per shader per clock; IEEE 32-bit floating-point pipeline supports long shader instructions
Big endian support	Available upon request
Shader model compatibility	SM3.0

47.4 GPU3D Hardware Features

The GPU3D block has the following hardware features:

- OpenGL ES 2.0 / 1.1 compliance, including extensions; OpenVG 1.1
- IEEE 32-bit floating-point pipeline
- Ultra-threaded, unified vertex and fragment shaders
- Low bandwidth at both high and low data rates
- Low CPU loading
- Up to 12 programmable elements per vertex
- Dependent texture operation with high-performance
- Alpha blending
- Depth and stencil compare
- Point sampling, bi-linear sampling, tri-linear filtering, and cubic textures
- Resolve and fast clear
- 8k x 8k texture size and 8k x 8k rendering target
- 4 Vertex DMA streams
- MMU functionality supported

Table 47-4. Unified vertex-fragment shader features

Feature	GPU Support
Shader type and execution units	Unified shader, SIMD4, SFP32 Trans
Swizzle capabilities	Full 32-bit word level swizzle in a 128-bit vector
GPR's per shader	Up to 512 general purpose registers, 128 bits each
Uniform registers	Total uniform registers: 320 (Uniform Vertex Shader: 256 registers, 128 bits each, and Uniform Fragment Shader: 256 registers, 128 bits each)
FP denorm and rounding options	Denorms are set to zero. Supports rounding to zero.
Maximum number of data input attributes	Maximum of 12 vertex shader input elements; Maximum of 8 fragment shader input elements;
Maximum number of instructions	256 for vertex and 256 for fragment shaders
Maximum number of vertex streams	4
Maximum number of threads in flight per shader core	256
Subroutines	4 levels
Conditional branch support	GT, LT, EQ, GE, LE, NE
Shader instruction rate	1-cycle throughput for all shader instructions
Floating-point instruction precision	SIMD4 (vector): 23.5 bits
Fragment shader video	Supports video texture
Samplers	Total 12:4 VS Samplers and 8PS Samplers

Table 47-5. Vertex Processing Features

Feature	GPU Support
Vx D3D, OGL ES formats supported	BYTE, UBYTE, SHORT, USHORT, INT, UINT, DEC, UDEC, FLOAT, FLOAT16, D3DCOLOR, FIXED16DOT16
Vertex data size limits	256 bytes
Pre shader cache	1 kB
Post shader cache	8 vertices

Table 47-6. Primitive Processing Features

Feature	GPU Support
Primitives supported	triangle strip, fan, and list; line strip and list; point list
Vertex/primitive geometry input index sizes	8-bit ,16-bit and 32-bit indices
Setup parameters available to fragment shader	8 vec4 parameters; all available to fragment shader

Table 47-7. Texture Processing Features

Feature	GPU Support
Fixed-point input texture formats	A8, L8, I8, A8L8, ARGB4, XRGB4, ARGB8, XRGB8, ABGR8, XBGR8, R5G6B5, A1RGB5, X1RGB5, YUY2, UYVY, D16, D24X8, A8_OES, DXT1, DXT2, DXT3, DXT4, DXT5, ETC1, R8,R8G8

Table continues on the next page...

Table 47-7. Texture Processing Features (continued)

<table>
<tr><th>Feature</th><th>GPU Support</th></tr>
<tr><td></td><td>All fixed-point formats are filtered. .
<table>
<tr><td>Bits</td><td>Format</td><td>Alpha</td><td>R</td><td>G</td><td>B</td></tr>
<tr><td>16</td><td>ARGB4444</td><td>4</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>XRGB4444</td><td>4 don't care</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>ARGB1555</td><td>1</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>XRGB1555</td><td>1 don't care</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>RGB565</td><td>0</td><td>5</td><td>6</td><td>5</td></tr>
<tr><td>32</td><td>ARGB8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XRGB8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>ABGR8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XBGR8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
</table>
<table>
<tr><td>Planes</td><td>Format</td><td>Mode</td><td>Y</td><td>U</td><td>V</td><td>UV</td><td>YUYV</td><td>UYVY</td></tr>
<tr><td>1</td><td>YUY2</td><td>4:2:2</td><td></td><td></td><td></td><td></td><td>1</td><td></td></tr>
<tr><td>1</td><td>UYVY</td><td>4:2:2</td><td></td><td></td><td></td><td></td><td></td><td>1</td></tr>
</table></td></tr>
<tr><td>Texture compression</td><td>4 bits and 8 bits per texel</td></tr>
<tr><td>Additional texture formats supported through Resolve conversion (Optional, not available in default configuration)</td><td>Resolve converts these planar formats to YUV4:2:2 packed:
<table>
<tr><th>Planes</th><th>Format</th><th>Mode</th><th>Y</th><th>U</th><th>V</th><th>UV</th><th>YUYV</th><th>UYVY</th></tr>
<tr><td>3</td><td>YV12</td><td>4:2:0</td><td>1</td><td>1</td><td>1</td><td></td><td></td><td></td></tr>
<tr><td>2</td><td>NV12</td><td>4:2:0</td><td>1</td><td></td><td></td><td>1</td><td></td><td></td></tr>
</table></td></tr>
<tr><td>Compressed texture formats</td><td>DXT1, DXT2, DXT3, DXT4, DXT5, ETC1
All compressed formats are filtered.</td></tr>
<tr><td>Compressed texture supertile</td><td>Supported</td></tr>
<tr><td>Texture size maximum</td><td>8k x 8k</td></tr>
<tr><td>Addressing modes</td><td>Wrap, mirror, clamp</td></tr>
<tr><td>Mipmap support</td><td>14 mipmap levels; programmable LOD biasing & replacement</td></tr>
<tr><td>Shadow texture</td><td>Depth texture PCF filtering</td></tr>
<tr><td>Texture coordinate fraction bits</td><td>5 bits</td></tr>
<tr><td>Texture sampler units</td><td>8 samplers, indexable</td></tr>
<tr><td>Textures per fragment maximum</td><td>8 texture samplers</td></tr>
<tr><td>Dependent texture operation</td><td>High performance; unlimited dependent texture reads</td></tr>
<tr><td>Dependent tx per fragment max, relative sampling</td><td>No limit</td></tr>
<tr><td>Texture repeat max</td><td>256</td></tr>
<tr><td>Texture types</td><td>2D, cube map, 1D, projected, depth, bump map, displacement map</td></tr>
</table>

Table continues on the next page...

Table 47-7. Texture Processing Features (continued)

Feature	GPU Support
Texture filters	Point sample, bi-linear, tri-linear
Texture component mapping: D3D, OGL, ES options	Supports both D3D and OES options
Texture size types	Power-of-2, Non-square texture support
Texture cache size	32 cache lines, with 64 bytes per cache line; 2 KB texture cache total

47.4.1 Rasterization

Table 47-8. Rasterization features

Feature	GPU Support
Interpolant attributes limit	8
Render target size	8K x 8K
Clipping window	Clipping rectangle supported
Early Z	Yes

47.4.2 Fragment Processing

Table 47-9. Fragment Processing Features

<table>
<tr><th>Feature</th><th>GPU Support</th></tr>
<tr><td>Fragment color, alpha, Z, stencil precision</td><td>A8, R8, and R8G8 as listed below
<table>
<tr><th>Bits</th><th>Format</th><th>Alpha</th><th>R</th><th>G</th><th>B</th></tr>
<tr><td>16</td><td>ARGB4444</td><td>4</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>XRGB4444</td><td>4 don't care</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>16</td><td>ARGB1555</td><td>1</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>XRGB1555</td><td>1 don't care</td><td>5</td><td>5</td><td>5</td></tr>
<tr><td>16</td><td>RGB565</td><td>0</td><td>5</td><td>6</td><td>5</td></tr>
<tr><td>32</td><td>ARGB8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XRGB8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>ABGR8888</td><td>8</td><td>8</td><td>8</td><td>8</td></tr>
<tr><td>32</td><td>XBGR8888</td><td>8 don't care</td><td>8</td><td>8</td><td>8</td></tr>
</table>
<table>
<tr><th>Bits</th><th>Format</th><th>Depth</th><th>Stencil</th></tr>
<tr><td>16</td><td>D16</td><td>16</td><td>0</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

Table 47-9. Fragment Processing Features (continued)

Feature	GPU Support			
	Bits	Format	Depth	Stencil
	32	D24S8	24	8
Fragment storage	For color: 16-/32-bit. For depth: 16-bit z or 24-bit z & 8-bit stencil for each fragment. Lossless compression, no storage reduction			
Alpha support	Individual fragment alpha masking			
Fragment cache	16 cache lines for color 16 cache lines for Z 16 bytes per cache line			

47.4.3 Destination formats

Table 47-10. Destination formats

Feature	GPU Support					
Destination color formats	Bits	Format	Alpha	R	G	B
	16	ARGB4444	4	4	4	4
	16	ARGB1555	1	5	5	5
	16	RGB565	0	5	6	5
	32	ARGB8888	8	8	8	8
	32	ABGR8888	8	8	8	8

47.4.4 Z/Stencil Buffer

Table 47-11. Z/Stencil Buffer Features

Feature	GPU Support
Z/stencil formats	16-bit Z; 24-bit Z plus 8-bit stencil, with lossless compression support
Z/stencil buffer	16 cache lines; 64 bytes per line;
Stencil support	Stencil compression, two-sided stencil

47.4.5 Render Target

Feature	GPU Support
Formats	16-bit and 32-bit, with lossless compression support
RT buffer cache	16 cache lines; 64 bytes per line; RT caches are fully set associative.
Blend modes	Porter-Duff blending modes
Render target dithering support	Yes

47.5 Usage Mode

The GPU3D should be programmed through the NXP provided driver. NXP does not provide support for software that directly programs the GPU3D registers. APIs for programming the GPU3D through the software driver are described in separate driver documentation.

Chapter 48
Enhanced LCD Interface (LCDIF)

48.1 Chip-specific LCDIF information

Table 48-1. Reference links to related information

Topic	Related module	Reference
Full description	LCDIF	LCDIF
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

48.1.1 Liquid Crystal Display Interface (LCDIF)

The LCDIF is a general purpose display controller used to drive a wide range of display devices varying in size and capabilities. The LCDIF is used as a bridge (or gasket) between the DSI controller and the NIC0 crossbar. It uses 64-bit AXI master bus (Read only) as well as 32-bit IP Bus.

Table 48-2. LCDIF configuration

Parameter	Description
Name	Liquid Crystal Display Interface (LCDIF)
Supported standard version	MIPI DPI equivalent signal interface
Instances	1
Configurable features	No Read mode
Interface speed	NA
External I/O pins	LCDIF IO signals are internally connected to DSI controller DPI interface. There are no external signals available from the LCDIF.

48.1.2 Register Access

Any accesses made to LCDIF reserved addresses may incur on unknown effects.

48.2 Overview

The enhanced Liquid Crystal Display Interface (LCDIF) is a general purpose display controller

The LCDIF block supports the following:

- Displays that support moving pictures and require the RGB interface mode (DOTCLK interface).

The LCDIF provides fully programmable functionality to supported interfaces:

- Bus master interface to source frame buffer data for display refresh.
- 8/16/18/24 bit LCD data bus support available.
- Programmable timing and parameters for DOTCLK LCD interfaces.

48.3 External Signals

No LCDIF pin signals are pinned out on this chip.

48.4 Functional Description

Bus Interface Mechanisms through Initializing the LCDIF, describe the internal pipeline for the LCDIF interfaces. Differences for each mode are then described in separate sections, as follows:

- DOTCLK Interface

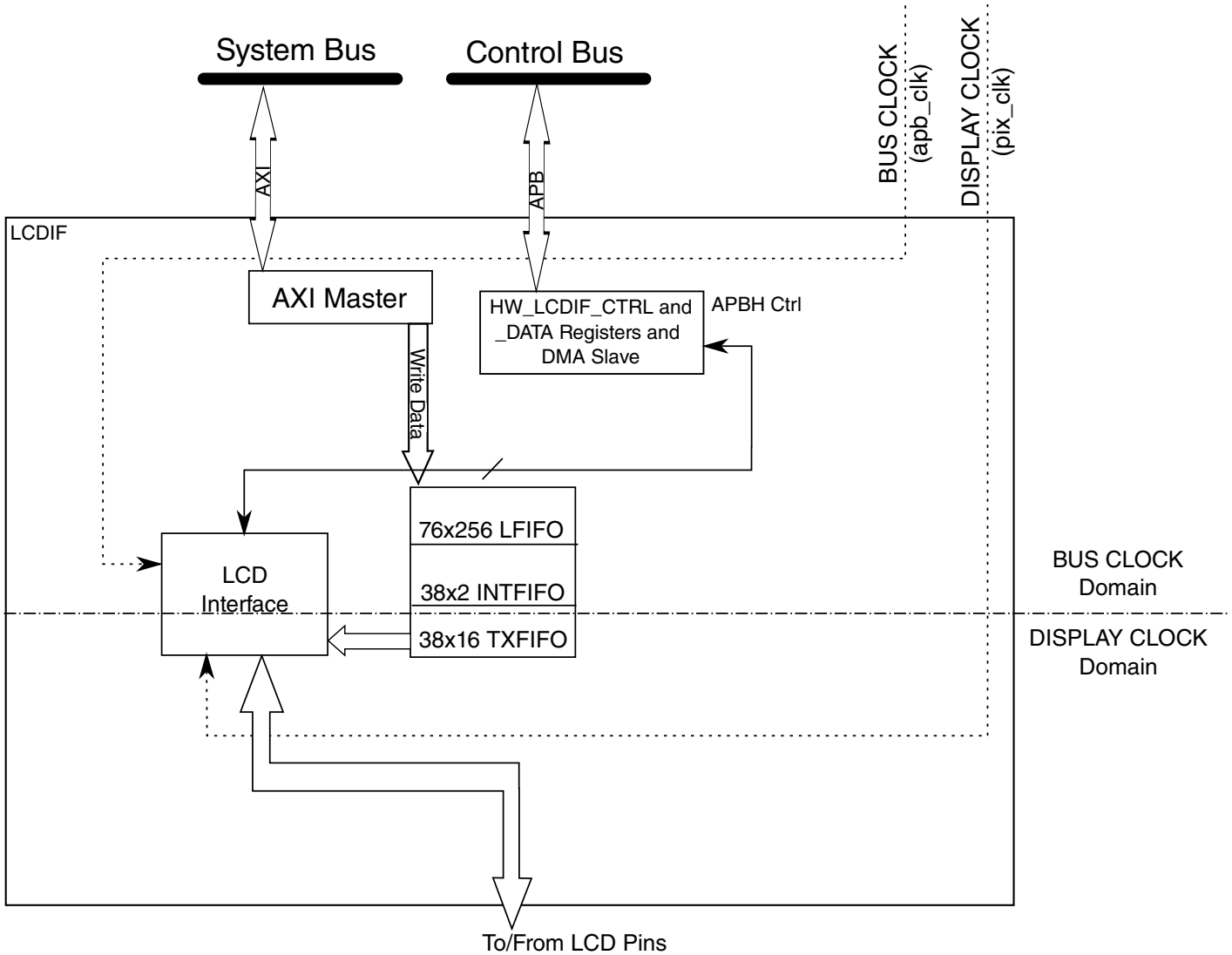


Figure 48-1. Top-Level Block Diagram of LCDIF subsystem

48.4.1 Bus Interface Mechanisms

The LCDIF module has memory-mapped control, data and status registers. It provides several interfaces to transfer data between the display and SoC.

The bus master interface is used to initiate the requests to transfer data from external memory to the display. It is completely autonomous, or no CPU intervention is required, to manage the cyclical nature of refreshing standard display types.

The host CPU executes display drivers to manage the display solution. The following sections describe the system bus interface mechanisms.

48.4.1.1 Bus Master Operation in Write/Display Modes

The LCDIF block has a bus master interface that initiates requests for data to drive the display. The LCDIF_MASTER bit must be set to 1 to enable the bus master interface. Software should program all control registers required to transfer the frame sequence.

The DOTCLK mode is used to refresh the display at the desired refresh rate and resolution, and drive displays that don't integrate a display buffer memory. When the display is refreshed, the LCDIF will automatically update the LCDIF_CUR_BUF_ADDR register with the value in LCDIF_NEXT_BUF_ADDR at the end of current frame and start fetching the next frame from the new address. If the LCDIF_NEXT_BUF_ADDR register was not updated within a frame refresh cycle, LCDIF will keep transmitting the last frame until a new value is programmed into that register.

48.4.1.2 System Bus Master Performance

The performance of the LCDIF block can be controlled by changing the burst length and the outstanding cycle issuing capability depending on the memory bandwidth requirements. Two fields in the LCDIF_CTRL2 register will throttle system memory requests. The LCDIF_CTRL2_OUTSTANDING_REQS field will control how many requests the LCDIF can have in flight on any given clock cycle. This should be programmed based on the expected system bus latency for returned read data. Also, the LCDIF_CTRL2_BURST_LEN_8 bit will set the number of 64 bit words requested for each LCDIF system bus request to either 8 or 16 QWORDS. Generally, 4 outstanding requests of length 16 will provide enough performance to drive any standard display resolution. These configuration bits are intended to change the access pattern of the LCDIF to optimize system bus throughput when other system masters will contend for system memory resources.

48.4.2 Write Data Path

LCDIF supports raster based frame buffers and there is no support for tiled buffers.

There are several options to accommodate endianness of display buffers in memory before the data is processed for the external display. The LCDIF_CTRL[INPUT_DATA_SWIZZLE] field provides the following options for data word multiplexing:

```
00 (0): No swizzle (little-endian)
01 (1): Swap bytes 0 and 3, swap bytes 1 and 2 (big-endian)
10 (2): Swap half-words
11 (3): Swap bytes within each half-word
```

The LCDIF_CTRL[WORD_LENGTH] field indicates the input data/pixel format. LCDIF_TRANSFER_COUNT register denotes how much data is contained in each frame. The LCDIF_TRANSFER_COUNT[H_COUNT] field indicates the number of pixels per line and LCDIF_TRANSFER_COUNT[V_COUNT] indicates the total number of lines per frame. The LCDIF_CTRL1[BYTE_PACKING_FORMAT] field can be used to specify which bytes within the 32-bit word are going to be valid. For example, if the entire 32-bit word is valid, LCDIF_CTRL1[BYTE_PACKING_FORMAT] should be set to 0xF, if only lower 3 bytes of each word in the frame buffer are valid, then LCDIF_CTRL1[BYTE_PACKING_FORMAT] should be set to 0x7.

The LCDIF_CTRL[LCD_DATABUS_WIDTH] field suggests the width of the bus going to the display controller. There is an option to source all 32 bits of the input word and transfer it to the output display interface. If the LCDIF_CTRL[LCD_DATABUS_WIDTH] is not the same as LCDIF_CTRL[WORD_LENGTH], LCDIF will perform RGB to RGB color space conversion. For example, if the input frame has fewer bits per pixel than the display, as in a 16 bpp input frame going to 24 bpp LCD, LCDIF will pad the MSBs of each color to the LSBs of the same color for each pixel. If the input frame has more bits per pixel than the display, for example, 24 bpp input frame going to 16 bpp LCD, LCDIF will drop the LSBs of each color channel to convert to the lower color depth. LCDIF also has the capability to support delta pixel displays by swizzling the R, G and B colors of each pixel in the odd and even lines of the frame separately by programming the LCDIF_CTRL2[ODD_LINE_PATTERN] and the LCDIF_CTRL2[EVEN_LINE_PATTERN] bit fields. This operation occurs after the RGB-to-RGB color space conversion operation.

The following list shows how the different input/output combinations can be obtained:

- LCDIF_CTRL[WORD_LENGTH]=1 indicates that the input is 8-bit data. This is most likely going to be used for a gray scale image. Any combination of LCDIF_CTRL1[BYTE_PACKING_FORMAT] is permissible.

  Limitation: LCDIF_TRANSFER_COUNT[H_COUNT] must be a multiple of the sum of BYTE_PACKING_FORMAT [3], BYTE_PACKING_FORMAT [2], BYTE_PACKING_FORMAT [1] and BYTE_PACKING_FORMAT [0]. LCDIF_CTRL[LCD_DATABUS_WIDTH] must be 1, indicating an 8-bit data bus.

- LCDIF_CTRL[WORD_LENGTH]=0 implies the input frame buffer is RGB 16 bits per pixel. LCDIF_CTRL[DATA_FORMAT_16_BIT] field determines the pixels are RGB 555 or RGB 565.

Limitation: LCDIF_CTRL1[BYTE_PACKING_FORMAT] should be 0x3 or 0xC if there is only one pixel per word. If there are two pixels per word, it should be 0xF and LCDIF_TRANSFER_COUNT[H_COUNT] will be restricted to be a multiple of 2 pixels.

- LCDIF_CTRL[WORD_LENGTH]=2 indicates that input frame buffer is RGB 18 bits per pixel, that is, RGB 666. The valid RGB values can be left-aligned or right-aligned within a 32-bit word. The alignment of the valid 18 bits within a word is indicated by the LCDIF_CTRL[DATA_FORMAT_18_BIT] bit.

  Limitation: LCDIF_CTRL1[BYTE_PACKING_FORMAT] can be 0x7, 0xE or 0xF. Packed pixels are not supported in this case.
  LCDIF_TRANSFER_COUNT[H_COUNT] can be any number.

- LCDIF_CTRL[WORD_LENGTH]=3 indicates that the input frame-buffer is RGB 24 bits per pixel (RGB 888). If LCDIF_CTRL1[BYTE_PACKING_FORMAT] is 0x7, it indicates that there is only one pixel per 32-bit word and there is no restriction on LCDIF_TRANSFER_COUNT[H_COUNT].

  Limitation: If LCDIF_CTRL1[BYTE_PACKING_FORMAT] is 0xF, it indicates that the pixels are packed, that is, there are 4 pixels in 3 words or 12 bytes and LCDIF_TRANSFER_COUNT[H_COUNT] must be a multiple of 4 pixels.

After the RGB to RGB color space conversions, there is one more opportunity to swizzle the data before sending it out to the display or the encoder. This can be done with the LCDIF_CTRL[CSC_DATA_SWIZZLE] field, and it provides the same options as the LCDIF_CTRL[INPUT_DATA_SWIZZLE] register.

Finally, there is an option to shift the output data before sending it out to the display. This is done based on the LCDIF_CTRL[SHIFT_DIR] and LCDIF_CTRL[SHIFT_NUM_BITS] fields.

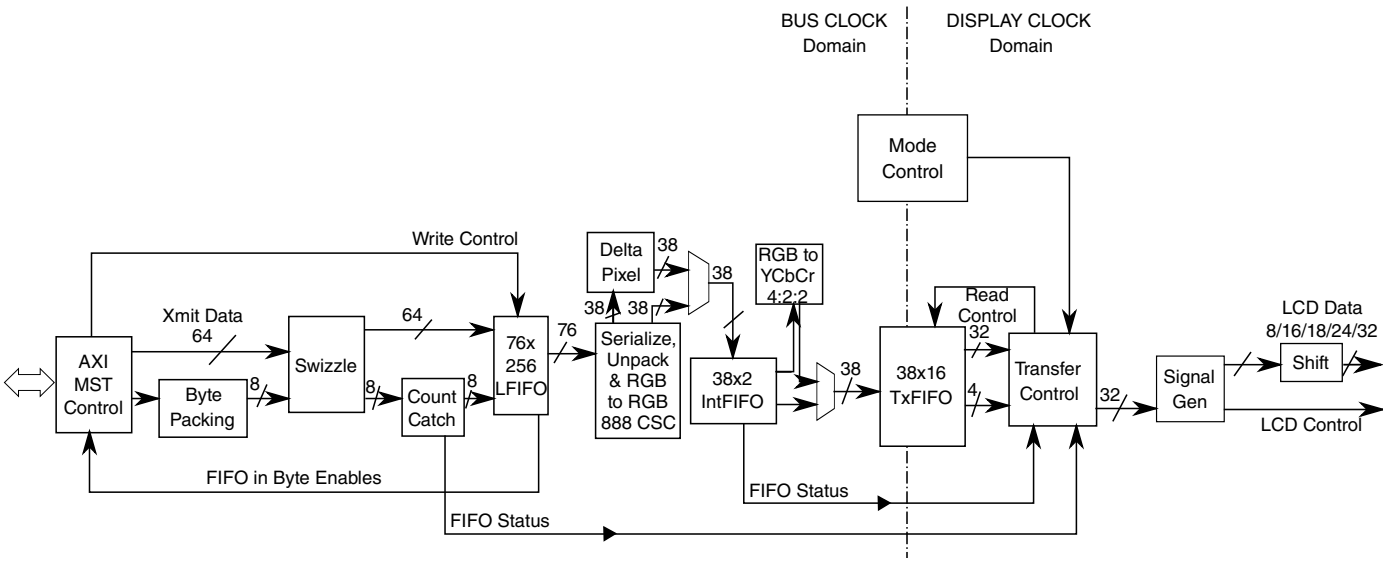


Figure 48-2. General Operations in Write Data Path

The examples in the following figures illustrate some different combinations of register programming for write mode. Assume that the data transferred over the system bus within a 32 bit word is organized as {A7-A0, B7-B0, C7-C0, D7-D0} in 8-bit mode and {A15-A0, B15-B0} in 16-bit mode.

In this example, all 32 bits of the input word are transferred out over an 8 bit display bus. Each byte within the 32 bit word is shifted to the right with zeros appended to bits D[7:6]. The input data bits [7:2] are shifted to the right by 2 bits and presented on the D[5:0].

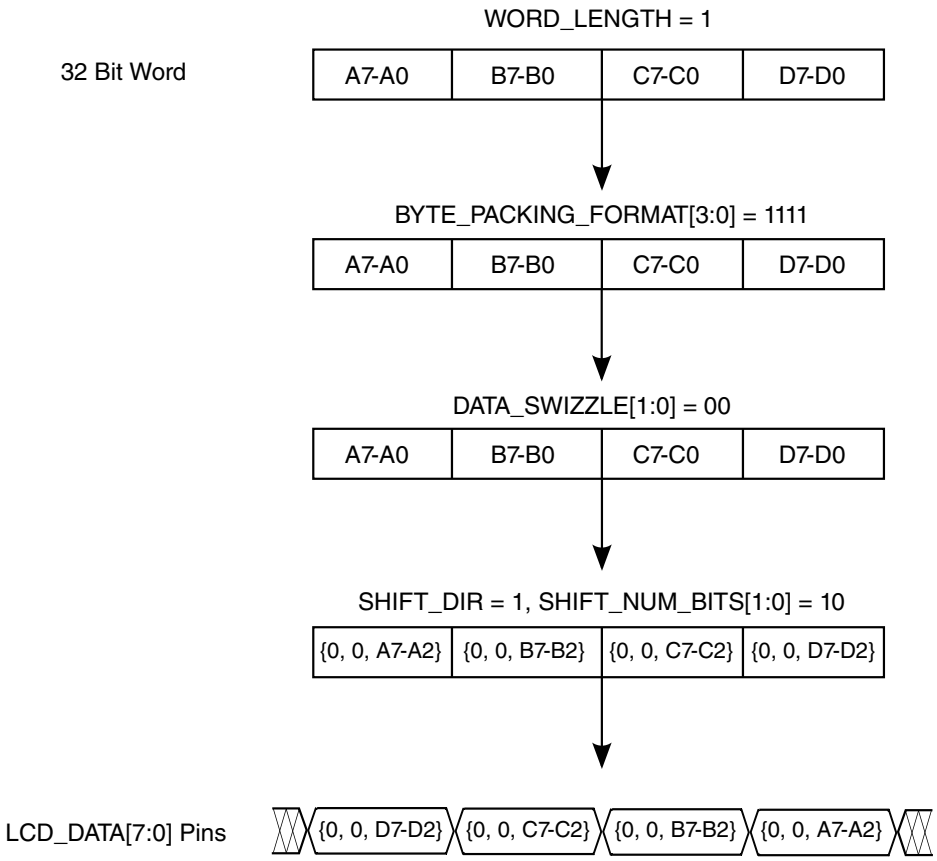


Figure 48-3. Register programming for write mode

In this 8 bit display interface example, one byte of the input word is deleted and not transferred over the external 8 bit display interface. This mode could be used to transfer 24bpp pixels over the 8 bit interface. In this case, the 4th unused byte is not transferred.

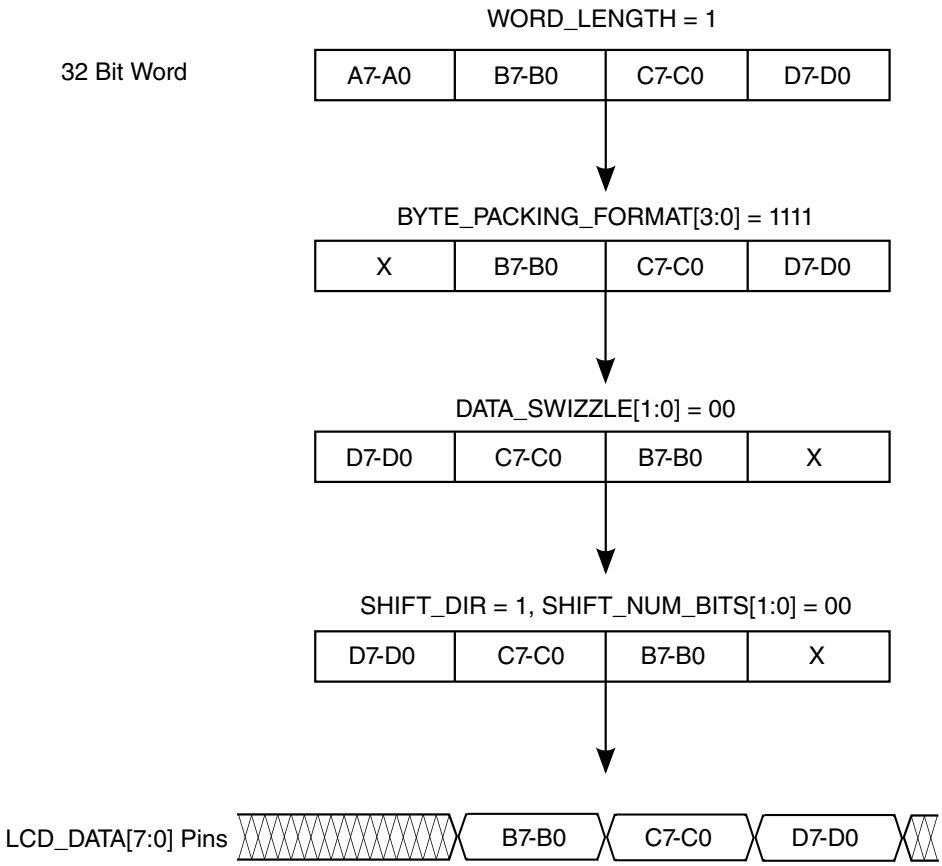


Figure 48-4. Register programming for write mode

The following example uses a 16 bit display interface. Each 16 bit half word is shifted to the right by two bits with zeros appended to the most significant two bits.

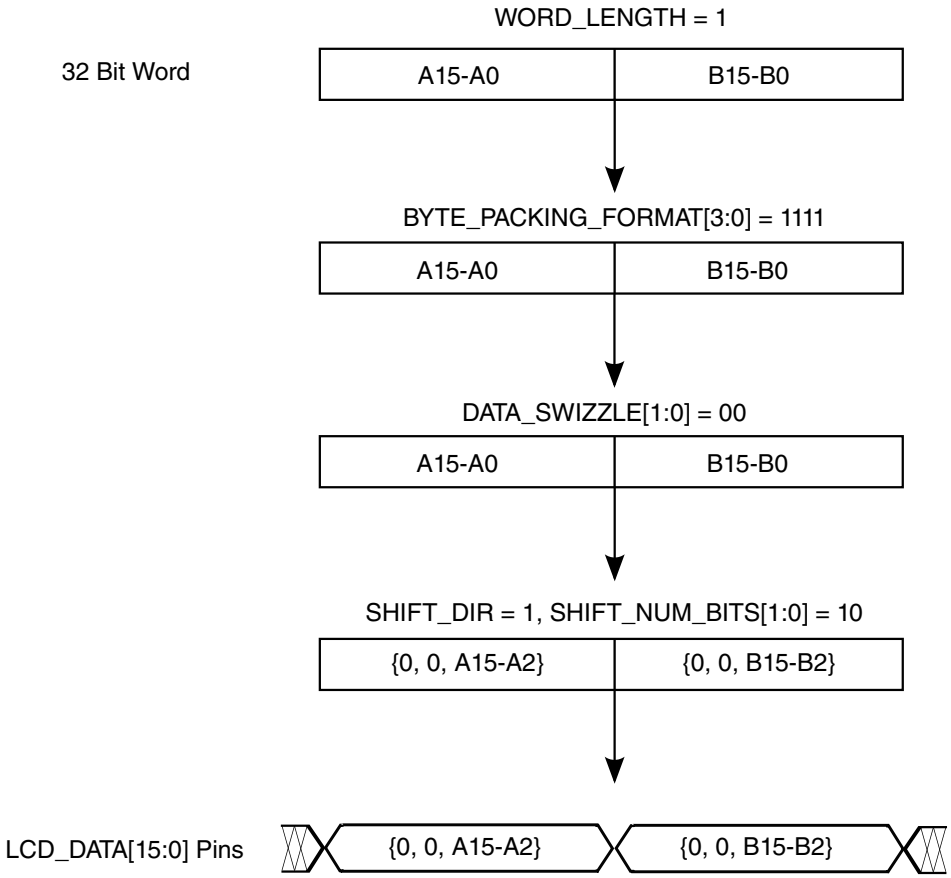


Figure 48-5. Register programming for write mode

This example indicates how an unpacked frame buffer can sourced for display. Only a single 16 bit half word within the 32 bit word is tranferred out via the 16 display bus.

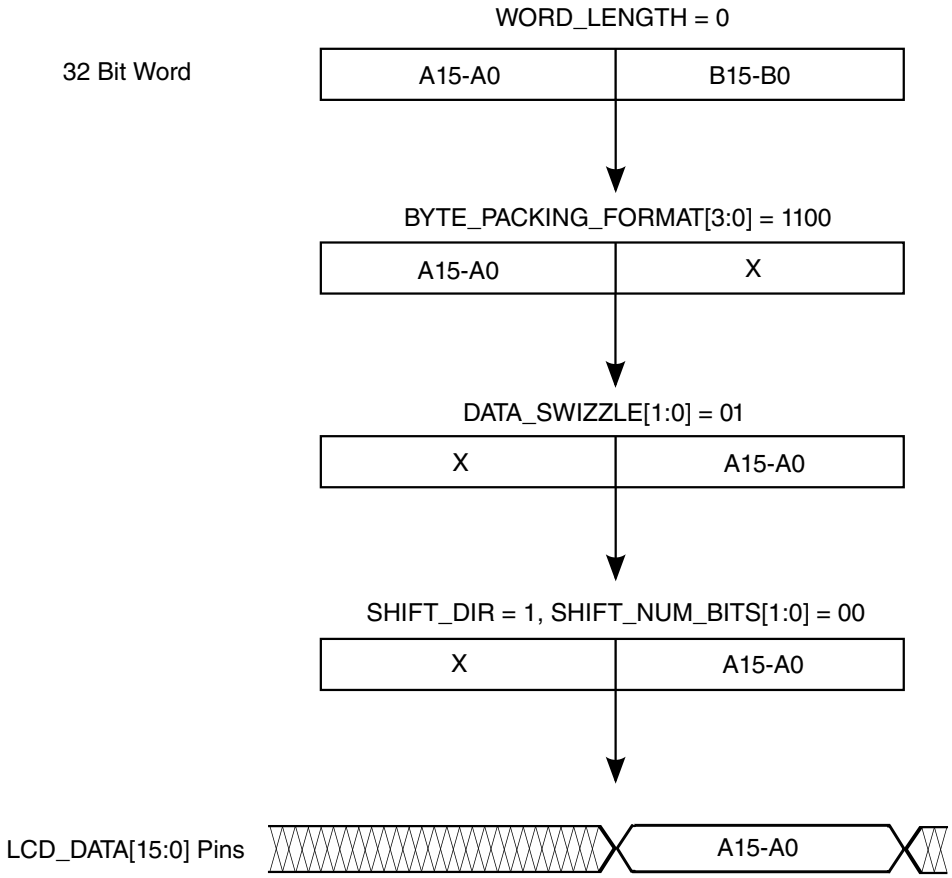


Figure 48-6. Register programming for write mode

48.4.3 LCDIF Interrupts

LCDIF supports a number of interrupts to aid controlling and status reporting of the block.

All the interrupts have individual mask bits for enabling or disabling each of them. They all get funneled through a single interrupt line connected to the interrupt collector (ICOLL).

The following list describes the different interrupts supported by LCDIF:

- Underflow interrupt is asserted when the clock domain crossing FIFO (TXFIFO) becomes empty but the block is in active display portion during that time. Software should take corrective action to make sure that this does not happen.
- In the bus master mode, the overflow interrupt will be asserted if the block has requested more data than it's FIFOs could hold.
- Cur_frame_done interrupt occurs at the end of every frame .

48.4.4 Initializing the LCDIF

This section describes write modes.

48.4.4.1 Write Modes

The following initialization steps are common to all LCDIF write modes of operation before entering any particular mode.

Initialization steps:

1. Configure the external I/Os to correctly interface the external display, when required.
2. Start the DISPLAY CLOCK (pix_clk) clock and set the appropriate frequency by programming the registers in SCG module.
3. Start the BUS CLOCK (apb_clk) and set the appropriate frequency by programming the registers in SCG module.
4. Bring the LCDIF out of soft reset and disable the clock gate bit.
5. Reset the LCD controller by setting LCDIF_CTRL1[RESET] bit appropriately, being careful to observe the reset requirements of the controller. See Behavior During Reset for more information on Reset requirements.
6. Set the transfer mode of operation to bus master. The LCDIF_CTRL[MASTER] bit determines the transfer mode selected. Bus master (LCDIF_CTRL[MASTER] =1) is the transfer mode to select.
7. Set the LCDIF_CTRL[INPUT_DATA_SWIZZLE] according to the endianness of the LCD controller. Also, set the LCDIF_CTRL[DATA_SHIFT_DIR] and LCDIF_CTRL[SHIFT_NUM_BITS] if it is required to shift the data left or right before it is output.
8. Set the LCDIF_CTRL[WORD_LENGTH] field appropriately: 0 = 16-bit input, 1 = 8-bit input, 2 = 18-bit input, 3 = 24/32-bit input. Also, select the correct 16/18/24 bit data format with the corresponding fields in LCDIF_CTRL register.
9. Set the LCDIF_CTRL1[BYTE_PACKING_FORMAT] field according to the input frame.
10. Set the LCDIF_CTRL[LCD_DATABUS_WIDTH] appropriately: 0 = 16-bit output, 1 = 8-bit output, 2 = 18-bit output, 3 = 24/32-bit output.
11. Enable the necessary IRQs.

48.4.5 DOTCLK Interface

The DOTCLK interface is another mode used in moving picture displays.

It includes the VSYNC, HSYNC, DOTCLK and (optional) ENABLE signals. The interface is popularly called the RGB interface if the ENABLE signal is present.

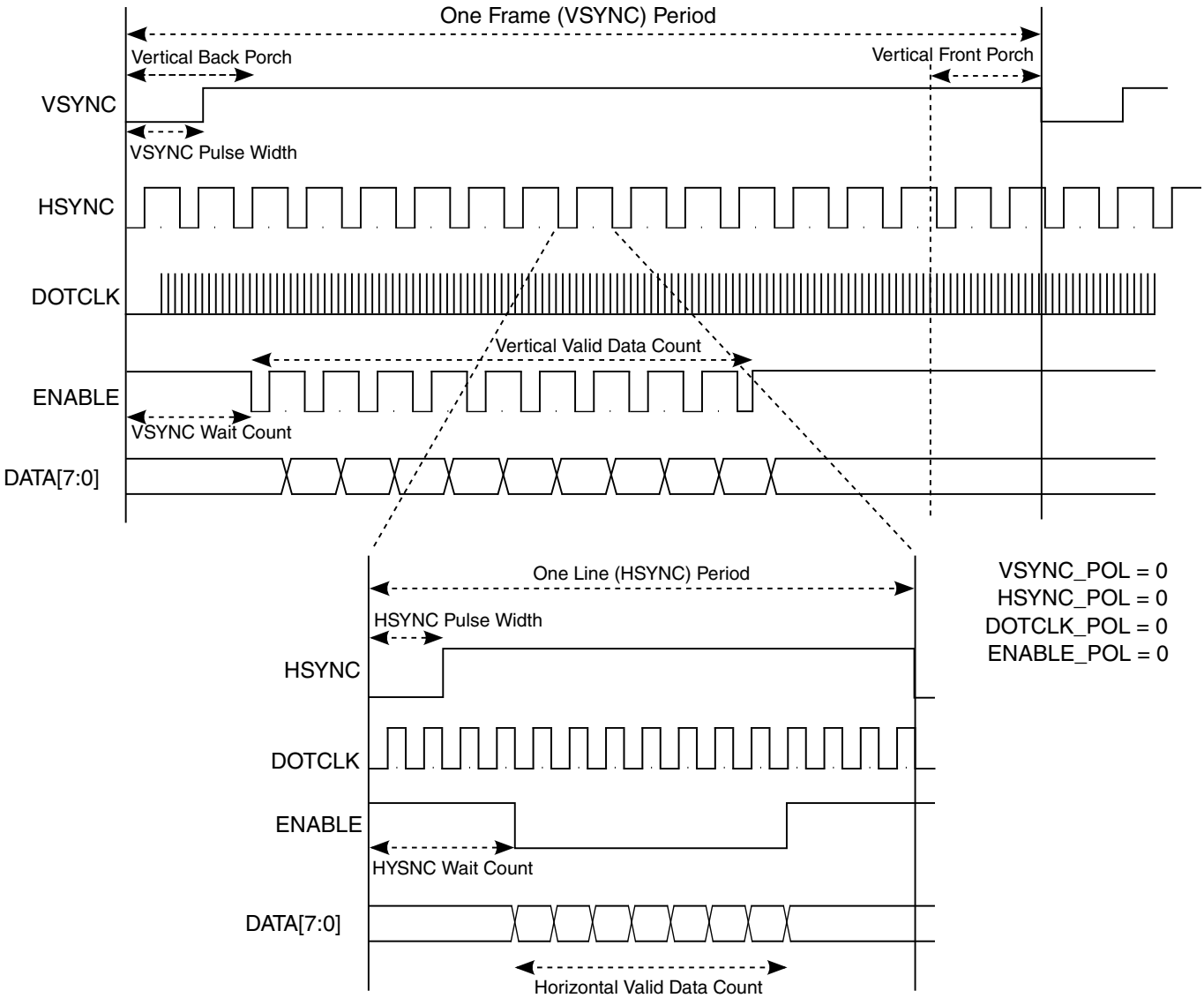


Figure 48-7. DOTCLK protocol with programmable parameters

The DOTCLK mode writes data at high speed to the LCD, and the display operation is synchronized with the VSYNC, HSYNC, ENABLE and DOTCLK signals. The polarities, periods and pulse-widths of the sync signals are programmable using the LCDIF_VDCTRL0-4 registers. The units for the VSYNC signal must be number of horizontal lines and can be selected using the VSYNC_PULSE_WIDTH_UNIT and VSYNC_PERIOD_UNIT bit fields. The VERTICAL_WAIT_CNT is by default given the same unit as the VSYNC_PERIOD. The DISPLAY CLOCK (pix_clk) frequency is managed by the SCG module.

In DOTCLK mode, LCDIF_CTRL_BYPASS_COUNT bit must be set to 1. To end the current transfer, the software should make the DOTCLK_MODE bit 0, so that all data that is currently in the LCDIF LFIFO and TXFIFO is transmitted. Once that transfer is complete, the block will automatically clear the RUN bit and issue the cur_frame_done interrupt.

48.4.5.1 Code Example

The following code shows an example for programming a 320x240 display.

NOTE

Setting up the display must be done via SPI.

```
// Note: Common initialization steps in Initializing the LCDIF must also be
// executed along with the following code
BF_CS1 (LCDIF_CTRL, DOTCLK_MODE, 1);
BF_CS1 (LCDIF_CTRL, BYPASS_COUNT, 1); //Always for DOTCLK mode
BF_CS1 (LCDIF_VDCTRL0, VSYNC_OEB, 0); //Vsync is always an output in the DOTCLK mode
BF_CS4 (LCDIF_VDCTRL0, VSYNC_POL, 0, HSYNC_POL, 0, DOTCLK_POL, 0, ENABLE_POL, 0);
BF_CS1 (LCDIF_VDCTRL0, ENABLE_PRESENT, 1);
BF_CS2 (LCDIF_VDCTRL0, VSYNC_PERIOD_UNIT, 1, VSYNC_PULSE_WIDTH_UNIT, 1);
BF_CS1 (LCDIF_VDCTRL0, VSYNC_PULSE_WIDTH, 2);
BF_CS1 (LCDIF_VDCTRL1, VSYNC_PERIOD, 280);
BF_CS2 (LCDIF_VDCTRL2, HSYNC_PULSE_WIDTH, 10, HSYNC_PERIOD, 360); //Assuming
                                                    //  LCD_DATABUS_WIDTH is 24bit
BF_CS2 (LCDIF_VDCTRL3, VSYNC_ONLY, 0);
BF_CS2 (LCDIF_VDCTRL3, HORIZONTAL_WAIT_CNT, 20, VERTICAL_WAIT_CNT, 20);
BF_CS1 (LCDIF_VDCTRL4, DOTCLK_H_VALID_DATA_CNT, 320);//Note that DOTCLK_V_VALID_DATA_CNT is
                              //implicitly assumed to be HW_LCDIF_TRANSFER_COUNT_V_COUNT
BF_CS1 (LCDIF_VDCTRL4, SYNC_SIGNALS_ON, 1);
BF_CS1 (LCDIF_CTRL, RUN, 1);
```

To stop the transfer completely, the ideal way is to make DOTCLK_MODE = 0. In that case, the block will transmit the contents in the FIFO and reset the RUN bit.

48.4.6 Alpha Blending Interface

The LCDIF have the capability to add an extra overlay on the normal display buffer,LCDIF can fetch data from two buffers and combine them before display,one buffer data can have the alpha value with the RGB pixels. With LCDIF_AS_CTRL[AS_ENABLE] is set, the LCDIF will start fetching alpha surface buffer data in bus master mode and combine it with another buffer.

The LCDIF_AS_CTRL[ALPHA_CTRL] bits determines how the alpha value is constructed for the alpha surface.

48.4.6.1 Alpha Blending/Color Key

Regardless of pixel input format, the PS and AS pixels are normalized to 32-bits, organized as one alpha and three data bytes. Alpha blending occurs in the RGB space, if blending is required, PS pixels should be converted to RGB space. If no alpha blending is required, then YUV pixels can bypass the alpha blending ALU without color space conversion.

48.4.6.2 Alpha Blend

The alpha value for an individual pixel represents a mathematical weighting factor applied to the AS pixel. An alpha value of 0x00 corresponds to a transparent pixel and a value of 0xFF corresponds to an opaque pixel.

The effective alpha value for an AS pixel is determined by the AS_CTRL[ALPHA] and AS_CTRL[ALPHA_CTRL] register fields. If AS_CTRL[ALPHA_CTRL] = ALPHA_OVERRIDE , the alpha value for the pixel is taken from the AS_CTRL[ALPHA] . This can be useful for applying a constant alpha to an entire image or for image formats that don't include an alpha value. If AS_CTRL[ALPHA_CTRL] = ALPHA_MULTIPLY, the pixel's alpha value will be multiplied by the pixel's ALPHA value in order to allow scaling of the pixel's alpha or to provide better control for pixel formats such as RGB1555, which only contains a single bit of alpha.

For each color channel, the equation used to blend two source pixels is defined below:

Gá = PIO programmed global alpha (8-bit value).
Eá = Embedded alpha associated with AS pixel.
á = Gá * Eá + 0x80

The result for the red channel as an example:

R[7:0] = (á * PS.r) + ((1 - á) * AS.r)

When á is 0xff, the PS pixel will not be blended with the AS pixel, but PS will be passed as the output pixel and will not be blended with AS. In this case, AS will be discarded. Likewise, if á is 0x00 for a given pixel, PS will be loaded as the output pixel.

AS_CTRL[ALPHA_INVERT] provides the option to invert the final alpha value. This essentially inverts the effect the alpha value has on the AS and PS blending operation.

48.4.6.3 Color Key

The color key function is provided to create transparent effects on the output pixel.

Color keying is applied on the input pixels after they are converted to 8-bits for each red, green, and blue color channels (color keys are not applied directly to 16-bit pixel formats but to their corresponding 24-bit representation). A color key range is programmable for both PS and AS pixels. If the PS 24-bit pixel is within the PS color key range, then AS is passed through the pixel pipeline. In this case, alpha blending does NOT occur. Conversely, if PS is within the AS color key range, then PS is passed via the LCDIF data pipeline. If both PS and AS color key tests pass, then the back ground color register is passed onto following LCDIF processing components in the pipeline.

The condition for color keying to be satisfied is:

CK0.r.low <= PS.r <= CK0.r.high

CK0.g.low <= PS.g <= CK0.g.high

CK0.b.low <= PS.b <= CK0.b.high

For example, if the "red" 8-bit value for the PS pixel (or PS.r) is between the color key low and high values (CK0.r.l and CK0.r.h), the condition is true for the red color plane. When ALL three color planes meet this condition, then only the PS pixel is loaded into the output register.

To disable color keying, program the low color key register value to 0xff and the high value to 0x00. This will guarantee that the color key range test will never be true.

48.4.6.4 Color Key Processing (AS_CTRL)

The AS_CTRL register also contains an ENABLE_COLORKEY bit that can be used to enable or disable color key substitution for the AS.

When enabled, the pixel values are compared to the ASCOLORKEYLOW and ASCOLORKEYHIGH registers to determine if a match has occurred. When an AS pixel matches the color key range, the pixel from the AS image is considered transparent and the corresponding PS pixel is rendered. If both the PS and AS pixels match their corresponding color key ranges, the AS pixel is displayed unmodified.

AS color keys are handled in a manner similar to PS color keys. The same images used in the PS color key example could be used with the images swapped. In this case, matches on the AS image to the ASCOLORKEY register would display the PS pixels.

48.5 Behavior During Reset

BUS CLOCK (apb_clk) and DISPLAY CLOCK (pix_clk) must be running before making any changes to SFTRST or CLKGATE bits.

A soft reset (SFTRST) can take multiple clock periods to complete, so do not set CLKGATE when setting SFTRST.

The reset process gates the clocks automatically.

48.6 LCDIF Memory Map/Register Definition

Some of the LCDIF registers (XXX_SET, XXX_CLR, and XXX_TOG) allow direct bit field masking and access.

- When writing 1 to XXX_SET bit fields, these registers allow setting the masked 1 bit fields, while keeping unchanged all bit fields which remain on 0 logic state.
- When writing 1 to XXX_CLR bit fields, these registers allow clearing the masked 1 bit fields, while keeping unchanged all other bit fields which remained on 0 logic state.
- When writing 1 to XXX_TOG bit fields, these registers allow inverting the logic state of all masked 1 bit fields, while they keep unchanged the remaining bit fields which were kept on 0 logic state.

LCDIF Hardware Register Format Summary

LCDIF memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AA_0000	LCDIF General Control Register (LCDIF0_CTRL)	32	R/W	C000_0000h	48.6.1/1971
40AA_0004	LCDIF General Control Register (LCDIF0_CTRL_SET)	32	R/W	C000_0000h	48.6.1/1971
40AA_0008	LCDIF General Control Register (LCDIF0_CTRL_CLR)	32	R/W	C000_0000h	48.6.1/1971
40AA_000C	LCDIF General Control Register (LCDIF0_CTRL_TOG)	32	R/W	C000_0000h	48.6.1/1971
40AA_0010	LCDIF General Control1 Register (LCDIF0_CTRL1)	32	R/W	000F_0000h	48.6.2/1973
40AA_0014	LCDIF General Control1 Register (LCDIF0_CTRL1_SET)	32	R/W	000F_0000h	48.6.2/1973
40AA_0018	LCDIF General Control1 Register (LCDIF0_CTRL1_CLR)	32	R/W	000F_0000h	48.6.2/1973
40AA_001C	LCDIF General Control1 Register (LCDIF0_CTRL1_TOG)	32	R/W	000F_0000h	48.6.2/1973
40AA_0020	LCDIF General Control2 Register (LCDIF0_CTRL2)	32	R/W	0020_0000h	48.6.3/1976

Table continues on the next page...

LCDIF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40AA_0024	LCDIF General Control2 Register (LCDIF0_CTRL2_SET)	32	R/W	0020_0000h	48.6.3/1976
40AA_0028	LCDIF General Control2 Register (LCDIF0_CTRL2_CLR)	32	R/W	0020_0000h	48.6.3/1976
40AA_002C	LCDIF General Control2 Register (LCDIF0_CTRL2_TOG)	32	R/W	0020_0000h	48.6.3/1976
40AA_0030	LCDIF Horizontal and Vertical Valid Data Count Register (LCDIF0_TRANSFER_COUNT)	32	R/W	0001_0000h	48.6.4/1978
40AA_0040	LCD Interface Current Buffer Address Register (LCDIF0_CUR_BUF)	32	R/W	0000_0000h	48.6.5/1978
40AA_0050	LCD Interface Next Buffer Address Register (LCDIF0_NEXT_BUF)	32	R/W	0000_0000h	48.6.6/1979
40AA_0070	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF0_VDCTRL0)	32	R/W	0000_0000h	48.6.7/1979
40AA_0074	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF0_VDCTRL0_SET)	32	R/W	0000_0000h	48.6.7/1979
40AA_0078	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF0_VDCTRL0_CLR)	32	R/W	0000_0000h	48.6.7/1979
40AA_007C	LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF0_VDCTRL0_TOG)	32	R/W	0000_0000h	48.6.7/1979
40AA_0080	LCDIF VSYNC Mode and Dotclk Mode Control Register1 (LCDIF0_VDCTRL1)	32	R/W	0000_0000h	48.6.8/1981
40AA_0090	LCDIF VSYNC Mode and Dotclk Mode Control Register2 (LCDIF0_VDCTRL2)	32	R/W	0000_0000h	48.6.9/1981
40AA_00A0	LCDIF VSYNC Mode and Dotclk Mode Control Register3 (LCDIF0_VDCTRL3)	32	R/W	0000_0000h	48.6.10/ 1982
40AA_00B0	LCDIF VSYNC Mode and Dotclk Mode Control Register4 (LCDIF0_VDCTRL4)	32	R/W	0000_0000h	48.6.11/ 1983
40AA_0190	Bus Master Error Status Register (LCDIF0_BM_ERROR_STAT)	32	R/W	0000_0000h	48.6.12/ 1984
40AA_01A0	CRC Status Register (LCDIF0_CRC_STAT)	32	R/W	0000_0000h	48.6.13/ 1984
40AA_01B0	LCD Interface Status Register (LCDIF0_STAT)	32	R	9500_0000h	48.6.14/ 1985
40AA_0210	LCDIF AS Buffer Control Register (LCDIF0_AS_CTRL)	32	R/W	0000_0000h	48.6.15/ 1987
40AA_0220	Alpha Surface Buffer Pointer (LCDIF0_AS_BUF)	32	R/W	0000_0000h	48.6.16/ 1989
40AA_0230	LCDIF0_AS_NEXT_BUF	32	R/W	0000_0000h	48.6.17/ 1989
40AA_0240	LCDIF Overlay Color Key Low (LCDIF0_AS_CLRKEYLOW)	32	R/W	00FF_FFFFh	48.6.18/ 1990
40AA_0250	LCDIF Overlay Color Key High (LCDIF0_AS_CLRKEYHIGH)	32	R/W	0000_0000h	48.6.19/ 1990

48.6.1 LCDIF General Control Register (LCDIFx_CTRLn)

The LCD Interface Control Register provides overall control of the LCDIF block. The LCDIF Control Register provides a variety of control functions to the programmer. These functions allow the interface to be very flexible to work with a variety of LCD controllers, and to minimize overhead and increase performance of LCD programming. The register has been organized such that switching between the different LCD modes can be done with minimum PIO writes.

Address: 40AA_0000h base + 0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SFTRST	CLKGATE	-	Reserved	Reserved	DATA_SHIFT_DIR	SHIFT_NUM_BITS					Reserved	BYPASS_COUNT	Reserved	DOTCLK_MODE	Reserved
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INPUT_ DATA_ SWIZZLE		CSC_DATA_SWIZZLE		LCD_ DATABUS_ WIDTH		WORD_ LENGTH		Reserved	Reserved	MASTER	Reserved	DATA_FORMAT_16_BIT	DATA_FORMAT_18_BIT	DATA_FORMAT_24_BIT	RUN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_CTRLn field descriptions

Field	Description
31 SFTRST	This bit must be set to zero to enable normal operation of the LCDIF. When set to one, it forces a block level reset.
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clocks to the block.
29 -	Reserved
28 -	This field is reserved. Reserved

Table continues on the next page...

LCDIFx_CTRLn field descriptions (continued)

Field	Description
27 -	This field is reserved. Reserved
26 DATA_SHIFT_DIR	Use this bit to determine the direction of shift of transmit data. 0x0 **TXDATA_SHIFT_LEFT** — Data to be transmitted is shifted LEFT by SHIFT_NUM_BITS bits. 0x1 **TXDATA_SHIFT_RIGHT** — Data to be transmitted is shifted RIGHT by SHIFT_NUM_BITS bits.
25–21 SHIFT_NUM_BITS	The data to be transmitted is shifted left or right by this number of bits.
20 -	This field is reserved. Reserved
19 BYPASS_COUNT	When this bit is 0, it means that LCDIF will stop the block operation and turn off the RUN bit after the amount of data indicated by the LCDIF_TRANSFER_COUNT register has been transferred out. When this bit is set to 1, the block will continue normal operation indefinitely until it is told to stop. This bit must be 0 in MPU and VSYNC modes, and must be 1 in DOTCLK and DVI modes of operation.
18 -	This field is reserved. Reserved
17 DOTCLK_MODE	Set this bit to 1 to make the hardware go into the DOTCLK mode, i.e. VSYNC/HSYNC/DOTCLK/ENABLE interface mode. ENABLE is optional, selected by the ENABLE_PRESENT bit. Toggle this bit from 1 to 0 to make the hardware go out of DOTCLK mode after completing all data transfer and deasserting the RUN bit.
16 -	This field is reserved. Reserved
15–14 INPUT_DATA_SWIZZLE	This field specifies how to swap the bytes fetched by the bus master interface. The swizzle function is independent of the WORD_LENGTH bit. The supported swizzle configurations are: 0x0 **NO_SWAP** — No byte swapping.(Little endian) 0x0 **LITTLE_ENDIAN** — Little Endian byte ordering (same as NO_SWAP). 0x1 **BIG_ENDIAN_SWAP** — Big Endian swap (swap bytes 0,3 and 1,2). 0x1 **SWAP_ALL_BYTES** — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 **HWD_SWAP** — Swap half-words. 0x3 **HWD_BYTE_SWAP** — Swap bytes within each half-word.
13–12 CSC_DATA_SWIZZLE	This field specifies how to swap the bytes after the data has been converted into an internal representation of 24 bits per pixel and before it is transmitted over the LCD interface bus. The data is always transmitted with the least significant byte/hword (half word) first after the swizzle takes place. So, INPUT_DATA_SWIZZLE takes place first on the incoming data, and then CSC_DATA_SWIZZLE is applied. The swizzle function is independent of the WORD_LENGTH or the LCD_DATABUS_WIDTH fields. If RGB_TO_YCRCB422_CSC bit is set, the swizzle occurs on the Y, Cb, Cr values. The supported swizzle configurations are: 0x0 **NO_SWAP** — No byte swapping.(Little endian) 0x0 **LITTLE_ENDIAN** — Little Endian byte ordering (same as NO_SWAP). 0x1 **BIG_ENDIAN_SWAP** — Big Endian swap (swap bytes 0,3 and 1,2). 0x1 **SWAP_ALL_BYTES** — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 **HWD_SWAP** — Swap half-words. 0x3 **HWD_BYTE_SWAP** — Swap bytes within each half-word.
11–10 LCD_DATABUS_WIDTH	LCD Data bus transfer width. 0x0 **16_BIT** — 16-bit data bus mode.

Table continues on the next page...

LCDIF*x*_CTRL*n* field descriptions (continued)

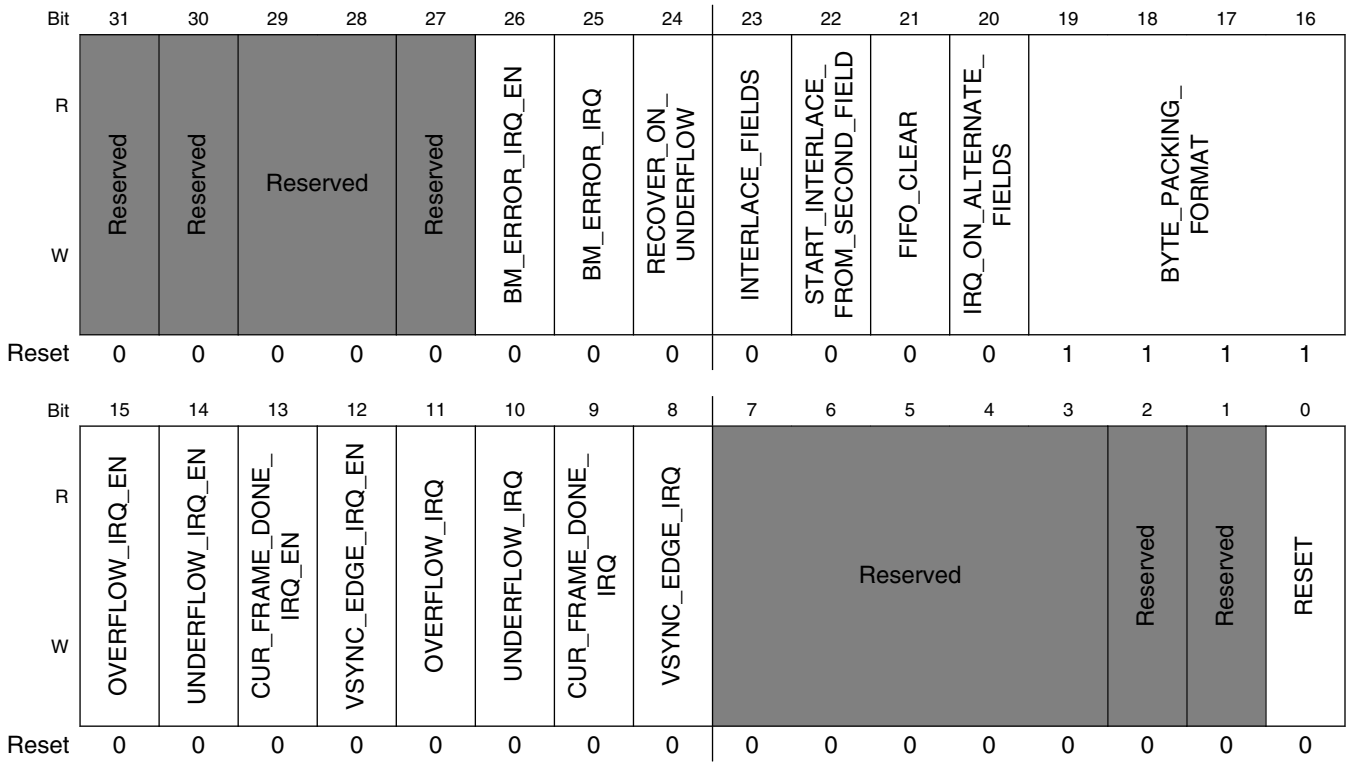
Field	Description
	0x1 **8_BIT** — 8-bit data bus mode. 0x2 **18_BIT** — 18-bit data bus mode. 0x3 **24_BIT** — 24-bit data bus mode.
9–8 WORD_LENGTH	Input data format. 0x0 **16_BIT** — Input data is 16 bits per pixel. 0x1 **8_BIT** — Input data is 8 bits wide. 0x2 **18_BIT** — Input data is 18 bits per pixel. 0x3 **24_BIT** — Input data is 24 bits per pixel.
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5 MASTER	Set this bit to make the LCDIF act as a bus master.
4 RSRVD0	This field is reserved. Reserved bits. Write as 0.
3 DATA_ FORMAT_16_ BIT	When this bit is 1 and WORD_LENGTH = 0, it implies that the 16-bit data is in ARGB555 format. When this bit is 0 and WORD_LENGTH = 0, it implies that the 16-bit data is in RGB565 format. When WORD_LENGTH is not 0, this bit does not care.
2 DATA_ FORMAT_18_ BIT	Used only when WORD_LENGTH = 2, i.e. 18-bit. 0x0 **LOWER_18_BITS_VALID** — Data input to the block is in 18 bpp format, such that lower 18 bits contain RGB 666 and upper 14 bits do not contain any useful data. 0x1 **UPPER_18_BITS_VALID** — Data input to the block is in 18 bpp format, such that upper 18 bits contain RGB 666 and lower 14 bits do not contain any useful data.
1 DATA_ FORMAT_24_ BIT	Used only when WORD_LENGTH = 3, i.e. 24-bit. Note that this applies to both packed and unpacked 24-bit data. 0x0 **ALL_24_BITS_VALID** — Data input to the block is in 24 bpp format, such that all RGB 888 data is contained in 24 bits. 0x1 **DROP_UPPER_2_BITS_PER_BYTE** — Data input to the block is actually RGB 18 bpp, but there is 1 color per byte, hence the upper 2 bits in each byte do not contain any useful data, and should be dropped.
0 RUN	When this bit is set by software, the LCDIF will begin transferring data between the SoC and the display. This bit must remain set until the operation is complete.

48.6.2 LCDIF General Control1 Register (LCDIF*x*_CTRL1*n*)

The LCDIF Control Register provides overall control of the LCDIF block.

The LCDIF Control1 Register provides additional programming to the LCDIF. It implements some bits which are unlikely to change often in a particular application. It also carries interrupt-related bits which are common across more than one mode of operation.

Address: 40AA_0000h base + 10h offset + (4d × i), where i=0d to 3d



LCDIFx_CTRL1*n* field descriptions

Field	Description
31 -	This field is reserved. Reserved bits. Write as 0.
30 -	This field is reserved. Reserved bits. Write as 0.
29–28 -	This field is reserved. Reserved bits. Write as 0.
27 -	This field is reserved. Reserved.
26 BM_ERROR_IRQ_EN	This bit is set to enable bus master error interrupt in the LCDIF master mode.
25 BM_ERROR_IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. This bit will be set when the LCDIF is in master mode and an error response was returned by the slave. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
24 RECOVER_ON_UNDERFLOW	Set this bit to enable the LCDIF block to recover in the next field/frame if there was an underflow in the current field/frame.
23 INTERLACE_FIELDS	Set this bit if it is required that the LCDIF block fetches odd lines in one field and even lines in the other field. It will work only in LCDIF_MASTER is set to 1.

Table continues on the next page...

LCDIF*x*_CTRL1*n* field descriptions (continued)

Field	Description
22 START_ INTERLACE_ FROM_ SECOND_FIELD	The default is to grab the odd lines first and then the even lines. Set this bit if it is required to grab the even lines first and then the odd lines. (Line numbers start from 1, so odd lines are 1,3,5,etc. and even lines are 2,4,6, etc.)
21 FIFO_CLEAR	Set this bit to clear all the data in the latency FIFO (LFIFO), TXFIFO and the RXFIFO.
20 IRQ_ON_ ALTERNATE_ FIELDS	If this bit is set, the LCDIF block will assert the cur_frame_done interrupt only on alternate fields, otherwise it will issue the interrupt on both odd and even field. This bit is mostly relevant if INTERLACE_FIELDS is set.
19–16 BYTE_ PACKING_ FORMAT	This bitfield is used to show which data bytes in a 32-bit word are valid. Default value 0xf indicates that all bytes are valid. For 8-bit transfers, any combination in this bitfield will mean valid data is present in the corresponding bytes. In the 16-bit mode, a 16-bit half-word is valid only if adjacent bits [1:0] or [3:2] or both are 1. A value of 0x0 will mean that none of the bytes are valid and should not be used. For example, set the bit field value to 0x7 if the display data is arranged in the 24-bit unpacked format (A-R-G-B where A value does not have be transmitted).
15 OVERFLOW_ IRQ_EN	This bit is set to enable an overflow interrupt in the TXFIFO in the write mode.
14 UNDERFLOW_ IRQ_EN	This bit is set to enable an underflow interrupt in the TXFIFO in the write mode.
13 CUR_FRAME_ DONE_IRQ_EN	This bit is set to 1 enable an interrupt every time the hardware enters in the vertical blanking state.
12 VSYNC_EDGE_ IRQ_EN	This bit is set to enable an interrupt every time the hardware encounters the leading VSYNC edge in the VSYNC and DOTCLK modes, or the beginning of every field in DVI mode.
11 OVERFLOW_ IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. A latency FIFO (LFIFO) overflow in the write mode (MPU/VSYNC/DOTCLK/DVI mode) was detected, data samples have been lost. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
10 UNDERFLOW_ IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. A TXFIFO underflow in the write mode (MPU/VSYNC/DOTCLK/DVI mode) was detected. Could produce an error in the DOTCLK / DVI modes. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
9 CUR_FRAME_ DONE_IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. It indicates that the hardware has completed transmitting the current frame and is in the vertical blanking period in the DOTCLK/DVI modes. In the MPU and VSYNC modes, this IRQ is asserted at the end of the data transfer indicated by LCDIF_TRANSFER_COUNT register. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.

Table continues on the next page...

LCDIF*x*_CTRL1*n* field descriptions (continued)

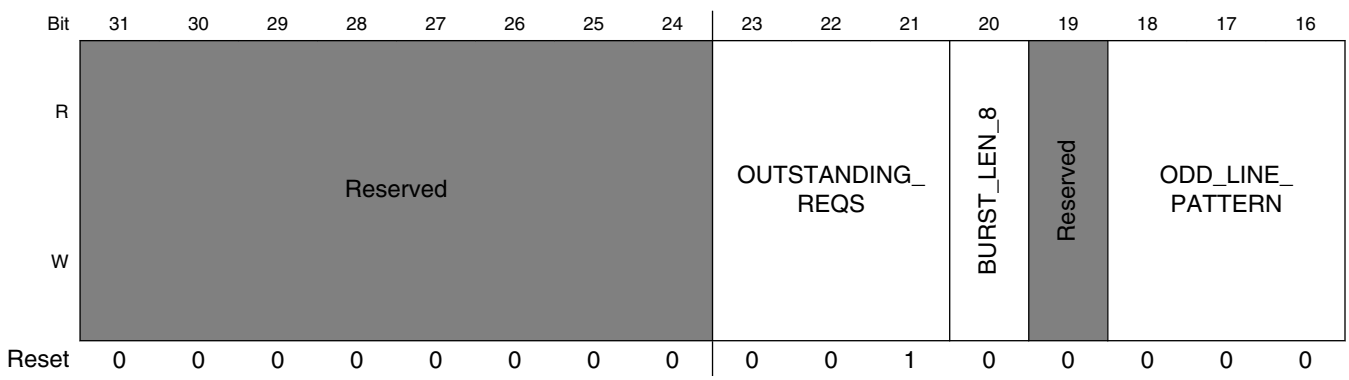
Field	Description
8 VSYNC_EDGE_IRQ	This bit is set to indicate that an interrupt is requested by the LCDIF block. This bit is cleared by software by writing a one to its SCT clear address. It is set whenever the leading VSYNC edge is detected in the VSYNC and DOTCLK modes. In the DVI mode, it is asserted every time the block enters a new field. 0x0 **NO_REQUEST** — No Interrupt Request Pending. 0x1 **REQUEST** — Interrupt Request Pending.
7–3 RSRVD0	This field is reserved. Reserved bits. Write as 0.
2 -	This field is reserved. Reserved
1 -	This field is reserved. Reserved.
0 RESET	Reset bit for the external LCD controller. This bit can be changed at any time. It CANNOT be reset by SFTRST. 0x0 **LCDRESET_LOW** — LCD_RESET output signal is low. 0x1 **LCDRESET_HIGH** — LCD_RESET output signal is high.

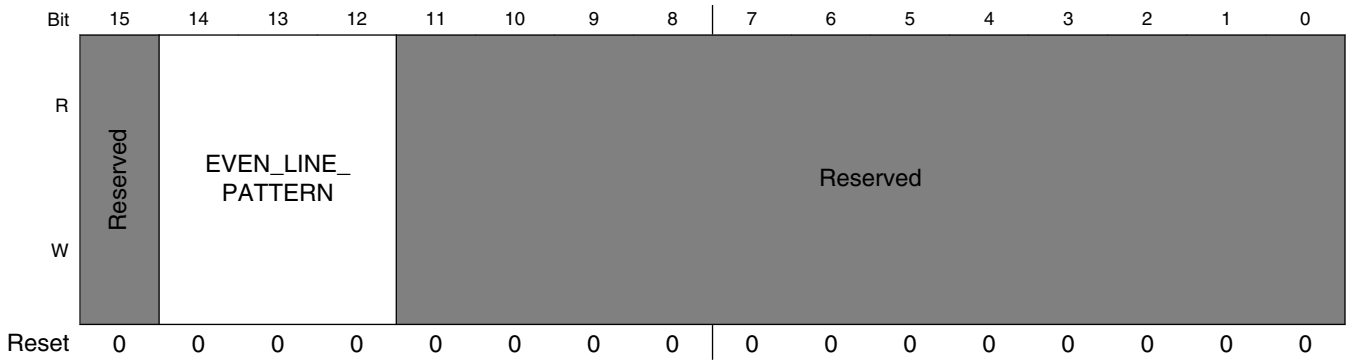
48.6.3 LCDIF General Control2 Register (LCDIF*x*_CTRL2*n*)

The LCDIF Control Register provides overall control of the LCDIF block.

The LCDIF Control2 Register provides additional programming to the LCDIF. It implements some bits which are unlikely to change often in a particular application.

Address: 40AA_0000h base + 20h offset + (4d × i), where i=0d to 3d





LCDIFx_CTRL2*n* field descriptions

Field	Description
31–24 RSRVD5	This field is reserved. Reserved bits. Write as 0.
23–21 OUTSTANDING_ REQS	This bitfield indicates the maximum number of outstanding transactions that LCDIF should request when it is acting as a bus master. Default is 2 outstanding transactions. 0x0 **REQ_1** — 0x1 **REQ_2** — 0x2 **REQ_4** — 0x3 **REQ_8** — 0x4 **REQ_16** —
20 BURST_LEN_8	By default, when the LCDIF is in the bus master mode, it will issue AXI bursts of length 16 (except when in packed 24 bpp mode, it will issue bursts of length 15). When this bit is set to 1, the block will issue bursts of length 8 (except when in packed 24 bpp mode, it will issue bursts of length 9). Note that this bitfield is only applicable when LCDIF_MASTER is set to 1.
19 RSRVD4	This field is reserved. Reserved bits. Write as 0.
18–16 ODD_LINE_ PATTERN	This field determines the order of the RGB components of each pixel in ODD lines (line numbers 1,3,5,..). 0x0 **RGB** — 0x1 **RBG** — 0x2 **GBR** — 0x3 **GRB** — 0x4 **BRG** — 0x5 **BGR** —
15 RSRVD3	This field is reserved. Reserved bits. Write as 0.
14–12 EVEN_LINE_ PATTERN	This field determines the order of the RGB components of each pixel in EVEN lines (line numbers 2,4,6,..). 0x0 **RGB** — 0x1 **RBG** — 0x2 **GBR** — 0x3 **GRB** — 0x4 **BRG** — 0x5 **BGR** —

Table continues on the next page...

LCDIFx_CTRL2n field descriptions (continued)

Field	Description
RSRVD0	This field is reserved. Reserved bits. Write as 0.

48.6.4 LCDIF Horizontal and Vertical Valid Data Count Register (LCDIFx_TRANSFER_COUNT)

This register tells the LCDIF how much data will be sent for this frame, or transaction. The total number of words is a product of the V_COUNT and H_COUNT fields. The word size is specified by the WORD_LENGTH field.

This register gives the dimensions of the input frame. For normal operation, but V_COUNT and H_COUNT should be non-zero.

Address: 40AA_0000h base + 30h offset = 40AA_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	V_COUNT																H_COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_TRANSFER_COUNT field descriptions

Field	Description
31–16 V_COUNT	Number of horizontal lines per frame which contain valid data. In DOTCLK mode, V_COUNT should be the same as the number of active horizontal lines in a progressive frame.
H_COUNT	Total valid data (pixels) in each horizontal line. The data size is given by the WORD_LENGTH. In 24-bit packed format (WORD_LENGTH=0x3, BYTE_PACKING_FORMAT=0xF), the H_COUNT must be a multiple of 4 pixels. In 16-bit packed format (WORD_LENGTH=0x0, BYTE_PACKING_FORMAT=0xF), the H_COUNT must be a multiple of 2 pixels.

48.6.5 LCD Interface Current Buffer Address Register (LCDIFx_CUR_BUF)

This register indicates the address of the current frame being transmitted by LCDIF.

When the LCDIF is behaving as a master, this address points to the address of the current frame of data being sent out via the LCDIF. When the current frame is done, the LCDIF block will assert the cur_frame_done interrupt for software to take action. The block will also copy the LCDIF_NEXT_BUF_ADDR into this bitfield so that the software can program the next frame address into the LCDIF_NEXT_BUF_ADDR bitfield. This address must always be double-word aligned.

Address: 40AA_0000h base + 40h offset = 40AA_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF*x*_CUR_BUF field descriptions

Field	Description
ADDR	Address of the current frame being transmitted by LCDIF.

48.6.6 LCD Interface Next Buffer Address Register (LCDIF*x*_NEXT_BUF)

This register indicates the address of next frame that will be transmitted by LCDIF.

When the LCDIF is behaving as a master, this address points to the address of the next frame of data that will be sent out via the LCDIF. It is up to the software to make sure that this register is programmed before the end of the current frame, otherwise it might result in old data going out the LCDIF. This address must always be double-word aligned.

Address: 40AA_0000h base + 50h offset = 40AA_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIF*x*_NEXT_BUF field descriptions

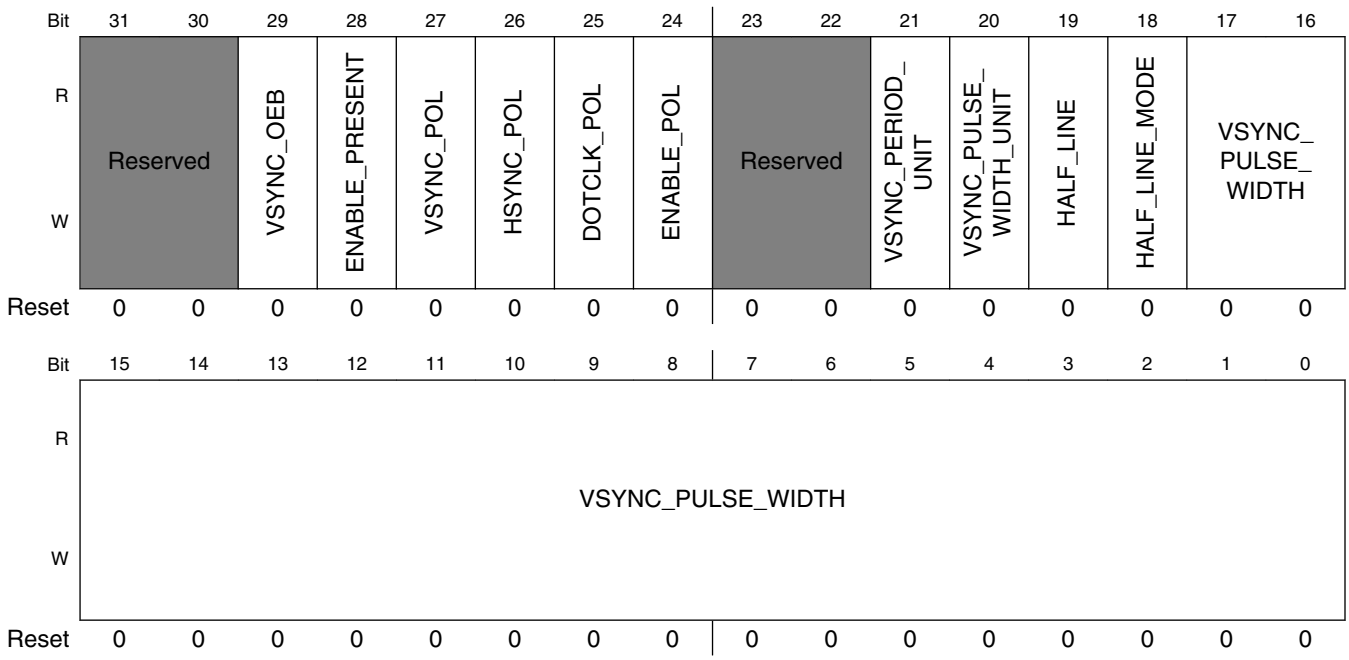
Field	Description
ADDR	Address of the next frame that will be transmitted by LCDIF.

48.6.7 LCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF*x*_VDCTRL0*n*)

This register is used to control the VSYNC and DOTCLK modes of the LCDIF so as to work with different types of LCDs like moving picture displays and delta pixel displays.

This register gives general programmability to the VSYNC signal including polarity, direction, pulse width, etc.

Address: 40AA_0000h base + 70h offset + (4d × i), where i=0d to 3d



LCDIFx_VDCTRL0*n* field descriptions

Field	Description
31–30 RSRVD2	This field is reserved. Reserved bits. Write as 0.
29 VSYNC_OEB	0 means the VSYNC signal is an output, 1 means it is an input. Should be set to 0 in the DOTCLK mode. 0x0 **VSYNC_OUTPUT** — The VSYNC pin is in the output mode and the VSYNC signal has to be generated by the LCDIF block. 0x1 **VSYNC_INPUT** — The VSYNC pin is in the input mode and the LCD controller sends the VSYNC signal to the block.
28 ENABLE_ PRESENT	Setting this bit to 1 will make the hardware generate the ENABLE signal in the DOTCLK mode, thereby making it the true RGB interface along with the remaining three signals VSYNC, HSYNC and DOTCLK.
27 VSYNC_POL	Default 0 active low during VSYNC_PULSE_WIDTH time and will be high during the rest of the VSYNC period. Set it to 1 to invert the polarity.
26 HSYNC_POL	Default 0 active low during HSYNC_PULSE_WIDTH time and will be high during the rest of the HSYNC period. Set it to 1 to invert the polarity.
25 DOTCLK_POL	Default is data launched at negative edge of DOTCLK and captured at positive edge. Set it to 1 to invert the polarity. Set it to 0 in DVI mode.
24 ENABLE_POL	Default 0 active low during valid data transfer on each horizontal line.
23–22 RSRVD1	This field is reserved. Reserved bits. Write as 0.
21 VSYNC_ PERIOD_UNIT	Default 0 for counting VSYNC_PERIOD in terms of DISPLAY CLOCK (pix_clk) cycles. Set it to 1 to count in terms of complete horizontal lines. DISPLAY CLOCK (pix_clk) cycles should be used in the VSYNC mode, while horizontal line should be used in the DOTCLK mode.
20 VSYNC_PULSE_ WIDTH_UNIT	Default 0 for counting VSYNC_PULSE_WIDTH in terms of DISPLAY CLOCK (pix_clk) cycles. Set it to 1 to count in terms of complete horizontal lines.

Table continues on the next page...

LCDIFx_VDCTRL0*n* field descriptions (continued)

Field	Description
19 HALF_LINE	Setting this bit to 1 will make the total VSYNC period equal to the VSYNC_PERIOD field plus half the HORIZONTAL_PERIOD field (i.e. VSYNC_PERIOD field plus half horizontal line), otherwise it is just VSYNC_PERIOD. Should be only used in the DOTCLK mode, not in the VSYNC interface mode.
18 HALF_LINE_ MODE	When this bit is 0, the first field (VSYNC period) will end in half a horizontal line and the second field will begin with half a horizontal line. When this bit is 1, all fields will end with half a horizontal line, and none will begin with half a horizontal line.
VSYNC_PULSE_ WIDTH	Number of units for which VSYNC signal is active. For the DOTCLK mode, the unit is determined by the VSYNC_PULSE_WIDTH_UNIT. If the VSYNC_PULSE_WIDTH_UNIT is 0 for DOTCLK mode, VSYNC_PULSE_WIDTH must be less than HSYNC_PERIOD. For the VSYNC interface mode, it should be in terms of number of DISPLAY CLOCK (pix_clk) cycles only.

48.6.8 LCDIF VSYNC Mode and Dotclk Mode Control Register1 (LCDIFx_VDCTRL1)

This register is used to control the VSYNC signal in the VSYNC and DOTCLK modes of the block.

This register determines the period and duty cycle of the VSYNC signal when it is generated in the block.

Address: 40AA_0000h base + 80h offset = 40AA_0080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VSYNC_PERIOD																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_VDCTRL1 field descriptions

Field	Description
VSYNC_PERIOD	Total number of units between two positive or two negative edges of the VSYNC signal. If HALF_LINE is set, it is implicitly calculated to be VSYNC_PERIOD plus half HSYNC_PERIOD.

48.6.9 LCDIF VSYNC Mode and Dotclk Mode Control Register2 (LCDIFx_VDCTRL2)

This register is used to control the HSYNC signal in the DOTCLK mode of the block.

This register determines the period and duty cycle of the HSYNC signal when it is generated in the block.

Address: 40AA_0000h base + 90h offset = 40AA_0090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HSYNC_PULSE_WIDTH																HSYNC_PERIOD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_VDCTRL2 field descriptions

Field	Description
31–18 HSYNC_PULSE_ WIDTH	Number of DISPLAY CLOCK (pix_clk) cycles for which HSYNC signal is active.
HSYNC_PERIOD	Total number of DISPLAY CLOCK (pix_clk) cycles between two positive or two negative edges of the HSYNC signal.

48.6.10 LCDIF VSYNC Mode and Dotclk Mode Control Register3 (LCDIFx_VDCTRL3)

This register is used to determine the vertical and horizontal wait counts.

This register determines the back porches of HSYNC and VSYNC signals when they are generated by the block.

Address: 40AA_0000h base + A0h offset = 40AA_00A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		MUX_SYNC_ SIGNALS	VSYNC_ONLY	HORIZONTAL_WAIT_CNT											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VERTICAL_WAIT_CNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_VDCTRL3 field descriptions

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.

Table continues on the next page...

LCDIFx_VDCTRL3 field descriptions (continued)

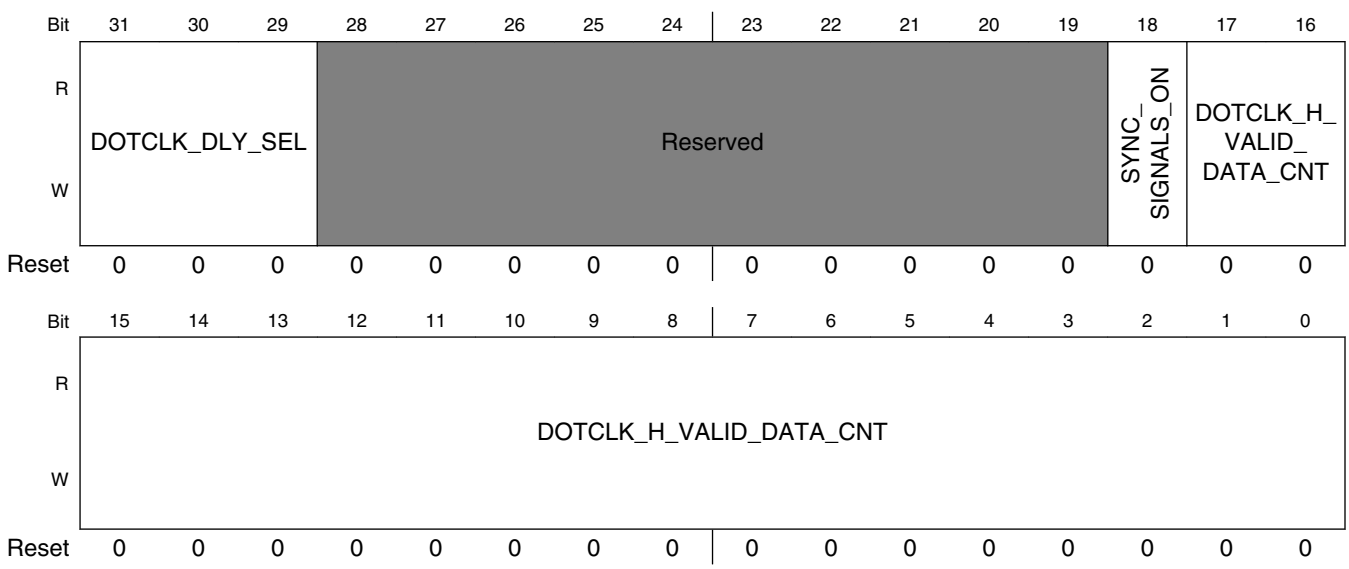
Field	Description
29 MUX_SYNC_ SIGNALS	When this bit is set, the LCDIF block will internally mux HSYNC with LCD_D14, DOTCLK with LCD_D13 and ENABLE with LCD_D12, otherwise these signals will go out on separate pins. This feature can be used to maintain backward compatible with 37xx.
28 VSYNC_ONLY	This bit must be set to 1 in the VSYNC mode of operation, and 0 in the DOTCLK mode of operation.
27–16 HORIZONTAL_ WAIT_CNT	In the DOTCLK mode, wait for this number of clocks from falling edge (or rising if HSYNC_POL is 1) of HSYNC signal to account for horizontal back porch plus the number of DOTCLKs before the moving picture information begins.
VERTICAL_ WAIT_CNT	In the VSYNC interface mode, wait for this number of DISPLAY CLOCK (pix_clk) cycles from the falling VSYNC edge (or rising if VSYNC_POL is 1) before starting LCD transactions and is applicable only if WAIT_FOR_VSYNC_EDGE is set. Minimum is CMD_SETUP+5. In the DOTCLK mode, it accounts for the vertical back porch lines plus the number of horizontal lines before the moving picture begins. The unit for this parameter is inherently the same as the VSYNC_PERIOD_UNIT.

48.6.11 LCDIF VSYNC Mode and Dotclk Mode Control Register4 (LCDIFx_VDCTRL4)

This register is used to control the DOTCLK mode of the block.

This register determines the active data in each horizontal line in the DOTCLK mode. Note that the total number of active horizontal lines in the DOTCLK mode is the same as the V_COUNT bitfield in the LCDIF_TRANSFER_COUNT register.

Address: 40AA_0000h base + B0h offset = 40AA_00B0h



LCDIFx_VDCTRL4 field descriptions

Field	Description
31–29 DOTCLK_DLY_SEL	This bitfield selects the amount of time by which the DOTCLK signal should be delayed before coming out of the LCD_DOTCK pin. 0 = 2ns; 1=4ns;2=6ns;3=8ns. Remaining values are reserved.
28–19 RSRVD0	This field is reserved. Reserved bits, write as 0.
18 SYNC_SIGNALS_ON	Set this field to 1 if the LCD controller requires that the VSYNC or VSYNC/HSYNC/DOTCLK control signals should be active at least one frame before the data transfers actually start and remain active at least one frame after the data transfers end. The hardware does not count the number of frames automatically. Rather, the VSYNC edge interrupt can be monitored by software to count the number of frames that have occurred after this bit is set and then the RUN bit can be set to start the data transactions. This bit must always be set in the DOTCLK mode of operation, and it must be set in the VSYNC mode of operation when VSYNC signal is an output.
DOTCLK_H_VALID_DATA_CNT	Total number of DISPLAY CLOCK (pix_clk) cycles on each horizontal line that carry valid data in DOTCLK mode.

48.6.12 Bus Master Error Status Register (LCDIFx_BM_ERROR_STAT)

This register reflects the virtual address at which the AXI master received an error response from the slave.

When the BM_ERROR_IRQ is asserted, the address of the bus error is updated in the register.

Address: 40AA_0000h base + 190h offset = 40AA_0190h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_BM_ERROR_STAT field descriptions

Field	Description
ADDR	Virtual address at which bus master error occurred.

48.6.13 CRC Status Register (LCDIFx_CRC_STAT)

This register reflects the CRC value of each frame sent out by LCDIF. The CRC is done on the final output bus, so the value will be dependent on the LCD_DATABUS_WIDTH bitfield even if the input data is the same.

This register will be updated when the CUR_FRAME_DONE_IRQ is asserted.

Address: 40AA_0000h base + 1A0h offset = 40AA_01A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CRC_VALUE																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_CRC_STAT field descriptions

Field	Description
CRC_VALUE	Calculated CRC value.

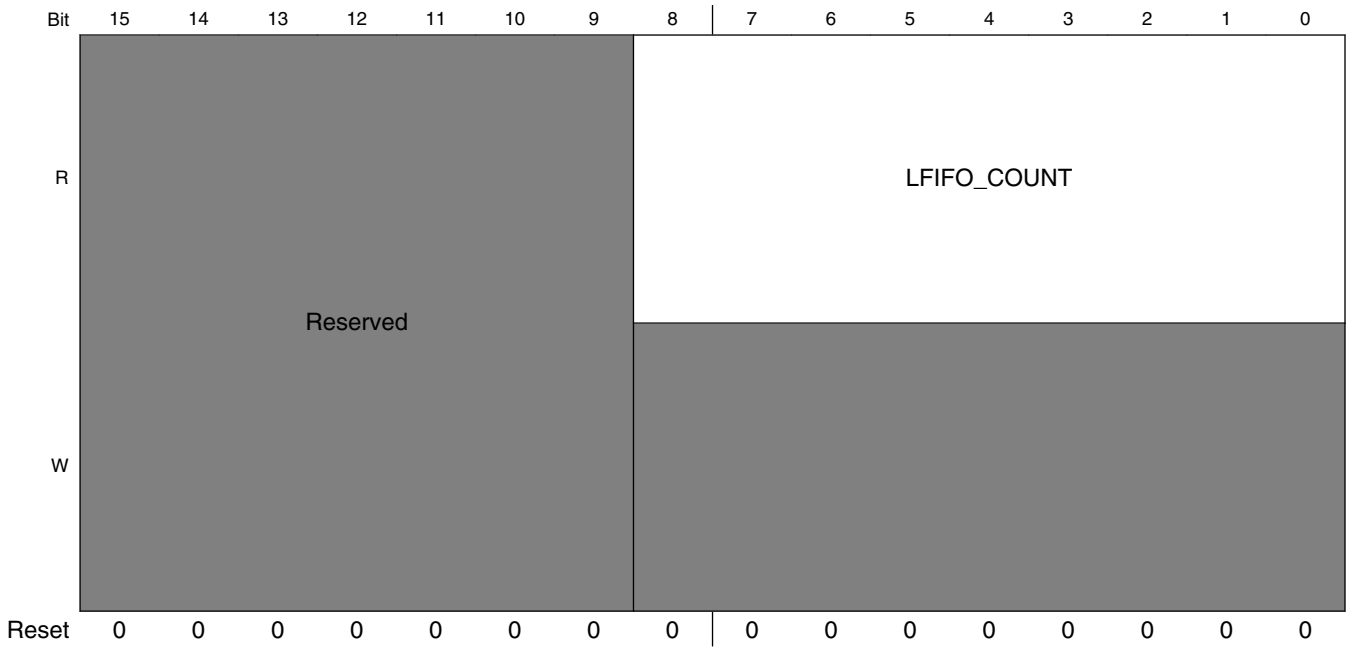
48.6.14 LCD Interface Status Register (LCDIFx_STAT)

The LCD interface status register can be used to check the current status of the LCDIF block.

The LCD interface status register that contains read only views of some parameters or current state of the block.

Address: 40AA_0000h base + 1B0h offset = 40AA_01B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRESENT	Reserved	LFIFO_FULL	LFIFO_EMPTY	TXFIFO_FULL	TXFIFO_EMPTY	Reserved	Reserved								
W																
Reset	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0



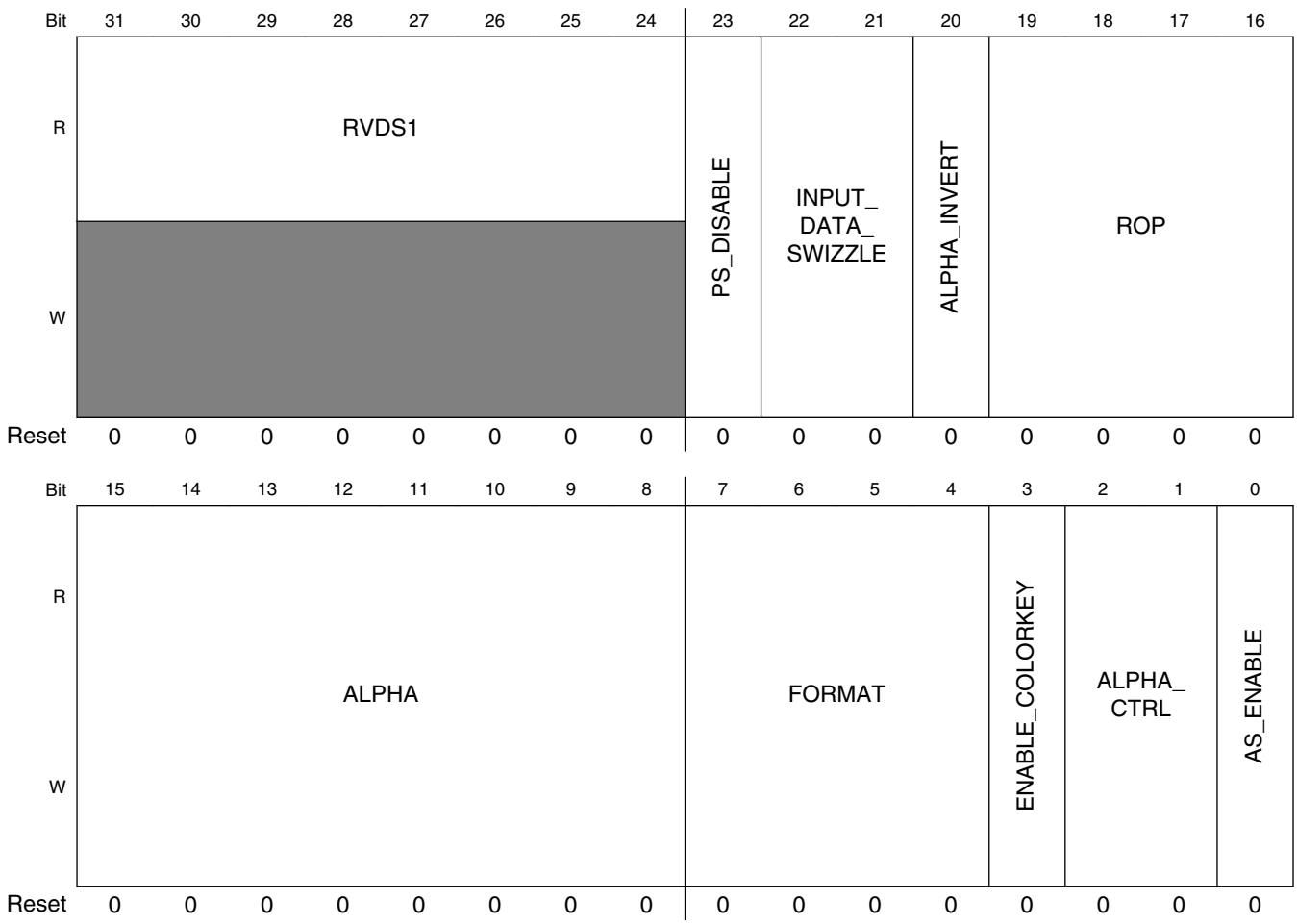
LCDIFx_STAT field descriptions

Field	Description
31 PRESENT	0: LCDIF not present on this product 1: LCDIF is present.
30 -	This field is reserved. Reserved.
29 LFIFO_FULL	Read only view of the signals that indicates LCD LFIFO is full.
28 LFIFO_EMPTY	Read only view of the signals that indicates LCD LFIFO is empty.
27 TXFIFO_FULL	Read only view of the signals that indicates LCD TXFIFO is full.
26 TXFIFO_EMPTY	Read only view of the signals that indicates LCD TXFIFO is empty.
25 -	This field is reserved. Reserved
24–9 RSRVD0	This field is reserved. Reserved bits. Write as 0.
LFIFO_COUNT	Read only view of the current count in Latency buffer (LFIFO).

48.6.15 LCDIF AS Buffer Control Register (LCDIFx_AS_CTRL)

The Alpha Surface Parameter register provides additional controls for AS.

Address: 40AA_0000h base + 210h offset = 40AA_0210h



LCDIFx_AS_CTRL field descriptions

Field	Description
31–24 RVDS1	Reserved, always set to zero.
23 PS_DISABLE	When this bit is set by software, the LCDIF will disable PS buffer data.
22–21 INPUT_DATA_ SWIZZLE	This field specifies how to swap the bytes either in the HW_LCDIF_DATA register or those fetched by the AXI master part of LCDIF. The swizzle function is independent of the WORD_LENGTH bit. See the explanation of the HW_LCDIF_DATA below for names and definitions of data register fields. The supported swizzle configurations are: NO_SWAP = 0x0 No byte swapping.(Little endian)

Table continues on the next page...

LCDIFx_AS_CTRL field descriptions (continued)

Field	Description
	LITTLE_ENDIAN = 0x0 Little Endian byte ordering (same as NO_SWAP). BIG_ENDIAN_SWAP = 0x1 Big Endian swap (swap bytes 0, 3 and 1, 2). SWAP_ALL_BYTES = 0x1 Swizzle all bytes, swap bytes 0, 3 and 1, 2 (aka Big Endian). HWD_SWAP = 0x2 Swap half-words. HWD_BYTE_SWAP = 0x3 Swap bytes within each half-word.
20 ALPHA_INVERT	Setting this bit to logic 0 will not alter the alpha value. A logic 1 will invert the alpha value and apply (1-alpha) for image composition.
19–16 ROP	Indicates a raster operation to perform when enabled. Raster operations are enabled through the ALPHA_CTRL field. MASKAS = 0x0 AS AND PS MASKNOTAS = 0x1 nAS AND PS MASKASNOT = 0x2 AS AND nPS MERGEAS = 0x3 AS OR PS MERGENOTAS = 0x4 nAS OR PS MERGEASNOT = 0x5 AS OR nPS NOTCOPYAS = 0x6 nAS NOT = 0x7 nPS NOTMASKAS = 0x8 AS NAND PS NOTMERGEAS = 0x9 AS NOR PS XORAS = 0xA AS XOR PS NOTXORAS = 0xB AS XNOR PS
15–8 ALPHA	Alpha modifier used when the ALPHA_MULTIPLY or ALPHA_OVERRIDE values are programmed in REG_AS_CTRL[ALPHA_CTRL]. The output alpha value will either be replaced (ALPHA_OVERRIDE) or scaled (ALPHA_MULTIPLY) when selected.
7–4 FORMAT	Indicates the input buffer format for AS. ARGB8888 = 0x0 32-bit pixels with alpha RGB888 = 0x4 32-bit pixels without alpha (unpacked 24-bit format) ARGB1555 = 0x8 16-bit pixels with alpha ARGB4444 = 0x9 16-bit pixels with alpha RGB555 = 0xC 16-bit pixels without alpha RGB444 = 0xD 16-bit pixels without alpha RGB565 = 0xE 16-bit pixels without alpha
3 ENABLE_ COLORKEY	Indicates that colorkey functionality is enabled for this alpha surface. Pixels found in the alpha surface colorkey range will be displayed as transparent (the PS pixel will be used).
2–1 ALPHA_CTRL	Determines how the alpha value is constructed for this alpha surface. Indicates that the value in the ALPHA field should be used instead of the alpha values present in the input pixels. Embedded = 0x0 Indicates that the AS pixel alpha value will be used to blend the AS with PS. The ALPHA field is ignored. Override = 0x1 Indicates that the value in the ALPHA field should be used instead of the alpha values present in the input pixels.

Table continues on the next page...

LCDIFx_AS_CTRL field descriptions (continued)

Field	Description
	Multiply = 0x2 Indicates that the value in the ALPHA field should be used to scale all pixel alpha values. Each pixel alpha is multiplied by the value in the ALPHA field. ROPs = 0x3 Enable ROPs. The ROP field indicates an operation to be performed on the alpha surface and PS pixels.
0 AS_ENABLE	When this bit is set by software, the LCDIF will start fetching AS buffer data in bus master mode and combine it with another buffer.

48.6.16 Alpha Surface Buffer Pointer (LCDIFx_AS_BUF)

This register is used to indicate the base address of the AS buffer.

Address: 40AA_0000h base + 220h offset = 40AA_0220h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_AS_BUF field descriptions

Field	Description
ADDR	Address pointer for the alpha surface 0 buffer.

48.6.17 LCDIFx_AS_NEXT_BUF

When the LCDIF is behaving as a master, this address points to the address of the next frame of data that will be sent out via the LCDIF. It is upto the software to make sure that this register is programmed before the end of the current frame, otherwise it might result in old data going out the LCDIF. This address must always be double-word aligned.

Address: 40AA_0000h base + 230h offset = 40AA_0230h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_AS_NEXT_BUF field descriptions

Field	Description
ADDR	Address of the next frame that will be transmitted by LCDIF.

48.6.18 LCDIF Overlay Color Key Low (LCDIFx_AS_CLRKEYLOW)

If a pixel in the current overlay image with a color that falls in the range from the ASCOLORKEYLOW to ASCOLORKEYHIGH range, it will use the PS pixel value for that location. Colorkey operations are higher priority than alpha or ROP operations.

Address: 40AA_0000h base + 240h offset = 40AA_0240h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RSVD1								PIXEL																							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

LCDIFx_AS_CLRKEYLOW field descriptions

Field	Description
31–24 RSVD1	Reserved, always set to zero.
PIXEL	Low range of RGB color key applied to AS buffer

48.6.19 LCDIF Overlay Color Key High (LCDIFx_AS_CLRKEYHIGH)

If a pixel in the current overlay image with a color that falls in the range from the ASCOLORKEYLOW to ASCOLORKEYHIGH range, it will use the PS pixel value for that location. Colorkey operations are higher priority than alpha or ROP operations.

Address: 40AA_0000h base + 250h offset = 40AA_0250h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RSVD1								PIXEL																							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LCDIFx_AS_CLRKEYHIGH field descriptions

Field	Description
31–24 RSVD1	Reserved, always set to zero.
PIXEL	High range of RGB color key applied to AS buffer

Chapter 49
Video Interface Unit (VIU)

49.1 Chip-specific VIU information

Table 49-1. Reference links to related information

Topic	Related module	Reference
Full description	VIU	VIU
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

49.1.1 Video Interface Unit (VIU)

The Video Interface Unit (VIU) provides a parallel interface for digital video. The VIU accepts 8-bit ITU656 compatible video input, 18-bit/24-bit RGB888/RGB666/RGB565 digital input, and 24-bit YUV444 input on its parallel interface, decodes it and optionally performs processes such as down-scaling, horizontal up-scaling, brightness and contrast adjustment, YUV to RGB conversion, deinterlacing and horizontal mirroring. The resultant video stream is then stored to system memory for subsequent post-processing and display by the Display Control Unit. The following table shows the configuration of the VIU.

It uses 64-bit AHB master bus and IP bus

Table 49-2. VIU configuration

Parameter	Description
Name	Video Interface Unit (VIU)
Instances	1
Configurable features	• Supports up to 24-bit • VGA configuration support only

Table continues on the next page...

Table 49-2. VIU configuration (continued)

Parameter	Description
	• VIUY 1024x8 • VIUU 1024x8 • VIUV 1024x8 • BC RAM 64x32 • FIFO RAM 256x64
Interface speed	66.7 MHz[1]
External I/O pins	PCLK, VSYNC, HSYNC, FID, D[23:0]. See the attached IOMUXC spreadsheet for pin details

1. This is the value of pix_clk

49.2 Introduction

The VIU is a bridge between video decoder and system memory. It accepts an ITU656 compatible video stream, or parallel YUV/RGB input, converting image data between YUV and RGB color spaces, scaling image size, adjusting the brightness/contrast, and writing image data to memory with many kinds of image data formats.

The figure below shows a block diagram of the Video-In (VIU).

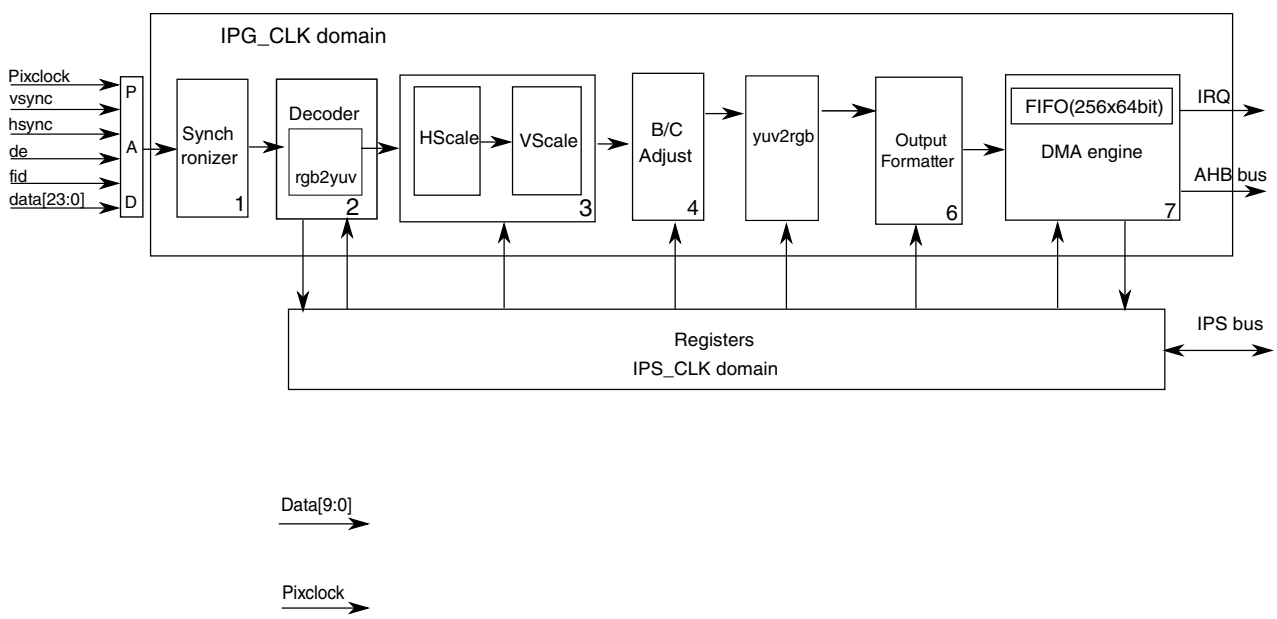


Figure 49-1. VIU Block Diagram

NOTE

To support full function of VIU, IPG_CLK frequency should be bigger than 3 times of Pixclock.

49.3 Features

- Supports QVGA to XGA input resolution
- 8-/10-bit ITU656 video input[1]
- RGB888/RGB666/RGB565 parallel input
- RGB888 serial input with one color component transferred per clock
- 24-bit YUV parallel input from analog video decoder
- 8-bit mono image input
- Multiple output formats: 8-/16-32-bit per pixel in RAM
- Up to 1/8 video down-scaling on horizontal and vertical direction
- Up to 2/1 horizontal video up-scaling
- Horizontal mirroring
- Brightness/contrast adjust
- YUV to RGB 888/565 conversion
- RGB to YUV conversion when input is RGB
- Simple de-interlace function (weaving) for interlaced or pseudo interlaced video input
- Internal DMA engine for transferring data from FIFO to system memory

49.4 Video Input Signal Mapping

Table 49-3. Video Input Signals

Signal	RGB565	RGB666	RGB888	RGB888 (serial)	ITU656 (10-bit)	ITU656 (8-bit)	Parallel YUV (24-bit)	Mono (8-bit)
ipp_pix_data[23]	R4	R5	R7	—	—	—	Y7	Y7
ipp_pix_data[22]	R3	R4	R6	—	—	—	Y6	Y6
ipp_pix_data[21]	R2	R3	R5	—	—	—	Y5	Y5
ipp_pix_data[20]	R1	R2	R4	—	—	—	Y4	Y4
ipp_pix_data[19]	R0	R1	R3	—	—	—	Y3	Y3
ipp_pix_data[18]	0	R0	R2	—	—	—	Y2	Y2
ipp_pix_data[17]	0	0	R1	—	—	—	Y1	Y1

Table continues on the next page...

1. When scaling and/or B/C adjust is enabled, the two LSB's of the 10-bit input are ignored.

Table 49-3. Video Input Signals (continued)

Signal	RGB565	RGB666	RGB888	RGB888 (serial)	ITU656 (10-bit)	ITU656 (8-bit)	Parallel YUV (24-bit)	Mono (8-bit)
ipp_pix_data[16]	0	0	R0	—	—	—	Y0	Y0
ipp_pix_data[15]	G5	G5	G7	—	—	—	U7	—
ipp_pix_data[14]	G4	G4	G6	—	—	—	U6	—
ipp_pix_data[13]	G3	G3	G5	—	—	—	U5	—
ipp_pix_data[12]	G2	G2	G4	—	—	—	U4	—
ipp_pix_data[11]	G1	G1	G3	—	—	—	U3	—
ipp_pix_data[10]	G0	G0	G2	—	—	—	U2	—
ipp_pix_data[9]	0	0	G1	—	Y9/C9	Y7/C7	U1	—
ipp_pix_data[8]	0	0	G0	—	Y8/C8	Y6/C6	U0	—
ipp_pix_data[7]	B4	B5	B7	R7/G7/B7	Y7/C7	Y5/C5	V7	—
ipp_pix_data[6]	B3	B4	B6	R6/G6/B6	Y6/C6	Y4/C4	V6	—
ipp_pix_data[5]	B2	B3	B5	R5/G5/C5	Y5/C5	Y3/C3	V5	—
ipp_pix_data[4]	B1	B2	B4	R4/G4/C4	Y4/C4	Y2/C2	V4	—
ipp_pix_data[3]	B0	B1	B3	R3/G3/C3	Y3/C3	Y1/C1	V3	—
ipp_pix_data[2]	0	B0	B2	R2/G2/C2	Y2/C2	Y0/C0	V2	—
ipp_pix_data[1]	0	0	B1	R1/G1/C1	Y1/C1	0	V1	—
ipp_pix_data[0]	0	0	B0	R0/G0/C0	Y0/C0	0	V0	—
ipp_ind_pix_clk	CLK	CLK	CLK	CLK	CLK	CLK	CLK	CLK
ipp_ind_pix_hs	HSYNC	HSYNC	HSYNC	HSYNC	—	—	HSYNC	HSYNC
ipp_ind_pix_vs	VSYNC	VSYNC	VSYNC	VSYNC	—	—	VSYNC	VSYNC
ipp_ind_pix_de	DE	DE	DE	DE	—	—	DE	DE
ipp_ind_pix_fid	FID	FID	FID	FID	—	—	FID	FID

49.5 Memory map and register definition

The memory map for the VIU module is given below. The total address for each register is the sum of the base address for the VIU module and the address offset for each register. Also, please note that the bit order for each register bit field is [highest:lowest], whereas the overall register bits are numbered [lowest:highest].

49.5.1 VIU register descriptions

49.5.1.1 VIU Memory map

VIU base address: 40A8_0000h

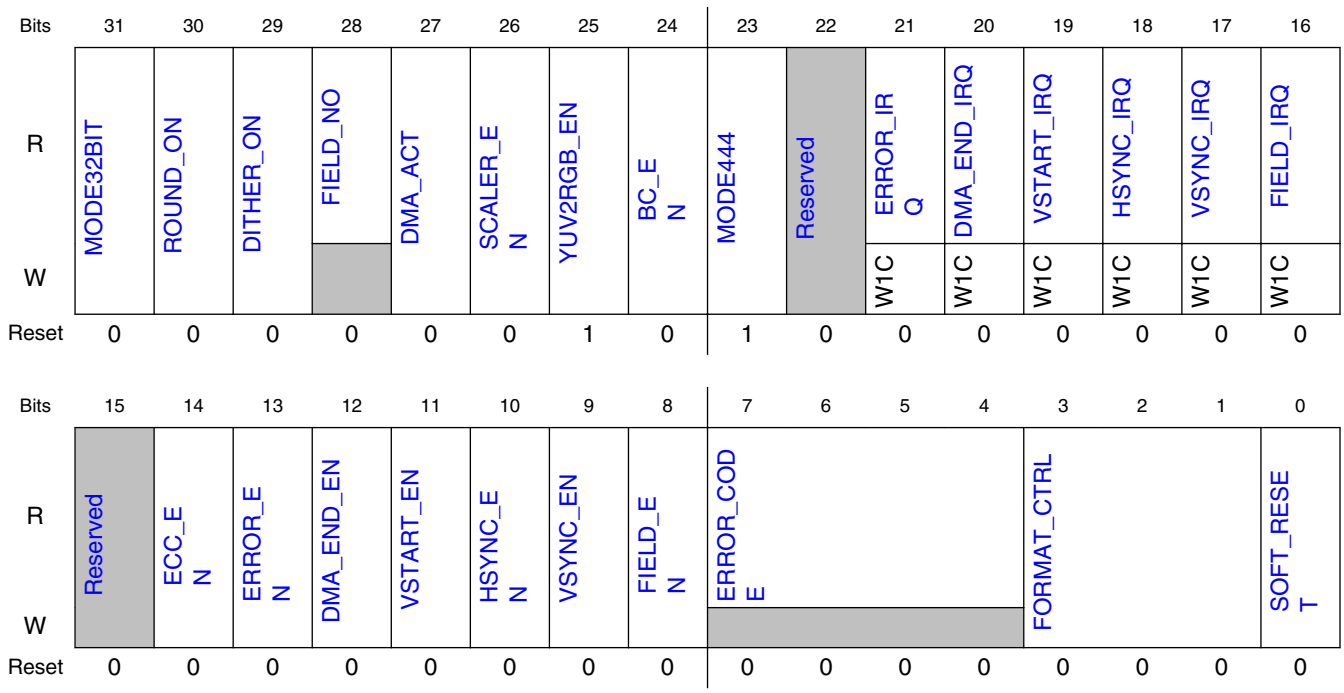
Offset	Register	Width (In bits)	Access	Reset value
0h	Status And Configuration Register (SCR)	32	RW	0280_0000h
8h	Detected Input Video Pixel and Line Counts (DINVSZ)	32	RW	0000_0000h
Ch	Detected Input Video Frame Length (DINVFL)	32	RW	0000_0000h
14h	Base Address Of Every Field/Frame Of Picture In Memory (DMA_ADDR)	32	RW	0000_0000h
18h	Horizontal DMA Increment (DMA_INC)	32	RW	0000_0000h
1Ch	Input Video Pixel and Line Count (INVSZ)	32	RW	00F0_02D0h
20h	High Priority Bus Request Alarm (HPRALRM)	32	RW	0000_0090h
24h	Programable Alpha Value (ALPHA)	32	RW	0000_00FFh
28h	Scaling Factor In Horizontal Direction (HFACTOR)	32	RW	0000_0100h
2Ch	Down Scaling Factor In Vertical Direction (VFACTOR)	32	RW	0000_0100h
30h	Scaling Destination Pixel and Line Count (VID_SIZE)	32	RW	00F0_02D0h
34h	B/C Adjust Look-up-table Current Address (LUT_ADDR)	32	RW	0000_0000h
38h	B/C Adjust Look-up-table Data Entry (LUT_DATA)	32	RW	See description.
3Ch	Extended Configuration Register (EXT_CONFIG)	32	RW	0000_0500h
4Ch	Active Image Origin (ACT_ORG)	32	RW	0000_0000h
50h	Active Image Size (ACT_SIZE)	32	RW	0000_0000h

49.5.1.2 Status And Configuration Register (SCR)

49.5.1.2.1 Offset

Register	Offset
SCR	0h

49.5.1.2.2 Diagram



49.5.1.2.3 Fields

Field	Function
31 MODE32BIT	MODE32BIT Select 32-bit or 16-bit output from the output formatter (block 6 in Figure 49-1) **NOTE:** For RGB666 parallel input, when MODE32BIT is cleared, the output is {R[5:1], G[5:0], B[5:1]}; when MODE32BIT is set, the output is {R[5:0], 2'b00, G[5:0], 2'b00, B[5:0], 2'b00} **NOTE:** This bit only acts when MODE_8BIT of EXT_CONFIG is 1'b0. See description of MODE_8BIT. 0b - 16-bit RGB or YUV 4:2:2 output 1b - 32-bit RGB or YUV 4:4:4 output. DITHER_ON and ROUND_ON are ignored if output is 32-bit RGB.
30 ROUND_ON	ROUND_ON Round is on. Used when video data is stored in buffer as RGB565 format.

Table continues on the next page...

Field	Function
29 DITHER_ON	DITHER_ON Dithering is on. Used when video data is stored in buffer as RGB565 format and ROUND_ON is not set.
28 FIELD_NO	FIELD_NO Field number, extracted from ITU-656 stream.
27 DMA_ACT	DMA_ACT DMA transfer of current field/frame is busy (write by software, cleared at end of transfer). When DMA_ACT is cleared, input video data is ignored and not put into FIFO.
26 SCALER_EN	SCALER_EN Scaling enable.
25 YUV2RGB_EN	YUV2RGB_EN YUV to RGB conversion enable.
24 BC_EN	BC_EN Bright/Contrast adjust enable.
23 MODE444	MODE444 YUV 4:4:4 mode enable bit. When it is set ITU decoder sends out YUV 4:4:4 format data, otherwise YUV 4:2:2 is sent by default. Scaler works on YUV 4:4:4 format. So this bit shall be set when scaling is enabled.
22 —	RESERVED
21 ERROR_IRQ	ERROR_IRQ Interrupt status bit. Write '1' to clear ERROR_IRQ.
20 DMA_END_IRQ	DMA_END_IRQ Interrupt status bit. Write '1' to clear DMA_END_IRQ.
19 VSTART_IRQ	VSTART_IRQ Interrupt status bit. Write '1' to clear VSTART_IRQ.
18 HSYNC_IRQ	HSYNC_IRQ Interrupt status bit. Write '1' to clear HSYNC_IRQ.
17 VSYNC_IRQ	VSYNC_IRQ Interrupt status bit. Write '1' to clear VSYNC_IRQ.
16 FIELD_IRQ	FIELD_IRQ Interrupt status bit. Write '1' to clear FIELD_IRQ.
15 —	RESERVED
14 ECC_EN	ECC_EN When this bit is set ECC errors generate ERROR_IRQ and the nature of ECC error gets reflected on the ERROR_CODE bit field.
13 ERROR_EN	ERROR_EN Interrupt enable bit for ERROR_IRQ.
12 DMA_END_EN	DMA_END_EN Interrupt enable bit for DMA_END_IRQ.
11	VSTART_EN

Table continues on the next page...

Field	Function
VSTART_EN	Interrupt enable bit for VSTART_IRQ.
10 HSYNC_EN	HSYNC_EN Interrupt enable bit for HSYNC_IRQ.
9 VSYNC_EN	VSYNC_EN Interrupt enable bit for VSYNC_IRQ.
8 FIELD_EN	FIELD_EN Interrupt enable bit for FIELD_IRQ.
7-4 ERROR_CODE	ERROR_CODE Error code. Signals errors that triggered error IRQ. Other values not given are reserved. 0000b - No error 0001b - DMA arm command given during vertical active, DMA_ACT does not accept the value on IPS bus. 0010b - DMA arm command given during vertical blanking when DMA_ACT is set. 0100b - Line too long 0101b - Too many lines in a field/frame 0110b - Line too short 0111b - Not enough lines in a field/frame 1000b - FIFO overflow 1001b - FIFO underflow 1010b - One bit ECC error 1011b - Two or more bits ECC error
3-1 FORMAT_CTRL	FORMAT_CTRL Output pixel data format control. See below or refer to Figure 49-1 for detailed definition. Here 32-bit means MODE32BIT is set and 16-bit means MODE32BIT is cleared. RGB mode means YUV2RGB_EN is set and YUV mode means YUV2RGB_EN is cleared. C means U or V. Dummy means "don't care" data. **NOTE:** This format control field only acts when MODE_8BIT of EXT_CONFIG is 1'b0. When MODE_8BIT is 1'b1, the output is 8-bit width. 16-bit RGB mode: 3'b000: {R[7:3], G[7:2], B[7:3]}; 3'b001: {G[4:2], B[7:3], R[7:3], G[7:5]}; 32-bit RGB mode: 3'b000: {alpha, R, G, B}; 3'b001: {alpha, B, G, R}; 3'b010: {R, G, B, alpha}; 3'b011: {B, G, R, alpha}; 16-bit YUV mode: 3'b000: {C,Y}; 3'b001: {Y,C}; 32-bit YUV mode: 3'b000: {dummy, Y, U, V}; 3'b001: {dummy, Y, V, U};

Table continues on the next page...

Field	Function
	3'b010: {dummy, U, V, Y}; 3'b011: {dummy, V, U, Y}; 3'b100: {Y, U, V, dummy}; 3'b101: {Y, V, U, dummy}; 3'b110: {U, V, Y, dummy}; 3'b111: {V, U, Y, dummy};
0 SOFT_RESET	SOFT_RESET Writing 1 to this bit generates an internal reset to all components except registers in the VIU block. This bit should be set by software when an error interrupt is detected, and it needs to be cleared by software to release the software reset.

49.5.1.3 Detected Input Video Pixel and Line Counts (DINVSZ)

49.5.1.3.1 Offset

Register	Offset
DINVSZ	8h

49.5.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DETECTED_LINEC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DETECTED_PIXELC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.5.1.3.3 Fields

Field	Function
31-16 DETECTED_LINEC	DETECTED_LINEC (DETECTED_LINEC[15:0]) Detected number of active lines in each input video field/frame.

Table continues on the next page...

Field	Function
15-0 DETECTED_PIXELC	DETECTED_PIXELC (DETECTED_PIXELC[15:0]) Detected number of active pixels in each input video line.

49.5.1.4 Detected Input Video Frame Length (DINVFL)

49.5.1.4.1 Offset

Register	Offset
DINVFL	Ch

49.5.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DETECTED_FRAME_HEIGHT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DETECTED_FRAME_WIDTH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.5.1.4.3 Fields

Field	Function
31-16 DETECTED_FRAME_HEIGHT	DETECTED_FRAME_HEIGHT (DETECTED_FRAME_HEIGHT[15:0]) Detected number of total lines of each input video field/frame.
15-0 DETECTED_FRAME_WIDTH	DETECTED_FRAME_WIDTH (DETECTED_FRAME_WIDTH[15:0]) Detected number of clock cycles of each input video line.

49.5.1.5 Base Address Of Every Field/Frame Of Picture In Memory (DMA_ADDR)

49.5.1.5.1 Offset

Register	Offset
DMA_ADDR	14h

49.5.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDR													Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.5.1.5.3 Fields

Field	Function
31-3 ADDR	ADDR (ADDR[31:3]) Base address of every field of picture in memory used by DMA. Rewrite only after receiving DMA done interrupt and before arming DMA. The lowest 3 bits (bits 0:2 as shown here) of ADDR cannot be set. It is always 3'b0.
2-0 —	RESERVED

49.5.1.6 Horizontal DMA Increment (DMA_INC)

49.5.1.6.1 Offset

Register	Offset
DMA_INC	18h

49.5.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INC													Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.5.1.6.3 Fields

Field	Function
31-16 —	RESERVED
15-3 INC	INC (INC[15:3]) Value of this field should be zero or memory size that one active line occupies in memory. It will be added to the memory mapped rounded address at the end of every line. Memory size of one active line depends on line pixel number. It is PIXEL_COUNT[15:2] + PIXEL_COUNT[1:0] when MODE32BIT=0; PIXEL_COUNT[15:1] + PIXEL_COUNT[0] when MODE32BIT=1; It shall only be configured when DMA is inactive, during vertical blanking.
2-0 —	RESERVED

49.5.1.7 Input Video Pixel and Line Count (INVSZ)

49.5.1.7.1 Offset

Register	Offset
INVSZ	1Ch

49.5.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LINEC															
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PIXELC															
Reset	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0	0

49.5.1.7.3 Fields

Field	Function
31-16 LINEC	LINEC (LINEC[15:0]) Expected number of active lines in each input video field/frame. It shall only be configured when DMA is non-active, during vertical blanking If more lines are found during data receive part, a "too many lines" error interrupt is generated when ERROR_IRQ is set. Redundant lines are discarded. If fewer lines are found during data receive part, a "not enough lines error interrupt is generated when ERROR_IRQ is set.
15-0 PIXELC	PIXELC (PIXELC[15:0]) Expected number of active pixels in each input video line, it shall be an integer multiple of 4. It shall only be configured when DMA is non-active, during vertical blanking. If more pixels are found during data receive part, a "line too long" error interrupt is generated when ERROR_IRQ isset. Redundant pixels are discarded. If less pixels are found during data receive part, a line too short error interrupt is generated when ERROR_IRQ is set.

49.5.1.8 High Priority Bus Request Alarm (HPRALRM)

49.5.1.8.1 Offset

Register	Offset
HPRALRM	20h

49.5.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ALARM															
Reset	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0

49.5.1.8.3 Fields

Field	Function
31-16 —	RESERVED
15-0 ALARM	ALARM (ALARM[15:0]) High priority alarm threshold. When FIFO_FILL (FIFO_FILL means the amount of data that is allowed to accumulate in the DMA FIFO due to the DMA being unable to get access to the bus, i.e. a high watermark) is higher than this value, high priority bus request will be asserted. The high watermark threshold is set in terms of 64-bit words.

49.5.1.9 Programable Alpha Value (ALPHA)

49.5.1.9.1 Offset

Register	Offset
ALPHA	24h

49.5.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								ALPHA							
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

49.5.1.9.3 Fields

Field	Function
31-8 —	RESERVED
7-0 ALPHA	ALPHA (ALPHA[7:0]) Alpha value used for picture blending. This register is configured during vertical blanking and used from the next video field.

49.5.1.10 Scaling Factor In Horizontal Direction (HFACTOR)

49.5.1.10.1 Offset

Register	Offset
HFACTOR	28h

49.5.1.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					FACTOR										
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

49.5.1.10.3 Fields

Field	Function
31-11 —	RESERVED
10-0 FACTOR	FACTOR Scaling factor at horizontal direction. FACTOR[10:8] is used as integer part of the factor. FACTOR[7:0] is used as fractional part of the factor.

49.5.1.11 Down Scaling Factor In Vertical Direction (VFACTOR)

49.5.1.11.1 Offset

Register	Offset
VFACTOR	2Ch

49.5.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					FACTOR										
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

49.5.1.11.3 Fields

Field	Function
31-11 —	RESERVED
10-0 FACTOR	FACTOR Down scaling factor at vertical direction. FACTOR[10:8] is used as integer part of the factor. FACTOR[7:0] is used as fractional part of the factor.

49.5.1.12 Scaling Destination Pixel and Line Count (VID_SIZE)

49.5.1.12.1 Offset

Register	Offset
VID_SIZE	30h

49.5.1.12.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LINEC															
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PIXELC															
Reset	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0	0

49.5.1.12.3 Fields

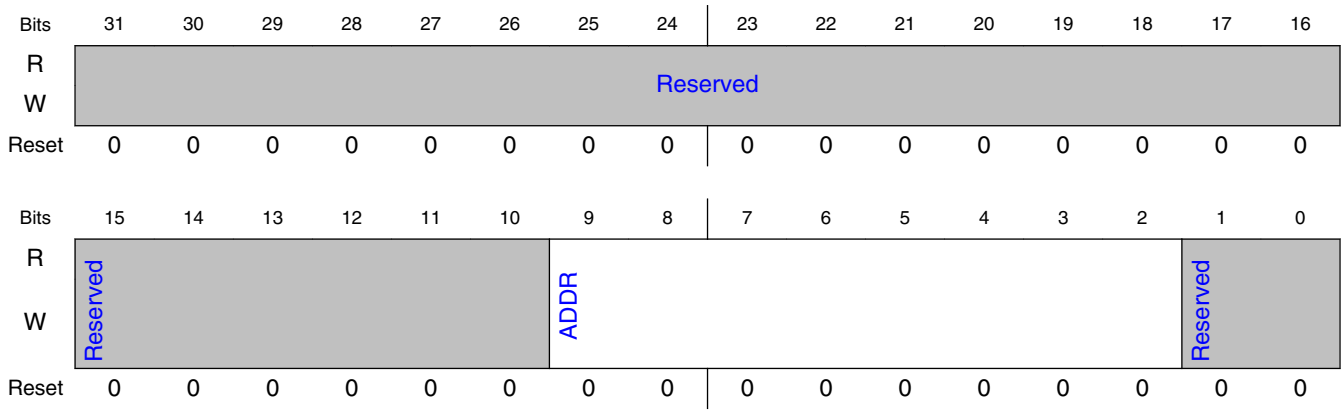
Field	Function
31-16 LINEC	LINEC (LINEC[15:0]) Expected number of lines in each output video frame after scaling.
15-0 PIXELC	PIXELC (PIXELC[15:0]) Expected number of pixels in each output video line after scaling. It shall be a multiple of 2 in 32-bit output mode, and a multiple of 4 in 16-bit output mode.

49.5.1.13 B/C Adjust Look-up-table Current Address (LUT_ADDR)

49.5.1.13.1 Offset

Register	Offset
LUT_ADDR	34h

49.5.1.13.2 Diagram



49.5.1.13.3 Fields

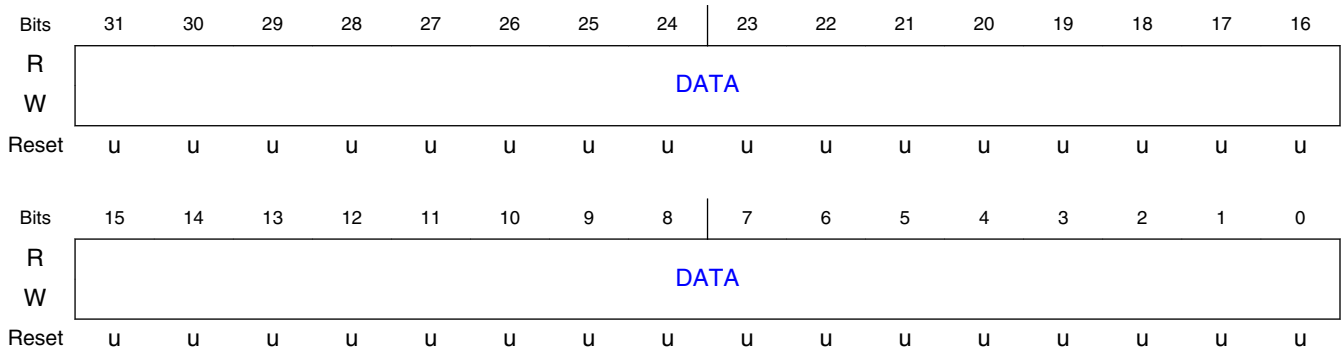
Field	Function
31-10 —	RESERVED
9-2 ADDR	ADDR (ADDR[9:2]) Current address pointer of the B/C adjust look-up-table. Value of this register increments (by 4) automatically at the end of each LUT_DATA write/read operation. This function allows fast update to the whole look-up-table, to the table of one color component, or even to any random field of the table. Note: ADDR reflects correct address only when clock of B/C adjust block is valid.
1-0 —	RESERVED

49.5.1.14 B/C Adjust Look-up-table Data Entry (LUT_DATA)

49.5.1.14.1 Offset

Register	Offset
LUT_DATA	38h

49.5.1.14.2 Diagram



49.5.1.14.3 Fields

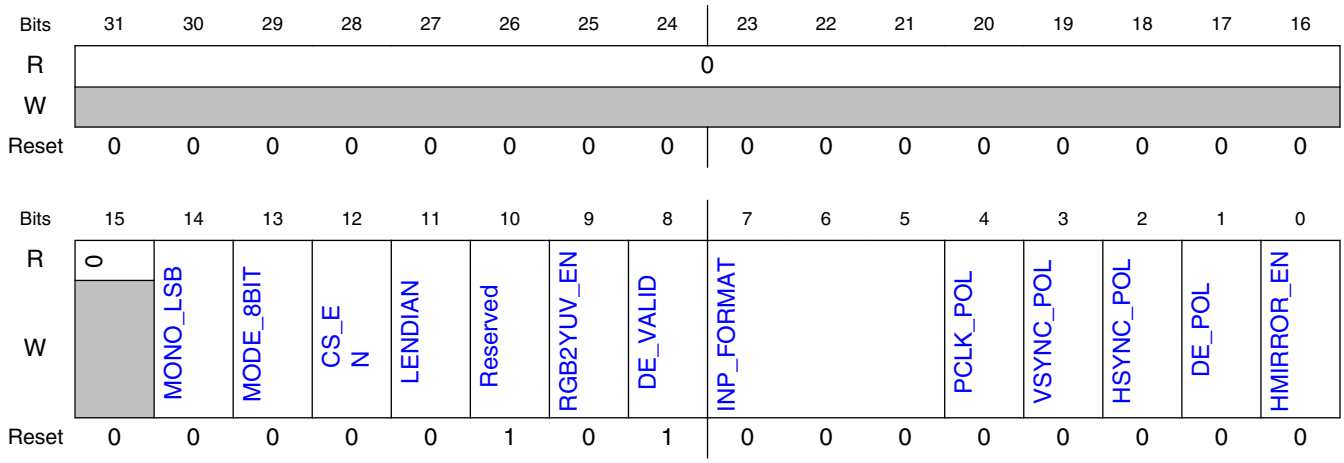
Field	Function
31-0 DATA	DATA (DATA[31:0]) B/C adjust look-up-table data entry. Data in this register is actually written/read to/from the address pointed to by the current LUT_ADDR value in the table. Note: DATA reflects correct data value only when clock of B/C adjust block is valid.

49.5.1.15 Extended Configuration Register (EXT_CONFIG)

49.5.1.15.1 Offset

Register	Offset
EXT_CONFIG	3Ch

49.5.1.15.2 Diagram



49.5.1.15.3 Fields

Field	Function
31-15 —	Reserved
14 MONO_LSB	Data location selector on input data bus. This bit decides the 8bit MONO input data is on 8 MSB or LSB. 0b - 8-bit MONO data is on LSB of input bus 1b - 8-bit MONO data is on MSB of input bus
13 MODE_8BIT	8 bit mode output format selector This mode select bit has higher priority than MODE32BIT of SCR register. When this bit is set, the output data will be 8 bit data per pixel. The user needs to clear this bit when using 16-bit or 32-bit mode. 0b - 8-bit output mode is not selected 1b - 8-bit output mode is selected
12 CS_EN	CS_EN Chroma swap enable bit. It's used to control how combining Y and C when output format is YUV422 mode. By default, for YUV422 mode(for example, assume 4 pixels per line here), the data in memory is Y0U0, Y1V0, Y2U2, Y3V2. When CS_EN is set, it becomes Y0V0, Y1U0, Y2V2, Y3U2. An application senario of CS_EN is that when mirror is on, the data will be Y3V2, Y2U2, Y1V0, Y0U0 by default. The first chrmo is V, instead of U. If CS_EN is set, the output data become Y3U2, Y2V2, Y1U0, Y0V0. 0b - Chroma swap is disabled. 1b - Chroma swap is enabled.
11 LENDIAN	LENDIAN Data endian control bit. This bit controls the endian of the IPM data bus. 0b - Big endian 1b - Little endian
10 —	Reserved Always write the reset value to this field.
9	RGB2YUV_EN

Table continues on the next page...

Field	Function
RGB2YUV_EN	RGB to YUV conversion enable
8 DE_VALID	DE_VALID External data enable indicator is valid at parallel input modes. 0b - DE is invalid 1b - DE is valid
7-5 INP_FORMAT	INP_FORMAT Input video format select bits. It should be set correctly according to the video input. 000b - 10/8bit ITU stream 001b - 24bit parallel YUV. Normally it is YUV444 010b - 8bit mono input. It is luminance Y. 011b - Reserved 100b - 24bit parallel RGB. It is RGB888 101b - 8bit serial RGB. It is RGB888 serial 110b - 18bit parallel RGB. It is RGB666 111b - 16bit parallel RGB. It is RGB565
4 PCLK_POL	PCLK_POL Pixel clock polarity control bit. pix_clk will be reversed when the bit is set. 0b - Active high 1b - Active low
3 VSYNC_POL	VSYNC_POL Vsync polarity control bit. Vsync will be reversed when the bit is set. 0b - Active high 1b - Active low
2 HSYNC_POL	HSYNC_POL Hsync polarity control bit. Hsync will be reversed when the bit is set. 0b - Active high 1b - Active low
1 DE_POL	DE_POL Data enable polarity control bit. DE will be reversed when the bit is set. 0b - Active high 1b - Active low
0 HMIRROR_EN	HMIRROR_EN Horizontal mirror enable.

49.5.1.16 Active Image Origin (ACT_ORG)

49.5.1.16.1 Offset

Register	Offset
ACT_ORG	4Ch

49.5.1.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ACT_ORG_Y															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ACT_ORG_X															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.5.1.16.3 Fields

Field	Function
31-16 ACT_ORG_Y	ACT_ORG_Y Y-coordinate of active image origin.
15-0 ACT_ORG_X	ACT_ORG_X X-coordinate of active image origin.

49.5.1.17 Active Image Size (ACT_SIZE)

49.5.1.17.1 Offset

Register	Offset
ACT_SIZE	50h

49.5.1.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ACT_IMG_HEIGHT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ACT_IMG_WIDTH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

49.5.1.17.3 Fields

Field	Function
31-16 ACT_IMG_HEIGHT	ACT_IMG_HEIGHT Height of image captured. Value '0' means no image clipping on Y direction.
15-0 ACT_IMG_WIDTH	ACT_IMG_WIDTH Width of image captured. Value '0' means no image clipping on X direction.

49.6 Functional Description

The VIU accepts parallel RGB, serial RGB, parallel YUV and ITU-R BT.656 compatible video streams on its parallel interface, decodes it and optionally performs processes like scaling, brightness and contrast adjust, RGB2YUV or YUV2RGB conversion, and de-interlace (weaving), then stores the result video stream to the system memory which can then be displayed by a display controller, or post processed.

Functions of the VIU are designed in a way that they can be flexibly enabled or disabled by software. But there are a few limitations as listed below:

- The scaler works on YUV 4:4:4 format or RGB888 formats, so to enable the scaler, decoder shall be configured in YUV 4:4:4 or RGB888 mode.
- To use the scaler, progressive video input shall be used for display quality's sake, and the de-interlace shall be disabled.

49.6.1 Input Formats

VIU accepts parallel RGB, serial RGB, parallel YUV and ITU-R BT.656 compatible video streams on its parallel interface. Below is the description for the timing of input video formats.

49.6.1.1 ITU656

The ITU-R BT.656-4 recommendation describes the means of interconnecting digital television equipment operating on the 525-line or 625-line standards and combines with the 4:2:2 encoding parameters as defined in the ITU-R BT.601 recommendation.

The data stream structure on ITU-R BT.656-4 interface is shown in the following figure. There are two timing reference signals, one at the beginning of each video data block (start of active video, SAV) and one at the end of each video data block (end of active video, EAV).

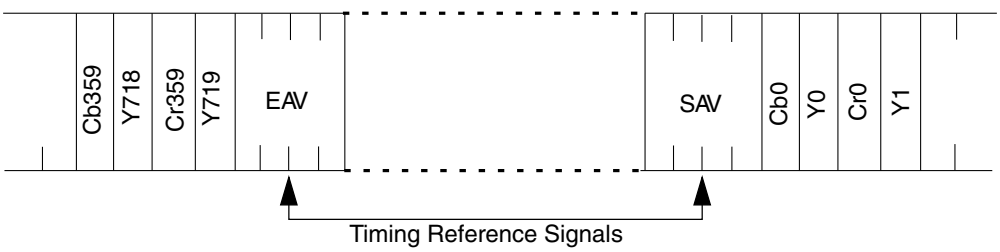


Figure 49-2. Interface Data Stream of ITU-R BT.656-4

Each timing reference signal consists of a four-word sequence in the following format: FF 00 00 XY. Values are expressed in hexadecimal notation. Value FF and 00 are reserved to be used in the timing reference signals.

The first three words are a fixed preamble. The fourth word contains information defining field 2 identification, the state of field blanking, and the state of line blanking. The assignment of bits within the timing reference signal is shown in the table below.

Table 49-4. Video Timing Reference Codes

Data Bit Number	First Word (FF)	Second Word (00)	Third Word (00)	Fourth Word (XY)
9(MSB)	1	0	0	1
8	1	0	0	F(0: field 1, 1: field 2)
7	1	0	0	V(0: elsewhere, 1: field blanking)
6	1	0	0	H(0: in SAV, 1: in EAV)
5	1	0	0	P3
4	1	0	0	P2

Table continues on the next page...

Table 49-4. Video Timing Reference Codes (continued)

Data Bit Number	First Word (FF)	Second Word (00)	Third Word (00)	Fourth Word (XY)
3	1	0	0	P1
2	1	0	0	P0
1	1	0	0	0
0	1	0	0	0

In above table, bits P0, P1, P2, P3 have states dependent on the states of the bits F, V and H. At the receiver side this arrangement permits one-bit errors to be corrected and two-bit errors to be detected.

Progressive video is also composed of fields; but instead of containing even lines in one field and odd lines in the other, the fields contain contiguous lines, i.e., field0 contains lines 1,2,3,... and field1 contains rest of the lines starting from (last line number of field0 + 1). And the F bit cannot be ignored in case of progressive frame.

Refer to the ITU-R BT.656-4 recommendation for more details.

49.6.1.2 Parallel Input Format

VIU also accepts parallel input video formats. It includes 24bit RGB888, YUV444, 18bit RGB666 and 16bit RGB565 formats. Their data streams are similar. The following caveats apply to the timing diagram below:

- FID must change state 3 Hsync pulses into Vsync.
- The first -//- break represents at least 10 lines.
- A field should contain more than 16 active lines.
- Hsync pulse should be longer than 60 pixel clock cycles.
- A active line should contain more than 24 pixels.

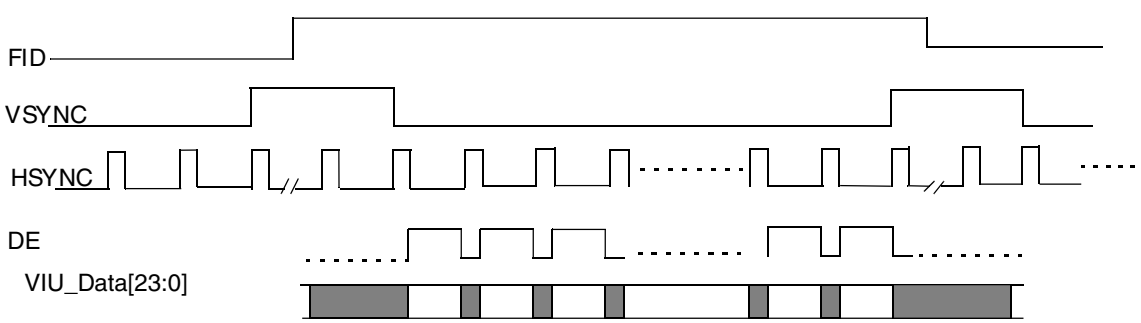


Figure 49-3. Parallel Input Timing

49.6.1.3 Parallel YC Format

Parallel YC video should be in YUV222 format. The timing diagram is provided below.

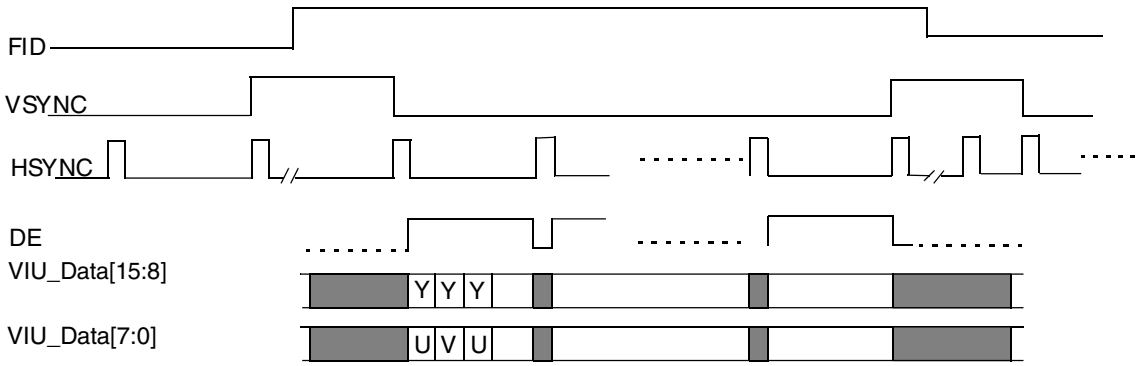


Figure 49-4. Parallel YC Timing

49.6.1.4 Serial RGB888 Format

The timing diagram of serial RGB888 is shown below.

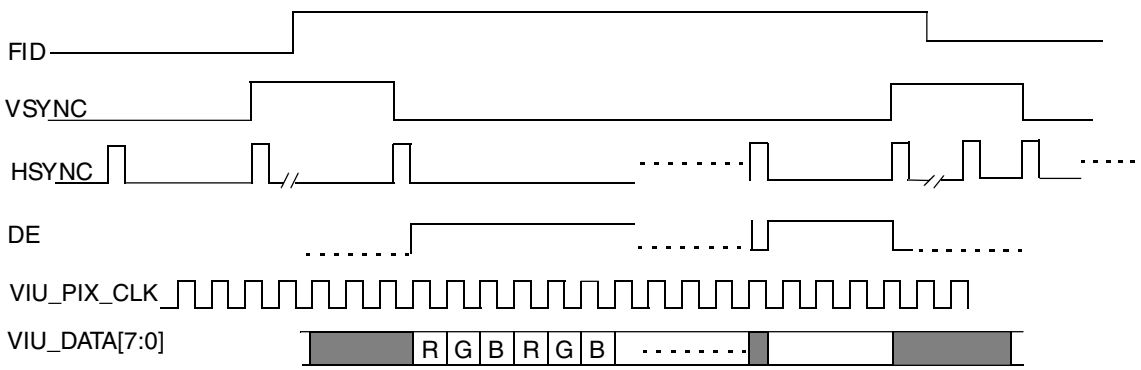


Figure 49-5. Serial RGB888 Timing

49.6.1.5 Mono Image Input Format

The timing diagram of mono input format is shown below.

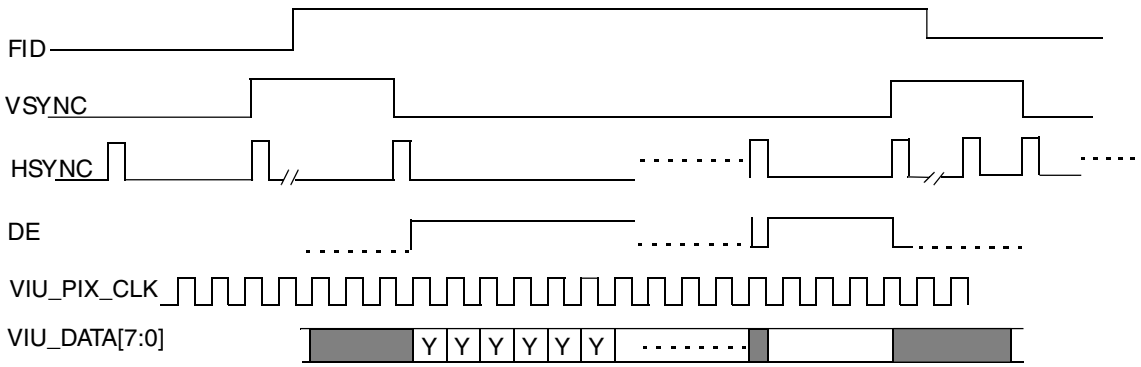


Figure 49-6. Mono Image Input Timing

49.6.2 Input Synchronizer

The Synchronizer block (1) captures ITU656 protocol signals from the interface, and synchronizes them to the IPG_CLK domain.

49.6.3 Decoder

The ITU Decoder block (2) detects the ITU656 timing reference signal (consists of a four-word sequence in the following format: FF-00-00-XY) and extracts HSYNC, VSYNC, field number signals and video data from the ITU data stream. The format of active pixel data from the ITU stream is YUV 4:2:2. The decoder block can directly send out this YUV 4:2:2 data, or interpolate it to YUV 4:4:4 format and send it out. This is determined by the MODE444 field in the SCR register. This bit shall be set if the scaler is enabled, because the scaler works on YUV 4:4:4 data format.

49.6.4 Scaling

VIU supports up to 1/8 horizontal and vertical down-scaling, as well as 2x horizontal up-scaling.

49.6.4.1 Down Scaling

The Down-scaler block (3) performs down-scaling to the incoming video stream. It is able to scale down the input video stream by a fractional scaling ratio up to 1/8. The down-scaler block can be enabled/disabled by software via the SCALER_EN bit in the SCR register. To use the down-scaler video data extracted from ITU decoder shall be in YUV 4:4:4 format, or the input format is non-ITU (INP_FORMAT is not 3'b000).

The down-scaler uses a bi-linear filter with simple linear interpolation between the two nearest neighbors, on both horizontal and vertical direction. Scaling is firstly done on the horizontal direction (called XScale), and then the vertical direction (called YScale). Different scaling factors are supported on horizontal direction and vertical direction.

Assuming the scaling factors are (factor_h, factor_v), in theory for every pixel (x, y) of the scaled picture the coordinates of the corresponding pixel in the incoming picture (x1, y1) can be calculated by multiplying the coordinates with the scaling factors, say (x1 = x * factor_h, y1 = y * factor_v). Now because the scaling factors can be fractions, the coordinates (x1, y1) are not integer anymore, they are fractional values. The scalers use the integer part of (x1, y1) to find the two neighboring pixels from the incoming picture as input to the filters, and the fractional part to derive the weighting factors for interpolation.

The scaling factors, (factor_h, factor_v), are both 11 bit with the lower 8 bit as the factional part and the highest 3 bit the integer part. It is capable of scaling the input picture by up to 8, in steps of 1/256. It is by 8 if the factor is programmed as all zeros.

Instead of using multipliers to calculate (x1, y1), two 20-bit phase accumulator is used to step through the source pixels/or lines, where the lower 8 bit is the fractional part. For every output pixel/or line the phase adder is added to the accumulator. With the 12 bit integer part of the accumulator, up to 4096 source pixels/or lines can be supported.

So for each target pixel, the current accumulator position is used to determine how the pixel is going to be produced. As an example, accumulator value 0x123.3a means the step position is between pixel[0x123] and pixel[0x124], and the output pixel will be calculated by (pixel[0x124] * (0x100 - 0x3a) + pixel[0x123] * 0x3a) >> 8. The same concept is used for horizontal and vertical scaling.

Because of the vertical scaling, a line buffer is used for each color component to store the data from horizontal scaling, it stores one line of data. Size of the line buffer determines the maximum video output size after scaling.

The scaling factors are programmed via the HFACTOR and VFACTOR registers. Video size after scaling is programmed via the VID_SIZE register.

Three scalers are instantiated in the down-scaler block, one for each component.

49.6.4.2 Up-scaling

The theory of up-scaling is the same with down-scaling. When HFACTOR is smaller than 1, up-scaling is implemented by VIU scaler. But note that VIU does not support vertical up-scaling, and HFACTOR should not be less than 0.5.

Note

The scaled pixel count per line shall be an integer multiple of 2 in 32-bit output mode, or 4 in 16-bit output mode. The user shall divide the input pixel count by the horizontal scaling factor and truncate the result to a multiple of 2 or 4.

Note

To achieve good display quality, progressive video input shall be used if down-scaling is needed.

49.6.5 Brightness and Contrast Adjust

The B/C Adjust block (4) performs brightness and contrast adjustment on the input video stream via three internal look-up-tables, one table per color component. Each table contains 256 8-bit entries, it maps every incoming pixel to the value of one of its entries according to the original value of the pixel. This feature allows the user to adjust brightness and/or contrast of the incoming picture according to three arbitrary adjustment curves, one adjustment curve per color component.

To use this feature, the B/C adjust look-up-table shall be programmed in software in the following format.

Table 49-5. B/C Adjust Look-up-table Format

Color Component	BC LUT Offset	Local Address [1:0]			
		00	01	10	11
Y	0x000	$BC0_Y$	$BC1_Y$	$BC2_Y$	$BC3_Y$
	.....				
	0x0FC	$BC252_Y$	$BC253_Y$	$BC254_Y$	$BC255_Y$
U	0x100	$BC0_U$	$BC1_U$	$BC2_U$	$BC3_U$
	.....				
	0x1FC	$BC252_U$	$BC253_U$	$BC254_U$	$BC255_U$
V	0x200	$BC0_V$	$BC1_V$	$BC2_V$	$BC3_V$
	.....				
	0x2FC	$BC252_V$	$BC253_V$	$BC254_V$	$BC255_V$

Two registers are provided to program the look-up-table, they are LUT_ADDR and LUT_DATA. LUT_ADDR is the current address pointer, which always points to the current table offset to be programmed. It can be set by software, and it increases (by 4) automatically when each word is written to the table, and it falls back to 0x000 if the current value is 0x2FC and the last word is written to the table. LUT_DATA is the table

data entry, data written to this register will be stored into the table, to the address pointed to by LUT_ADDR. This register shall only be written by 4-byte word. With the combination of these two registers the user can program the B/C look-up-table conveniently, either program the whole table, or reprogram the table for one color component only, or even change one single arbitrary word in the table.

The B/C adjust can be enabled/disabled by software via the BC_EN bit in the SCR register.

49.6.6 YUV to RGB Conversion

The YUV2RGB block (5) is used to convert YUV (4:2:2 or 4:4:4) to RGB (888 or 565). When the input is YUV 4:2:2, it is interpolated to YUV 4:4:4 first.

49.6.7 Round and Dither

In RGB565 output mode, when pixel data is converted from RGB888 to RGB565 the image is anamorphic more or less, because of losing color information conveyed by the least significant two or three bits of the original color components, which are dropped. VIU block provides two simple algorithms, round and dither, to compensate this color information loss.

49.6.7.1 Round

In round mode, VIU will round to 1 in LSB if the decimal fraction is bigger than 0.5 and ignore the smaller fraction when ROUND_ON is set in the SCR register.

49.6.7.2 Dither

Dither is a little more complex but better than round for recovering image. It is a statistical compensation algorithm. It does not render all pixels with the same grey or color level, but some with the lower one, and some with a color level of 1 LSB more. The selection of adding one LSB or not depends on the position of the pixel on the screen.

The figure below shows the implementation of dither in the VIU block.

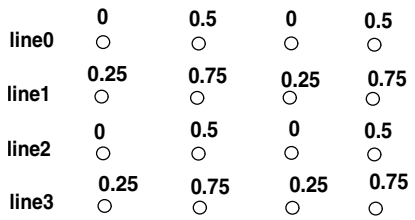


Figure 49-7. Dither Implementation

The number above the pixel position in the diagram is the compensation value for this pixel. When pixels have a value of 0.25, they are rendered 0 in 75% of the pixels and 1 in 25% of pixels. This averages to 0.25. Similarly, pixel values of 0.5, 0.75, and 1.0 are rendered 50%, 75% and 100% of the pixels as 1. For human eyes, this rendering result of dither makes the holistic image smoother and closer to the original one.

49.6.8 Output Formatter

The Output Formatter block (6) accepts YUV or RGB data from the YUV2RGB block, arranges them in expected format, and then sends it to the DMA engine. The FORMAT_CTRL field in SCR register controls how the VIU stores data to system memory; see the following figure for details.>

Table 49-6. VIU Output Data Stream Format

Pixel Format	FORMAT _CTRL[1]	Local Address [2:0]							
		000	001	010	011	100	101	110	111
RGB 565	000	R0[7:3], G0[7:2], B0[7:3]		R1[7:3], G1[7:2], B1[7:3]		R2[7:3], G2[7:2], B2[7:3]		R3[7:3], G3[7:2], B3[7:3]	
	001[2]	G0[4:2], B0[7:3], R0[7:3], G0[7:5]		G1[4:2], B1[7:3], R1[7:3], G1[7:5]		G2[4:2], B2[7:3], R2[7:3], G2[7:5]		G3[4:2], B3[7:3], R3[7:3], G3[7:5]	
RGB 8888	000	A0	R0	G0	B0	A1	R1	G1	B1
	001	A0	B0	G0	R0	A1	B1	G1	R1
	010	R0	G0	B0	A0	R1	G1	B1	A1
	011	B0	G0	R0	A0	B1	G1	R1	A1
YUV 4:2:2	000	U0	Y0	V0	Y1	U2	Y2	V2	Y3
	001	Y0	U0	Y1	V0	Y2	U2	Y3	V2
YUV 4:4:4	000	dummy	Y0	U0	V0	dummy	Y1	U1	V1
	001	dummy	Y0	V0	U0	dummy	Y1	V1	U1
	010	dummy	U0	V0	Y0	dummy	U1	V1	Y1
	011	dummy	V0	U0	Y0	dummy	V1	U1	Y1
	100	Y0	U0	V0	dummy	Y1	U1	V1	dummy
	101	Y0	V0	U0	dummy	Y1	V1	U1	dummy
	110	U0	V0	Y0	dummy	U1	V1	Y1	dummy
	111	V0	U0	Y0	dummy	V1	U1	Y1	dummy
8-bit Mode	XXX	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7

1. All values not shown in the table shall be considered as reserved and shall not be used.
2. This format is basically intended for data communication with any little-endian peripheral in the system, assuming big-endian system memory is used.

NOTE

RGB666 input can be stored in memory in RGB565 or RGB8888 formats. When stored as RGB565, the R[0] and B[0] component of RGB666 input will be discarded. When stored as RGB8888, the RGB666 input data will be reflected in memory as {R[5:0], 2'b00, G[5:0], 2'b00, B[5:0], 2'b00}. The position of Alpha bit will depend on the FORMAT_CTRL setting.

49.6.9 High Priority Alarm

FIFO_FILL is a status field which indicates the amount of data in the FIFO. The HI_PRIO_ALARM register is a threshold such that, when the data number in the FIFO is larger than HI_PRIO_ALARM, the DMA request will be high-priority on the AMBA crossbar side. The purpose of this register is to enhance DMA performance. Normally the reset value should be sufficient; however, if the FIFO is seen to overflow, the user should lower the threshold in order to allow the DMA to service the FIFO more frequently.

49.6.10 DMA and De-interlace

The DMA engine block (7) functions as the FIFO controller and DMA engine. It stores the data from the output formatter block into a FIFO and then writes them to system memory via the IPM interface.

VIU block has an embedded DMA. When video data is converted to RGB format and placed into a 256 × 64 bit FIFO, it waits to be transferred to memory by the internal DMA.

After doing some necessary register configuration, such as coefficients, INVSZ and DMA_ADDR, user can activate DMA by setting the DMA_ACT of the status and configuration register. But note that DMA_ACT can be configured only during vertical blanking since the VIU block will not transfer a fragment of video field to memory. If it is configured during field active time, DMA transfer cannot be started and an error interrupt will be asserted.

The VIU block asserts a transfer request when there is enough data in the FIFO for one transfer. Normally one transfer conveys 32 bytes from FIFO to memory. At the end of a line, all remaining data is transferred, though it may not be as much as 32 bytes.

VIU also provides a simple way to de-interlace for interlaced or pseudo-interlaced video images. It is implemented by setting the DMA_ADDR and DMA_INC registers.

The figure below shows the implementation of de-interlace. The value of DMA_INC is added to the rounded address at the end of every active line. So, when DMA_INC is zero, pixel data is stored in memory line by line, meaning de-interlace is off, as shown in figure (1) and (2); otherwise, when DMA_INC equals to one-line memory mapped pixel size and DMA_ADDR(2) = DMA_ADDR(1) + DMA_INC, the odd field and even field will be merged into one frame in memory, as shown in figure (3).

Here DMA_ADDR(1) means base address of field 1, and DMA_ADDR(2) means base address of field 2. Memory mapped line size means memory size that is occupied by one active line pixels. It depends on pixel number of one line and video data format (RGB888 or RGB565). See the register description section for more details.

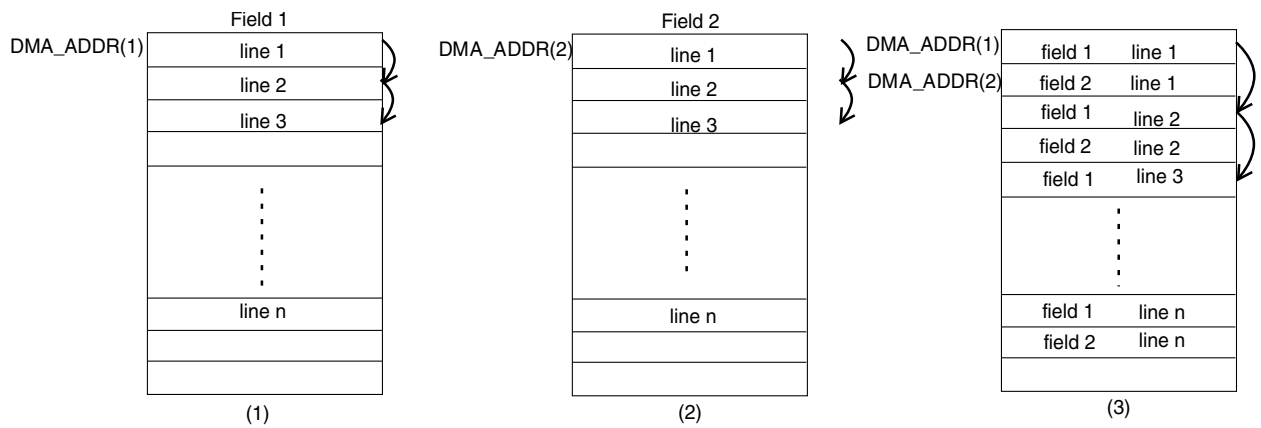


Figure 49-8. Implementation of De-interlace

When the HMIRROR_EN bit is set in the EXT_CONFIG register, the image can be mirrored in the horizontal direction, as illustrated below.

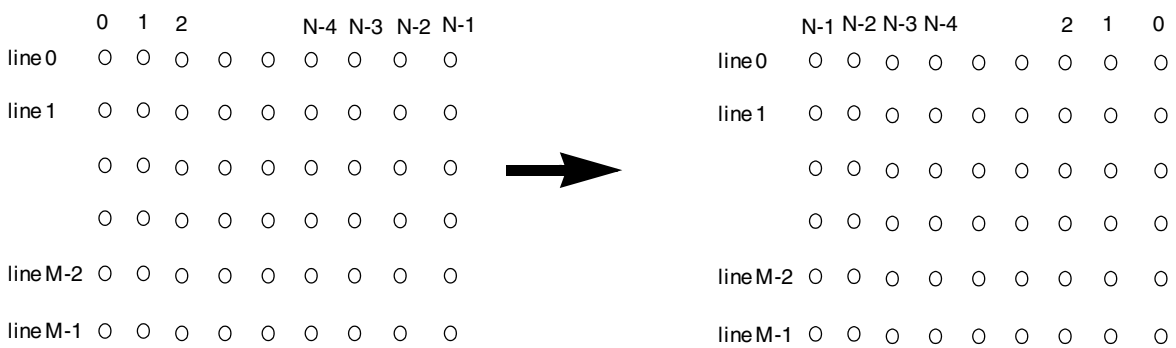


Figure 49-9. Horizontal Image Mirror

49.6.11 Error Case

Normally, the user should provide a standard and totally ITU-compatible video stream to the VIU block. However, it is difficult to avoid unexpected errors all the time. VIU can manage error cases like ECC error, line too long, line too short, too many lines, not enough lines in a field, and FIFO overflow.

- ECC error: ITU stream provides 4-bit error correcting code P[3:0] in its SAV and EAV. It is decoded in VIU to use the correct field number, horizontal sync and vertical sync bits. It can correct one bit errors and find two bit errors. When an ECC error is found, an interrupt is asserted.
- Line too long error: When pixels of active line is longer than PIXELC, a line too long error interrupt is asserted and redundant pixels are discarded.
- Too many lines error: When active lines of a field are bigger than LINEC, a too many lines error interrupt is asserted and redundant lines are discarded.
- Line too short error: When pixels of active line is less than PIXELC, a line too short error interrupt is asserted.
- Not enough line error: When active lines of a field is less than LINEC, a not enough line error interrupt is asserted.
- FIFO overflow error: If the system bus is blocked for a long time, video data is stored in FIFO and causes FIFO overflow. When FIFO overflow occurs, an interrupt is asserted and incoming data is discarded until FIFO works normally again. Current field is jumbled. However, VIU recovers to work at the next field if the bus is unblocked at that time.
- FIFO underflow: When the FIFO is read when it is empty, a FIFO underflow error interrupt is asserted. Normally, this error should not occur.

VIU can manage the above error cases to a certain extent. However, when it does not recover to a working state the user should write the SOFT_RESET bit of the status and configuration register to reset the VIU block.

49.7 Initialization/Application Information

Initialization steps and startup information are given below.

49.7.1 Initialization Information

When the VIU block comes out of reset, software should implement the following steps to start this block.

1. Program the SCR register to set the VIU to the desired operation mode
   - To enable YUV 4:2:2 to 4:4:4 interpolation in the ITU decoder, set the MODE444 bit[2]
   - To enable the scaler, set the SCALER_EN bit
   - To enable B/C adjust, set the BC_EN bit
   - To enable RGB565 output mode, set the YUV2RGB_EN bit and clear the MODE32BIT bit. Optionally set the DITHER_ON bit or the ROUND_ON bit to enable dither or round[3]
   - To enable RGB8888 output mode, set the YUV2RGB_EN and MODE32BIT bits
   - To enable YUV output mode, clear the YUV2RGB_EN bit
2. Set the FORMAT_CTRL field so that the VIU outputs data in the correct format. Configure the input video size via the INVSZ register.
3. If it is desired to use the scaling function, program the scaling factors and destination video size after scaling.
4. If it is desired to use the B/C adjust function, program the B/C adjust look-up-table via the LUT_DATA and LUT_ADDR registers.
5. Configure the HPRALRM and ALPHA registers if necessary.
6. Set the VSYNC_EN and/or FIELD_EN bits in the SCR register to enable vsync or field interrupt. Meanwhile, disable error interrupt.
7. When software receives a vsync interrupt and/or field interrupt, read FIELD_NO bit of the SCR register[4].
8. According to the FIELD_NO bit, program the DMA_ADDR register. This is the field start address in system memory, or frame start address in progressive video input mode.

2. MODE444 bit shall be set when the scaler is enabled.
3. If DITHER_ON or ROUND_ON are both set, only round will be enabled.
4. Reading the FIELD_NO bit is optional, especially in progressive video input mode.

9. If it is desired to use the de-interlace (weaving) function, program the line start address offset value in the DMA_INC register[5].
10. Clear error status first if necessary.
11. Write the DMA_ACT bit of the SCR register to start FIFO and DMA transfer. This operation actually starts the VIU to operate.

49.7.2 Application Information

Normally, the user shall not change the register values of the function enable bits, input/output video size, DMA_INC, MODE32BIT, scaling factors and B/C adjust look-up-table contents after VIU is started up. When the video input source is changed, the user should reset the VIU and re-configure the related registers.

49.7.2.1 Register Configuration Timing Window

As mentioned above, dynamic configuration of registers is not recommended because it may cause error if it is not configured in a certain timing window, especially for INVSZ and DMA_INC.

Register configuration timing window is shown in the below diagram. All registers, except for the SOFT_RESET bit, are recommended to be configured during vertical blanking, after DMA transfer is done and the field identification bit is changed (field interrupt is asserted).

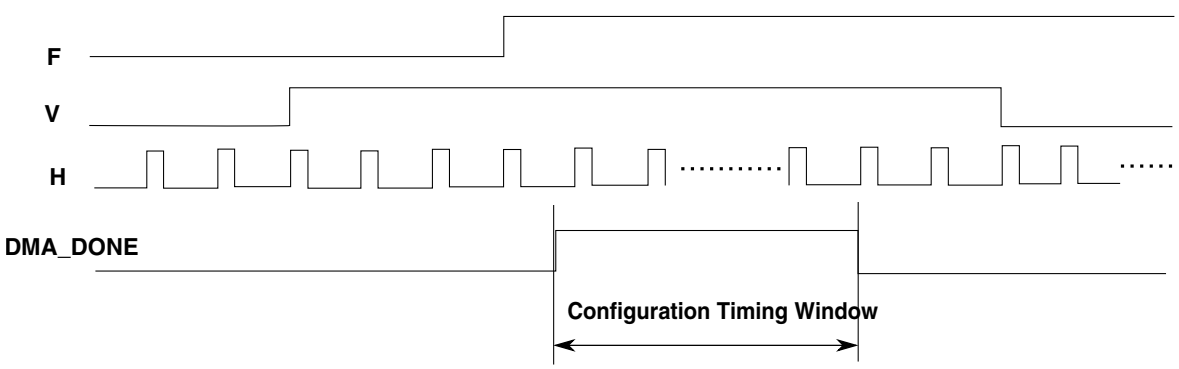


Figure 49-10. Register Configuration Timing Window

5. Progressive video input shall be used when scaling is enabled for better display quality.

Chapter 50
Flexible Input/Output (FlexIO)

50.1 Chip-specific FlexIO information

Table 50-1. Reference links to related information

Topic	Related module	Reference
Full description	FlexIO	FlexIO
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

50.1.1 FlexIO

The FlexIO module is capable of supporting a wide range of protocols including, but not limited to: UART, I2C, SPI, I2S, camera interface, display interface, PWM waveform generation, etc. The module can remain functional when the chip is in a low power mode provided the clock it is using remains active. It uses 32-bit IP bus.

Table 50-2. FlexIO configuration

Parameter	Description
Name	FlexIO
Instances	2
Configurable features	FLEXIO0-1: 8 shift registers, 8 timers, 32 pins per instance
Interface speed	FLEXIO0-1: set by I/O max
External I/O pins	FLEXIO0-1: D[0-31]. See attached IOMUXC spreadsheet for pinout details

50.1.2 Trigger connections

On i.MX 7ULP, TPM trigger sources get routed through the TRIGMUX module. See the Table 38-2 and Table 38-4 for external/internal trigger connections to this module.

50.2 Introduction

50.2.1 Overview

The FlexIO is a highly configurable module providing a wide range of functionality including:

- Emulation of a variety of serial/parallel communication protocols
- Flexible 16-bit timers with support for a variety of trigger, reset, enable and disable conditions
- Programmable logic blocks allowing the implementation of digital logic functions on-chip and configurable interaction of internal and external modules
- Programmable state machine for offloading basic system control functions from CPU

50.2.2 Features

The FlexIO module is capable of supporting a wide range of protocols including, but not limited to:

- UART
- I2C
- SPI
- I2S
- Camera IF
- Motorola 68K/Intel 8080 bus
- PWM/Waveform generation

The following key features are provided:

- Array of 32-bit shift registers with transmit, receive and data match modes
- Double buffered shifter operation for continuous data transfer
- Shifter concatenation to support large transfer sizes
- Automatic start/stop bit generation
- 1, 2, 4, 8, 16 or 32 multi-bit shift widths for parallel interface support
- Interrupt, DMA or polled transmit/receive operation

- Programmable baud rates independent of bus clock frequency, with support for asynchronous operation during stop modes
- Highly flexible 16-bit timers with support for a variety of internal or external trigger, reset, enable and disable conditions
- Programmable logic mode for integrating external digital logic functions on-chip or combining pin/shifter/timer functions to generate complex outputs
- Programmable state machine for offloading basic system control functions from CPU with support for up to 8 states, 8 outputs and 3 selectable inputs per state

50.2.3 Block Diagram

The following diagram gives a high-level overview of the configuration of FlexIO timers and shifters.

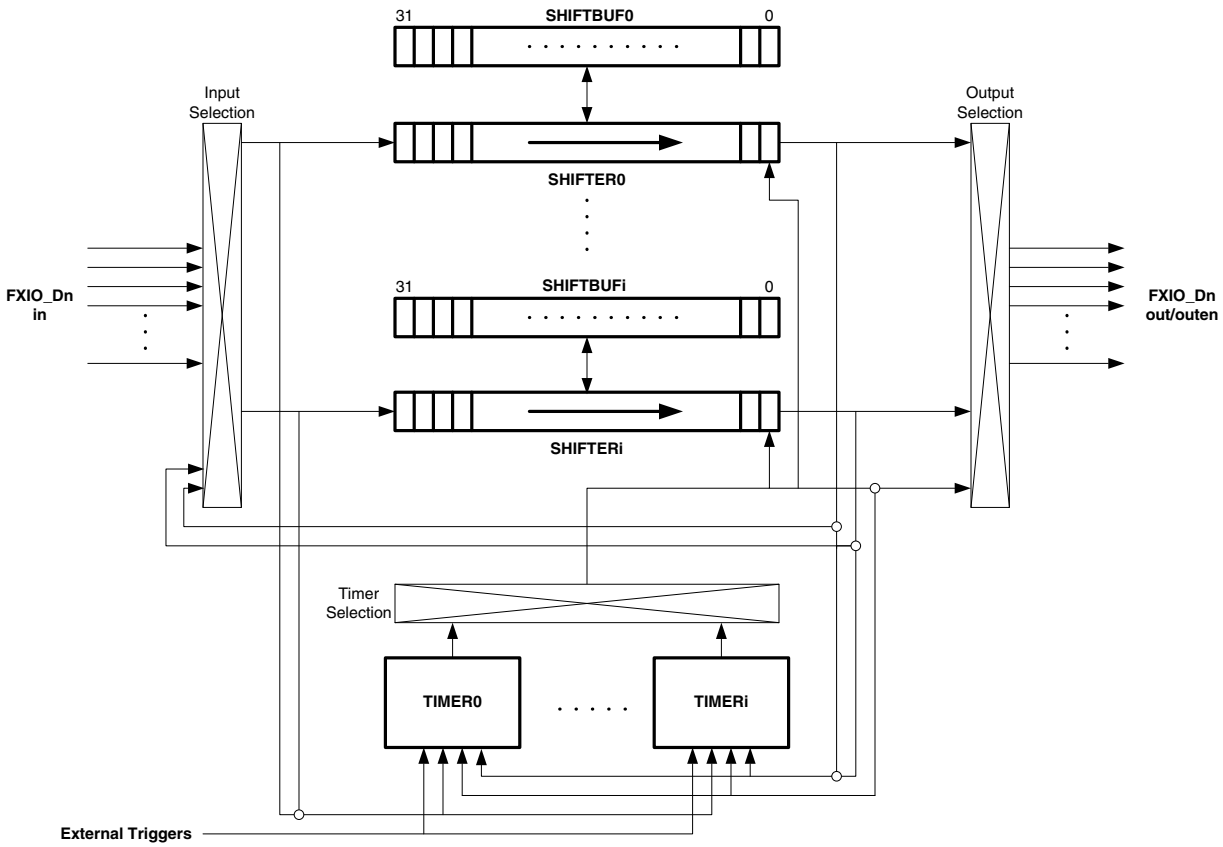


Figure 50-1. FlexIO block diagram

50.2.4 Modes of operation

The FlexIO module supports the chip modes described in the following table.

Table 50-3. Chip modes supported by the FlexIO module

Chip mode	FlexIO Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (CTRL[DOZEN]) is clear and the FlexIO is using an external or internal clock source which remains operating during stop/wait modes.
LLS	The Doze Enable (CTRL[DOZEN]) bit is ignored and the FlexIO will wait for all Timers to complete any pending operation before acknowledging LLS mode entry.
Debug	Can continue operating provided the Debug Enable bit (CTRL[DBGE]) is set.

50.2.5 FlexIO Signal Descriptions

Signal	Description	I/O
FXIO_Dn (n=0...31)	Bidirectional FlexIO Shifter and Timer pin inputs/outputs	I/O

50.3 Memory Map and Registers

50.3.1 FLEXIO register descriptions

NOTE

An invalid register access will result in a bus error. This includes reading a write-only register, writing a read-only register or accessing an invalid address.

50.3.1.1 FLEXIO Memory map

FLEXIO0 base address: 4103_2000h

FLEXIO1 base address: 4031_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0101_0001h
4h	Parameter Register (PARAM)	32	RO	0420_0808h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8h	FlexIO Control Register (CTRL)	32	RW	0000_0000h
Ch	Pin State Register (PIN)	32	RO	0000_0000h
10h	Shifter Status Register (SHIFTSTAT)	32	W1C	0000_0000h
14h	Shifter Error Register (SHIFTERR)	32	W1C	0000_0000h
18h	Timer Status Register (TIMSTAT)	32	W1C	0000_0000h
20h	Shifter Status Interrupt Enable (SHIFTSIEN)	32	RW	0000_0000h
24h	Shifter Error Interrupt Enable (SHIFTEIEN)	32	RW	0000_0000h
28h	Timer Interrupt Enable Register (TIMIEN)	32	RW	0000_0000h
30h	Shifter Status DMA Enable (SHIFTSDEN)	32	RW	0000_0000h
40h	Shifter State Register (SHIFTSTATE)	32	RW	0000_0000h
80h - 9Ch	Shifter Control N Register (SHIFTCTL0 - SHIFTCTL7)	32	RW	0000_0000h
100h - 11Ch	Shifter Configuration N Register (SHIFTCFG0 - SHIFTCFG7)	32	RW	0000_0000h
200h - 21Ch	Shifter Buffer N Register (SHIFTBUF0 - SHIFTBUF7)	32	RW	0000_0000h
280h - 29Ch	Shifter Buffer N Bit Swapped Register (SHIFTBUFBIS0 - SHIFTBUFBIS7)	32	RW	0000_0000h
300h - 31Ch	Shifter Buffer N Byte Swapped Register (SHIFTBUFBYS0 - SHIFTBUFBYS7)	32	RW	0000_0000h
380h - 39Ch	Shifter Buffer N Bit Byte Swapped Register (SHIFTBUFBBS0 - SHIFTBUFBBS7)	32	RW	0000_0000h
400h - 41Ch	Timer Control N Register (TIMCTL0 - TIMCTL7)	32	RW	0000_0000h
480h - 49Ch	Timer Configuration N Register (TIMCFG0 - TIMCFG7)	32	RW	0000_0000h
500h - 51Ch	Timer Compare N Register (TIMCMP0 - TIMCMP7)	32	RW	0000_0000h
680h - 69Ch	Shifter Buffer N Nibble Byte Swapped Register (SHIFTBUFNBS0 - SHIFTBUFNBS7)	32	RW	0000_0000h
700h - 71Ch	Shifter Buffer N Half Word Swapped Register (SHIFTBUFHWS0 - SHIFTBUFHWS7)	32	RW	0000_0000h
780h - 79Ch	Shifter Buffer N Nibble Swapped Register (SHIFTBUFNIS0 - SHIFTBUFNIS7)	32	RW	0000_0000h

50.3.1.2 Version ID Register (VERID)

50.3.1.2.1 Offset

Register	Offset
VERID	0h

50.3.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

50.3.1.2.3 Fields

Field	Function
31-24	Major Version Number
MAJOR	This read only field returns the major version number for the module specification.
23-16	Minor Version Number
MINOR	This read only field returns the minor version number for the module specification.
15-0	Feature Specification Number
FEATURE	This read only field returns the feature set number. 0000000000000000b - Standard features implemented. 0000000000000001b - Supports state, logic and parallel modes.

50.3.1.3 Parameter Register (PARAM)

50.3.1.3.1 Offset

Register	Offset
PARAM	4h

50.3.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRIGGER								PIN							
W																
Reset	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER								SHIFTER							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

50.3.1.3.3 Fields

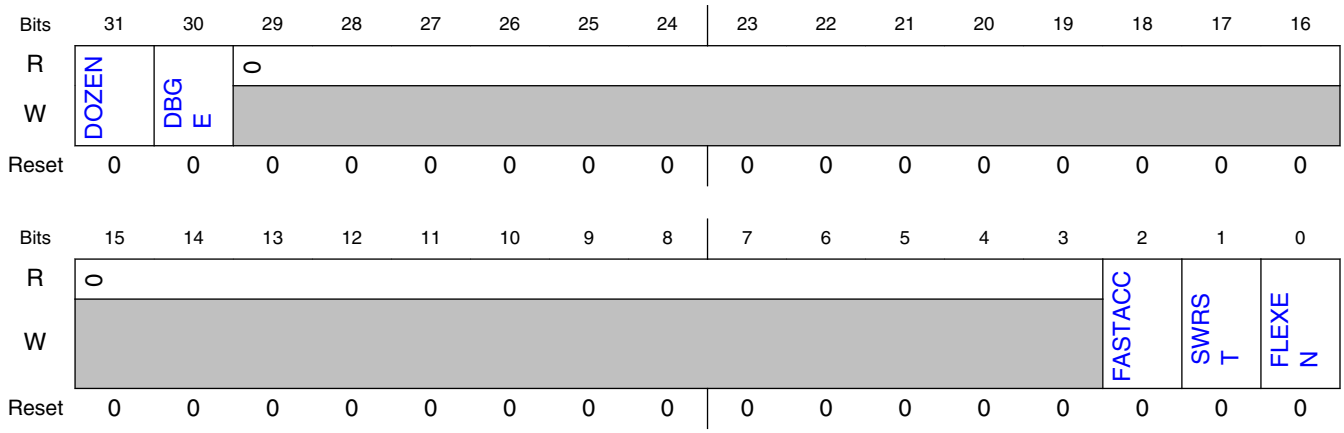
Field	Function
31-24 TRIGGER	Trigger Number Number of external triggers implemented.
23-16 PIN	Pin Number Number of Pins implemented.
15-8 TIMER	Timer Number Number of Timers implemented.
7-0 SHIFTER	Shifter Number Number of Shifters implemented.

50.3.1.4 FlexIO Control Register (CTRL)

50.3.1.4.1 Offset

Register	Offset
CTRL	8h

50.3.1.4.2 Diagram



50.3.1.4.3 Fields

Field	Function
31 DOZEN	Doze Enable Disables FlexIO operation in Doze modes. 0b - FlexIO enabled in Doze modes. 1b - FlexIO disabled in Doze modes.
30 DBGE	Debug Enable Enables FlexIO operation in Debug mode. 0b - FlexIO is disabled in debug modes. 1b - FlexIO is enabled in debug modes
29-3 —	Reserved.
2 FASTACC	Fast Access Enables fast register accesses to FlexIO registers, but requires the FlexIO functional clock to be at least twice the frequency of the bus clock. 0b - Configures for normal register accesses to FlexIO 1b - Configures for fast register accesses to FlexIO
1 SWRST	Software Reset The FlexIO Control Register is not affected by the software reset, all other logic in the FlexIO is affected by the software reset and register accesses are ignored until this bit is cleared. This register bit will remain set until cleared by software, and the reset has cleared in the FlexIO clock domain. 0b - Software reset is disabled 1b - Software reset is enabled, all FlexIO registers except the Control Register are reset.
0 FLEXEN	FlexIO Enable 0b - FlexIO module is disabled. 1b - FlexIO module is enabled.

50.3.1.5 Pin State Register (PIN)

50.3.1.5.1 Offset

Register	Offset
PIN	Ch

50.3.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDI															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.5.3 Fields

Field	Function
31-0 PDI	Pin Data Input Returns the input data on each of the FlexIO pins.

50.3.1.6 Shifter Status Register (SHIFTSTAT)

50.3.1.6.1 Offset

Register	Offset
SHIFTSTAT	10h

50.3.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.6.3 Fields

Field	Function
31-8 —	Reserved.
7-0 SSF	Shifter Status Flag The shifter status flag is updated when one of the following events occurs: For SMOD=Receive, the status flag is set when SHIFTBUF has been loaded with data from Shifter (SHIFTBUF is full), and the status flag is cleared when SHIFTBUF register is read. For SMOD=Transmit, the status flag is set when SHIFTBUF data has been transferred to the Shifter (SHIFTBUF is empty) or when initially configured for SMOD=Transmit, and the status flag is cleared when the SHIFTBUF register is written. For SMOD=Match Store, the status flag is set when a match has occured between SHIFTBUF and Shifter, and the status flag is cleared when the SHIFTBUF register is read. For SMOD=Match Continuous, returns the current match result between the SHIFTBUF and Shifter. The status flag cannot be cleared by reading SHIFTBUF. For SMOD=State, the status flag for a shifter will set when it is selected by the current state pointer. For SMOD=Logic, returns the current value of the programmable logic block output. The status flag can also be cleared by writing a logic one to the flag for all modes except Match Continuous/State/Logic. 0b - Status flag is clear. 1b - Status flag is set.

50.3.1.7 Shifter Error Register (SHIFTERR)

50.3.1.7.1 Offset

Register	Offset
SHIFTERR	14h

50.3.1.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SEF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.7.3 Fields

Field	Function
31-8 —	Reserved.
7-0 SEF	Shifter Error Flags The shifter error flag is set when one of the following events occurs: For SMOD=Receive, indicates Shifter was ready to store new data into SHIFTBUF before the previous data was read from SHIFTBUF (SHIFTBUF Overrun), or indicates that the received start or stop bit does not match the expected value. For SMOD=Transmit, indicates Shifter was ready to load new data from SHIFTBUF before new data had been written into SHIFTBUF (SHIFTBUF Underrun). For SMOD=Match Store, indicates a match event occured before the previous match data was read from SHIFTBUF (SHIFTBUF Overrun). For SMOD=Match Continuous, the error flag is set when a match has occured between SHIFTBUF and Shifter. For SMOD=Logic, the error flag is set when the output of the programmable logic block has asserted. Can be cleared by writing logic one to the flag. For SMOD=Match Continuous, can also be cleared when the SHIFTBUF register is read. 0b - Shifter Error Flag is clear. 1b - Shifter Error Flag is set.

50.3.1.8 Timer Status Register (TIMSTAT)

50.3.1.8.1 Offset

Register	Offset
TIMSTAT	18h

50.3.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TSF							
W									W1C							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.8.3 Fields

Field	Function
31-8 —	Reserved.
7-0 TSF	Timer Status Flags The timer status flag sets depending on the timer mode, and can be cleared by writing logic one to the flag. In 8-bit baud counter mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 8-bit high PWM mode, the timer status flag is set when the upper 8-bit counter equals zero and decrements. In 16-bit counter mode, the timer status flag is set when the 16-bit counter equals zero and decrements. 0b - Timer Status Flag is clear. 1b - Timer Status Flag is set.

50.3.1.9 Shifter Status Interrupt Enable (SHIFTSIEN)

50.3.1.9.1 Offset

Register	Offset
SHIFTSIEN	20h

50.3.1.9.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								SSIE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.9.3 Fields

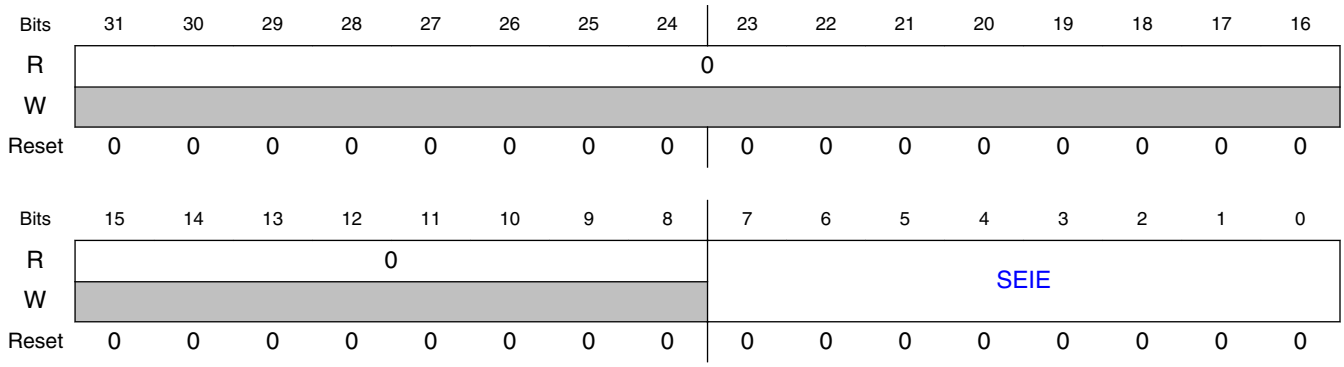
Field	Function
31-8 —	Reserved.
7-0 SSIE	Shifter Status Interrupt Enable Enables interrupt generation when corresponding SSF is set. 0b - Shifter Status Flag interrupt disabled. 1b - Shifter Status Flag interrupt enabled.

50.3.1.10 Shifter Error Interrupt Enable (SHIFTEIEN)

50.3.1.10.1 Offset

Register	Offset
SHIFTEIEN	24h

50.3.1.10.2 Diagram



50.3.1.10.3 Fields

Field	Function
31-8 —	Reserved.
7-0 SEIE	Shifter Error Interrupt Enable Enables interrupt generation when corresponding SEF is set. 0b - Shifter Error Flag interrupt disabled. 1b - Shifter Error Flag interrupt enabled.

50.3.1.11 Timer Interrupt Enable Register (TIMIEN)

50.3.1.11.1 Offset

Register	Offset
TIMIEN	28h

50.3.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TEIE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.11.3 Fields

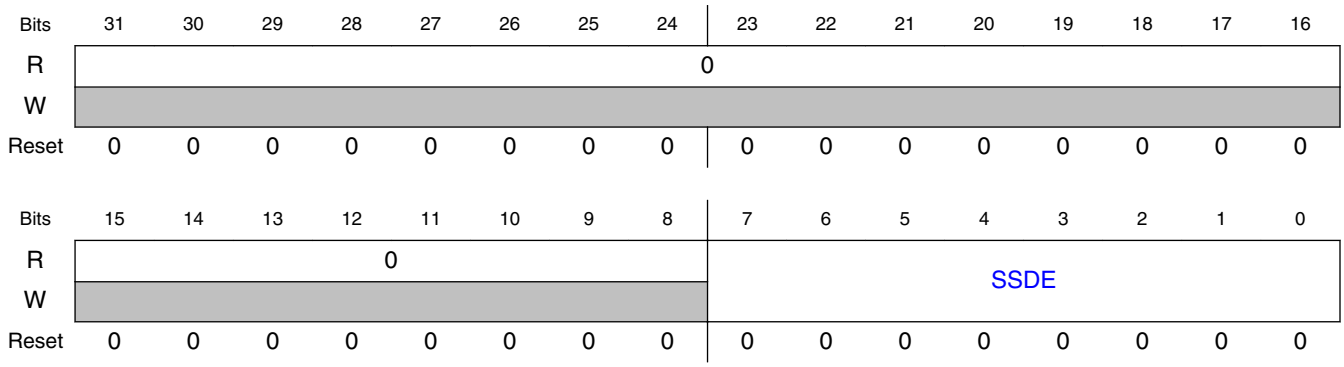
Field	Function
31-8 —	Reserved.
7-0 TEIE	Timer Status Interrupt Enable Enables interrupt generation when corresponding TSF is set. 0b - Timer Status Flag interrupt is disabled. 1b - Timer Status Flag interrupt is enabled.

50.3.1.12 Shifter Status DMA Enable (SHIFTSDEN)

50.3.1.12.1 Offset

Register	Offset
SHIFTSDEN	30h

50.3.1.12.2 Diagram



50.3.1.12.3 Fields

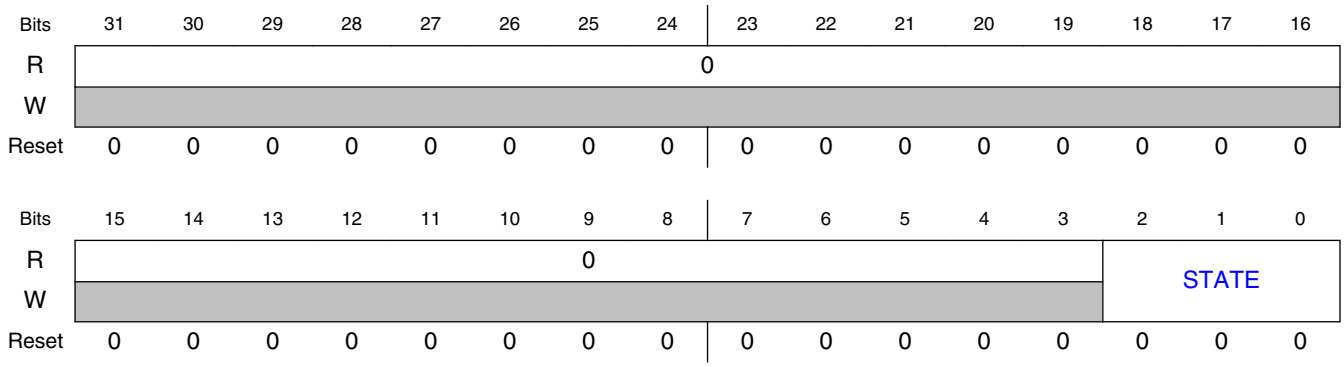
Field	Function
31-8 —	Reserved.
7-0 SSDE	Shifter Status DMA Enable Enables DMA request generation when corresponding SSF is set. 0b - Shifter Status Flag DMA request is disabled. 1b - Shifter Status Flag DMA request is enabled.

50.3.1.13 Shifter State Register (SHIFTSTATE)

50.3.1.13.1 Offset

Register	Offset
SHIFTSTATE	40h

50.3.1.13.2 Diagram



50.3.1.13.3 Fields

Field	Function
31-3 —	Reserved.
2-0 STATE	Current State Pointer The current state field maintains a pointer to keep track of the current Shifter (configured for State mode) enabled to drive outputs and compute the next state. Reading this register when the state pointer is updating can result in the incorrect state returned.

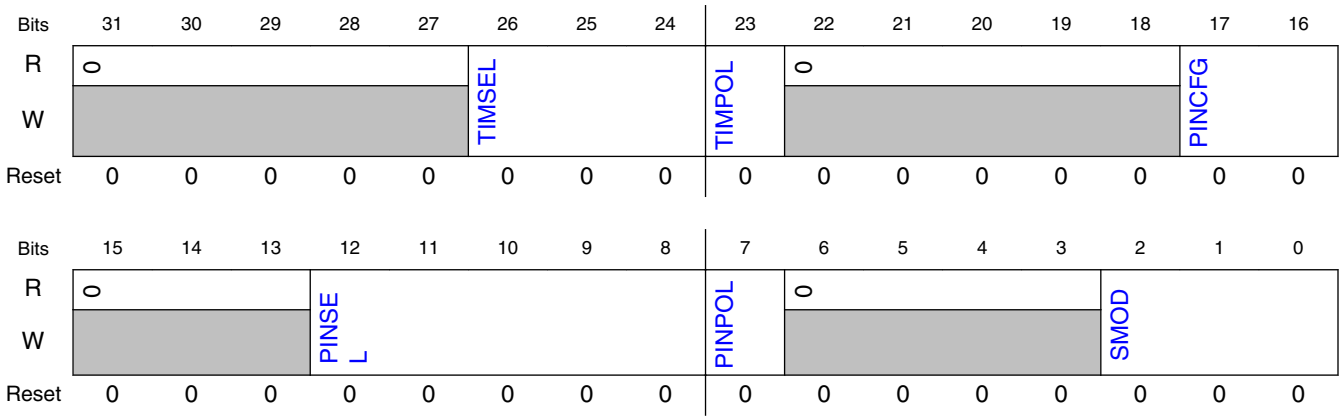
50.3.1.14 Shifter Control N Register (SHIFTCTL0 - SHIFTCTL7)

50.3.1.14.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTCTLa	80h + (a × 4h)

50.3.1.14.2 Diagram



50.3.1.14.3 Fields

Field	Function
31-27 —	Reserved.
26-24 TIMSEL	Timer Select Selects which Timer is used for controlling the logic/shift register and generating the Shift clock. TIMSEL=i will select TIMERi.
23 TIMPOL	Timer Polarity 0b - Shift on posedge of Shift clock 1b - Shift on negedge of Shift clock
22-18 —	Reserved.
17-16 PINCFG	Shifter Pin Configuration For pins configured as an output (PINCFG=11), this field will take effect when the register is written. 00b - Shifter pin output disabled 01b - Shifter pin open drain or bidirectional output enable 10b - Shifter pin bidirectional output data 11b - Shifter pin output
15-13 —	Reserved.
12-8 PINSEL	Shifter Pin Select Selects which pin is used by the Shifter input or output. PINSEL=i will select the FXIO_Di pin. For pins configured as an output (PINCFG=11), this field will take effect when the register is written.
7 PINPOL	Shifter Pin Polarity For pins configured as an output (PINCFG=11), this field will take effect when the register is written. 0b - Pin is active high 1b - Pin is active low
6-3 —	Reserved.

Table continues on the next page...

Field	Function
2-0 SMOD	Shifter Mode Configures the mode of the Shifter. 000b - Disabled. 001b - Receive mode. Captures the current Shifter content into the SHIFTBUF on expiration of the Timer. 010b - Transmit mode. Load SHIFTBUF contents into the Shifter on expiration of the Timer. 011b - Reserved. 100b - Match Store mode. Shifter data is compared to SHIFTBUF content on expiration of the Timer. 101b - Match Continuous mode. Shifter data is continuously compared to SHIFTBUF contents. 110b - State mode. SHIFTBUF contents are used for storing programmable state attributes. 111b - Logic mode. SHIFTBUF contents are used for implementing programmable logic look up table.

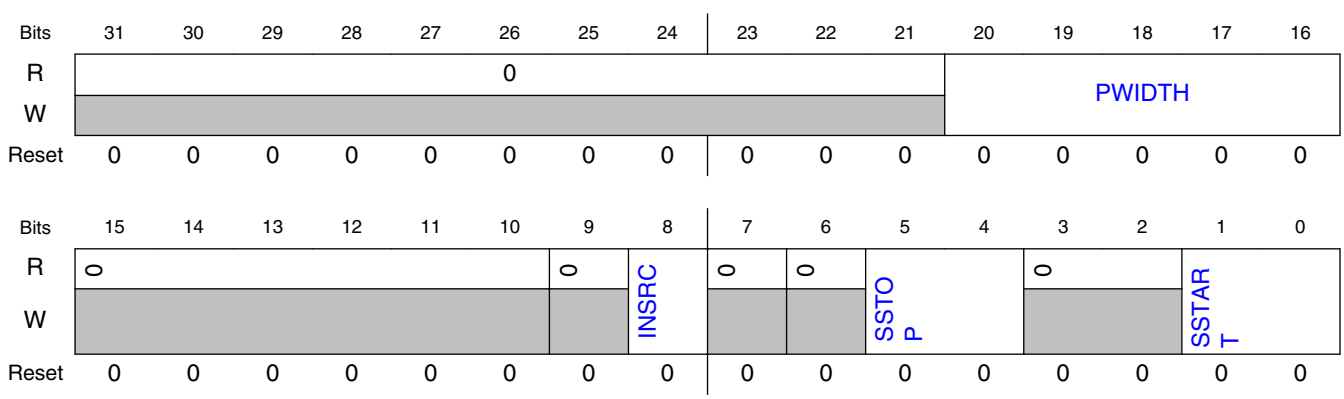
50.3.1.15 Shifter Configuration N Register (SHIFTCFG0 - SHIFTCFG7)

50.3.1.15.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTCFGa	100h + (a × 4h)

50.3.1.15.2 Diagram



50.3.1.15.3 Fields

Field	Function
31-21	Reserved.

Table continues on the next page...

Field	Function
—	
20-16 PWIDTH	Parallel Width For all Shifters, this register field configures the number of bits to be shifted on each Shift clock as follows: 1-bit shift for PWIDTH=0 4-bit shift for PWIDTH=1...3 8-bit shift for PWIDTH=4...7 16-bit shift for PWIDTH=8...15 32-bit shift for PWIDTH=16...31 For Shifters which support parallel transmit (SHIFTER0, SHIFTER4, ...) or parallel receive (SHIFTER3, SHIFTER7, ...), this register field, together with SHIFTCTL[PINSEL], also selects the pins to be driven or sampled on each Shift clock as follows: • FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL] Shifters which do not support parallel transmit or parallel receive only support parallel shift when INSRC=1. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail.
15-10 —	Reserved.
9 —	Reserved.
8 INSRC	Input Source Selects the input source for the shifter. Configuring INSRC=1 is not supported for the last shifter. 0b - Pin 1b - Shifter N+1 Output
7 —	Reserved.
6 —	Reserved.
5-4 SSTOP	Shifter Stop bit For SMOD=Transmit, this field allows automatic stop bit insertion if the selected timer has also enabled a stop bit. For SMOD=Receive or Match Store, this field allows automatic stop bit checking if the selected timer has also enabled a stop bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Stop bit disabled for transmitter/receiver/match store 01b - Reserved for transmitter/receiver/match store 10b - Transmitter outputs stop bit value 0 on store, receiver/match store sets error flag if stop bit is not 0 11b - Transmitter outputs stop bit value 1 on store, receiver/match store sets error flag if stop bit is not 1
3-2 —	Reserved.

Table continues on the next page...

Field	Function
1-0 SSTART	Shifter Start bit For SMOD=Transmit, this field allows automatic start bit insertion if the selected timer has also enabled a start bit. For SMOD=Receive or Match Store, this field allows automatic start bit checking if the selected timer has also enabled a start bit. For SMOD=State, this field is used to disable state outputs. See 'State Mode' section for more detail. For SMOD=Logic, this field is used to mask logic pin inputs. See 'Logic Mode' section for more detail. 00b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on enable 01b - Start bit disabled for transmitter/receiver/match store, transmitter loads data on first shift 10b - Transmitter outputs start bit value 0 before loading data on first shift, receiver/match store sets error flag if start bit is not 0 11b - Transmitter outputs start bit value 1 before loading data on first shift, receiver/match store sets error flag if start bit is not 1

50.3.1.16 Shifter Buffer N Register (SHIFTBUF0 - SHIFTBUF7)

50.3.1.16.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFa	200h + (a × 4h)

50.3.1.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUF															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.16.3 Fields

Field	Function
31-0 SHIFTBUF	Shift Buffer Shift buffer data is used for a variety of functions depending on the SMOD setting: For SMOD=Receive, Shifter data is transferred into SHIFTBUF at the expiration of Timer. The SHIFTBUF register should only be read when the corresponding shifter status flag is set, indicating new Shifter data is available. For SMOD=Transmit, SHIFTBUF data is transferred into the Shifter before the Timer begins. For SMOD=Match Store, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). The Match is checked when the Timer expires and Shifter data [31:16] is written to SHIFTBUF[31:16] whenever a match event occurs. The SHIFTBUF register should only be read when the corresponding shifter status flag is set, indicating new Shifter data is available. For SMOD=Match Continuous, SHIFTBUF[31:16] contains the data to be matched with the Shifter contents and SHIFTBUF[15:0] can be used to mask the match result (1=mask, 0=no mask). For SMOD=Logic, SHIFTBUF[31:0] is used to implement a 5-input, 32-bit programmable logic look-up table. See 'Logic Mode' section for more detail. For SMOD=State, SHIFTBUF[31:24] is used to drive the output value when this shifter is selected by the current state pointer and SHIFTBUF[23:0] is used to configure the value of the next state transition. See 'State Mode' section for more detail.

50.3.1.17 Shifter Buffer N Bit Swapped Register (SHIFTBUFBIS0 - SHIFTBUFBIS7)

50.3.1.17.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFBISa	280h + (a × 4h)

50.3.1.17.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.17.3 Fields

Field	Function
31-0 SHIFTBUFBIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped. Reads return SHIFTBUF[0:31].

50.3.1.18 Shifter Buffer N Byte Swapped Register (SHIFTBUFBYS0 - SHIFTBUFBYS7)

50.3.1.18.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFBYSa	300h + (a × 4h)

50.3.1.18.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBYS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.18.3 Fields

Field	Function
31-0 SHIFTBUFBYS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are byte swapped. Reads return { SHIFTBUF[7:0], SHIFTBUF[15:8], SHIFTBUF[23:16], SHIFTBUF[31:24] }.

50.3.1.19 Shifter Buffer N Bit Byte Swapped Register (SHIFTBUFBBS0 - SHIFTBUFBBS7)

50.3.1.19.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFBBSa	380h + (a × 4h)

50.3.1.19.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFBBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.19.3 Fields

Field	Function
31-0 SHIFTBUFBBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are bit swapped within each byte. Reads return { SHIFTBUF[24:31], SHIFTBUF[16:23], SHIFTBUF[8:15], SHIFTBUF[0:7] }.

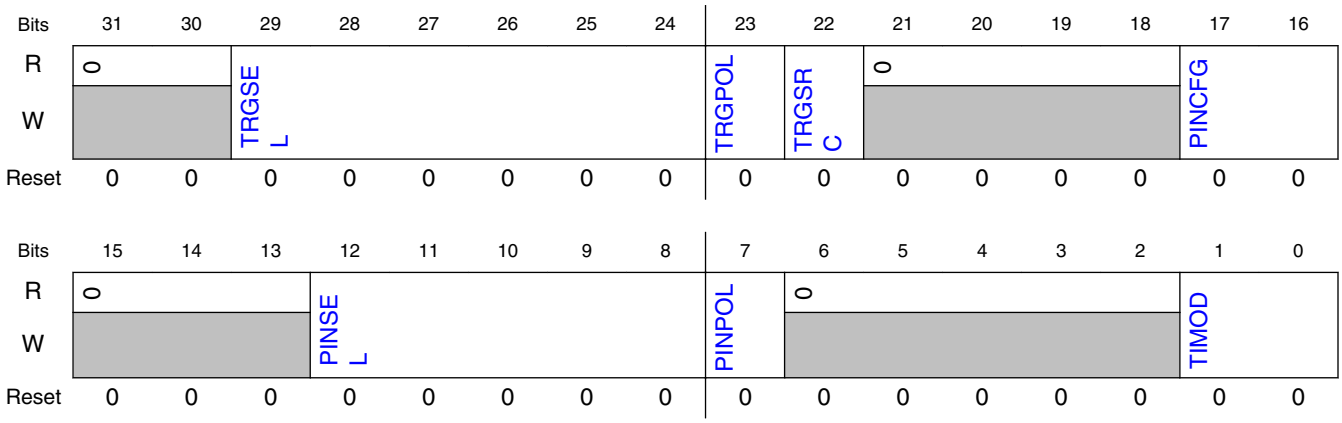
50.3.1.20 Timer Control N Register (TIMCTL0 - TIMCTL7)

50.3.1.20.1 Offset

For a = 0 to 7:

Register	Offset
TIMCTLa	400h + (a × 4h)

50.3.1.20.2 Diagram



50.3.1.20.3 Fields

Field	Function
31-30 —	Reserved.
29-24 TRGSEL	Trigger Select The valid values for TRGSEL will depend on the FLEXIO_PARAM register. • When TRGSRC = 1, the valid values for N will depend on PIN, TIMER, SHIFTER fields in the FLEXIO_PARAM register. • When TRGSRC = 0, the valid values for N will depend on TRIGGER field in FLEXIO_PARAM register. Refer to the chip-specific FlexIO information for external trigger selection. **NOTE:** For a pin, N=0 to 31. For a Shifter, N=0 to 7. For a Timer, N=0 to 7. When TRGSRC = 0 the trigger selection is configured as follows: • N - External trigger N input When TRGSRC = 1 the internal trigger can be configured to select an input pin as follows: • 2*N - Pin N input When TRGSRC = 1 the internal trigger can be configured to select a shifter/timer signal as follows: • 4*N+1 - Shifter N status flag • 4*N+3 - Timer N trigger output In other words, the expanded internal trigger selection (TRGSRC = 1) is as follows: • 0000 = Pin 0 • 0001 = Shifter 0 Flag • 0010 = Pin 1 • 0011 = Timer 0 Trigger • 0100 = Pin 2 • 0101 = Shifter 1 Flag • 0110 = Pin 3 • 0111 = Timer 1 Trigger • ... • This continues up to the Pin 31, Shifter 7 and Timer 7.
23	Trigger Polarity

Table continues on the next page...

Field	Function
TRGPOL	0b - Trigger active high 1b - Trigger active low
22 TRGSRC	Trigger Source 0b - External trigger selected 1b - Internal trigger selected
21-18 —	Reserved.
17-16 PINCFG	Timer Pin Configuration Configures the direction of the Timer pin. For pins configured as an output (PINCFG=11), this field will take effect when the register is written. 00b - Timer pin output disabled 01b - Timer pin open drain or bidirectional output enable 10b - Timer pin bidirectional output data 11b - Timer pin output
15-13 —	Reserved.
12-8 PINSEL	Timer Pin Select Selects which pin is used by the Timer input or output. PINSEL=i will select the FXIO_Di pin. For pins configured as an output (PINCFG=11), this field will take effect when the register is written.
7 PINPOL	Timer Pin Polarity For pins configured as an output (PINCFG=11), this field will take effect when the register is written. 0b - Pin is active high 1b - Pin is active low
6-2 —	Reserved.
1-0 TIMOD	Timer Mode In 8-bit baud counter mode, the lower 8-bits of the counter and compare register are used to configure the baud rate of the timer shift clock, and the upper 8-bits are used to configure the shifter bit count. In 8-bit PWM high mode, the lower 8-bits of the counter and compare register are used to configure the high period of the timer shift clock, and the upper 8-bits are used to configure the low period of the timer shift clock. The shifter bit count is configured using another timer or external signal. In 16-bit counter mode, the full 16-bits of the counter and compare register are used to configure either the baud rate of the shift clock or the shifter bit count. 00b - Timer Disabled. 01b - Dual 8-bit counters baud mode. 10b - Dual 8-bit counters PWM high mode. 11b - Single 16-bit counter mode.

50.3.1.21 Timer Configuration N Register (TIMCFG0 - TIMCFG7)

50.3.1.21.1 Offset

For a = 0 to 7:

Register	Offset
TIMCFGa	480h + (a × 4h)

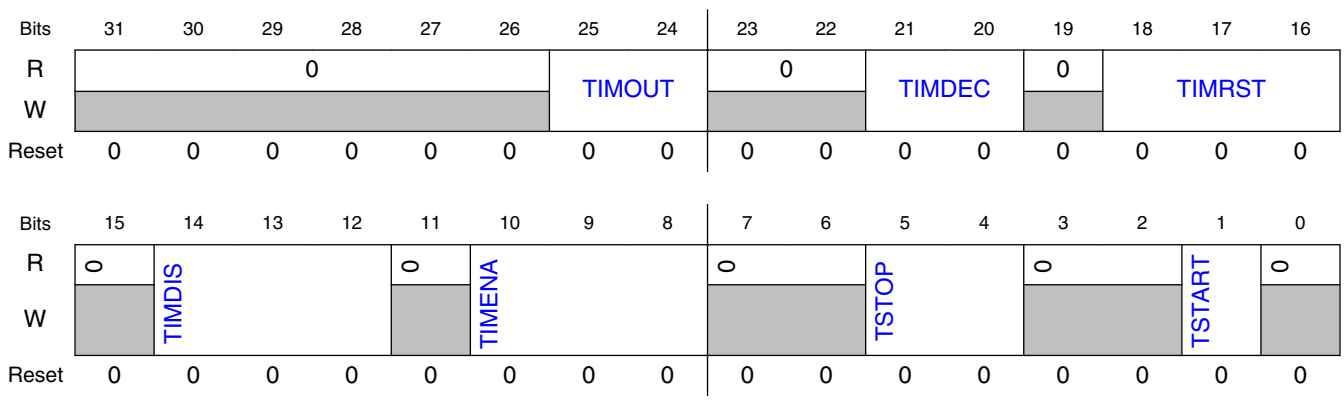
50.3.1.21.2 Function

The options to enable or disable the timer using the Timer N-1 enable or disable are reserved when N is evenly divisible by 4 (eg: Timer 0).

NOTE

The pin and trigger level/edges specified below refer to the signal state after being modified by the PINPOL or TRGPOL setting in the TIMCTL register. For example, "Trigger low" means a trigger is actually at logic level 1 if the TRGPOL is set to 1 (active low).

50.3.1.21.3 Diagram



50.3.1.21.4 Fields

Field	Function
31-26 —	Reserved.
25-24 TIMOUT	Timer Output Configures the initial state of the Timer Output and whether it is affected by the Timer reset. 00b - Timer output is logic one when enabled and is not affected by timer reset 01b - Timer output is logic zero when enabled and is not affected by timer reset 10b - Timer output is logic one when enabled and on timer reset 11b - Timer output is logic zero when enabled and on timer reset
23-22 —	Reserved.

Table continues on the next page...

Field	Function
21-20 TIMDEC	Timer Decrement Configures the source of the Timer decrement and the source of the Shift clock. 00b - Decrement counter on FlexIO clock, Shift clock equals Timer output. 01b - Decrement counter on Trigger input (both edges), Shift clock equals Timer output. 10b - Decrement counter on Pin input (both edges), Shift clock equals Pin input. 11b - Decrement counter on Trigger input (both edges), Shift clock equals Trigger input.
19 —	Reserved.
18-16 TIMRST	Timer Reset Configures the condition that causes the timer counter (and optionally the timer output) to be reset. In 8-bit counter mode, the timer reset will only reset the lower 8-bits that configure the baud rate. In all other modes, the timer reset will reset the full 16-bits of the counter. 000b - Timer never reset 001b - Reserved 010b - Timer reset on Timer Pin equal to Timer Output 011b - Timer reset on Timer Trigger equal to Timer Output 100b - Timer reset on Timer Pin rising edge 101b - Reserved 110b - Timer reset on Trigger rising edge 111b - Timer reset on Trigger rising or falling edge
15 —	Reserved.
14-12 TIMDIS	Timer Disable Configures the condition that causes the Timer to be disabled and stop decrementing. 000b - Timer never disabled 001b - Timer disabled on Timer N-1 disable 010b - Timer disabled on Timer compare (upper 8-bits match and decrement) 011b - Timer disabled on Timer compare (upper 8-bits match and decrement) and Trigger Low 100b - Timer disabled on Pin rising or falling edge 101b - Timer disabled on Pin rising or falling edge provided Trigger is high 110b - Timer disabled on Trigger falling edge 111b - Reserved
11 —	Reserved.
10-8 TIMENA	Timer Enable Configures the condition that causes the Timer to be enabled and start decrementing. 000b - Timer always enabled 001b - Timer enabled on Timer N-1 enable 010b - Timer enabled on Trigger high 011b - Timer enabled on Trigger high and Pin high 100b - Timer enabled on Pin rising edge 101b - Timer enabled on Pin rising edge and Trigger high 110b - Timer enabled on Trigger rising edge 111b - Timer enabled on Trigger rising or falling edge
7-6 —	Reserved.
5-4 TSTOP	Timer Stop Bit The stop bit can be added on a timer compare (between each word) or on a timer disable. When stop bit is enabled, configured shifters will output the contents of the stop bit when the timer is disabled. When

Table continues on the next page...

Field	Function
	stop bit is enabled on timer disable, the timer remains disabled until the next rising edge of the shift clock. If configured for both timer compare and timer disable, only one stop bit is inserted on timer disable. 00b - Stop bit disabled 01b - Stop bit is enabled on timer compare 10b - Stop bit is enabled on timer disable 11b - Stop bit is enabled on timer compare and timer disable
3-2 —	Reserved.
1 TSTART	Timer Start Bit When start bit is enabled, configured shifters will output the contents of the start bit when the timer is enabled and the timer counter will reload from the compare register on the first rising edge of the shift clock. 0b - Start bit disabled 1b - Start bit enabled
0 —	Reserved.

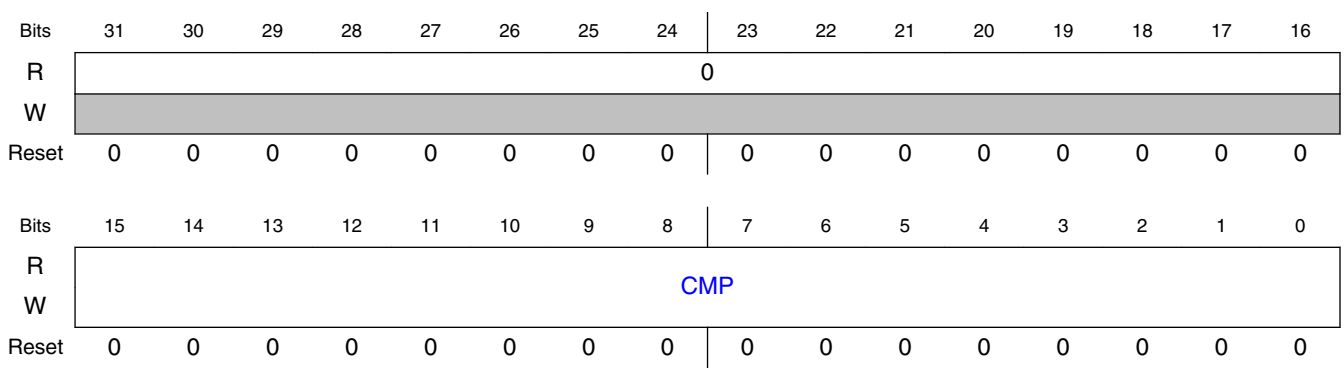
50.3.1.22 Timer Compare N Register (TIMCMP0 - TIMCMP7)

50.3.1.22.1 Offset

For a = 0 to 7:

Register	Offset
TIMCMPa	500h + (a × 4h)

50.3.1.22.2 Diagram



50.3.1.22.3 Fields

Field	Function
31-16 —	Reserved.
15-0 CMP	Timer Compare Value The timer compare value is loaded into the timer counter when the timer is first enabled, when the timer is reset and when the timer decrements down to zero. In 8-bit baud counter mode, the lower 8-bits configure the baud rate divider equal to (CMP[7:0] + 1) * 2. The upper 8-bits configure the number of bits in each word equal to (CMP[15:8] + 1) / 2. In 8-bit PWM high mode, the lower 8-bits configure the high period of the output to (CMP[7:0] + 1) and the upper 8-bits configure the low period of the output to (CMP[15:8] + 1). In 16-bit counter mode, the compare value can be used to generate the baud rate divider (if shift clock source is timer output) to equal (CMP[15:0] + 1) * 2. When the shift clock source is a pin or trigger input, the compare register is used to set the number of bits in each word equal to (CMP[15:0] + 1) / 2.

50.3.1.23 Shifter Buffer N Nibble Byte Swapped Register (SHIFTBUFNBS0 - SHIFTBUFNBS7)

50.3.1.23.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFNBSa	680h + (a × 4h)

50.3.1.23.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNBS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.23.3 Fields

Field	Function
31-0 SHIFTBUFNBS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped within each byte. Reads return { SHIFTBUF[27:24], SHIFTBUF[31:28], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[3:0], SHIFTBUF[7:4] }.

50.3.1.24 Shifter Buffer N Half Word Swapped Register (SHIFTBUFHWS0 - SHIFTBUFHWS7)

50.3.1.24.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFHWSa	700h + (a × 4h)

50.3.1.24.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFHWS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFHWS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.24.3 Fields

Field	Function
31-0 SHIFTBUFHWS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are half word swapped. Reads return { SHIFTBUF[15:0], SHIFTBUF[31:16] }.

50.3.1.25 Shifter Buffer N Nibble Swapped Register (SHIFTBUFNIS0 - SHIFTBUFNIS7)

50.3.1.25.1 Offset

For a = 0 to 7:

Register	Offset
SHIFTBUFNISa	780h + (a × 4h)

50.3.1.25.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SHIFTBUFNIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50.3.1.25.3 Fields

Field	Function
31-0 SHIFTBUFNIS	Shift Buffer Alias to SHIFTBUF register, except reads/writes to this register are nibble swapped. Reads return { SHIFTBUF[3:0], SHIFTBUF[7:4], SHIFTBUF[11:8], SHIFTBUF[15:12], SHIFTBUF[19:16], SHIFTBUF[23:20], SHIFTBUF[27:24], SHIFTBUF[31:28] }.

50.4 Functional description

50.4.1 Clocking and Resets

50.4.1.1 Functional clock

The FlexIO functional clock is asynchronous to the bus clock and can remain enabled in low power modes. The FlexIO functional clock must be enabled before accessing any FlexIO registers. Provided the FlexIO functional clock is at least two times faster than the bus clock, the CTRL[FASTACC] bit can be set to support fast register accesses.

50.4.1.2 Bus clock

The bus clock is only used for bus accesses to the control and configuration registers.

50.4.1.3 Chip reset

The logic and registers for the FlexIO are reset to their default state on a chip reset.

50.4.1.4 Software reset

The FlexIO implements a software reset bit in its Control Register. The CTRL[RST] will reset all logic and registers to their default state, except for the CTRL itself.

50.4.2 Shifter operation

Shifters are responsible for buffering and shifting data into or out of the FlexIO. The timing of shift, load and store events are controlled by the Timer assigned to the Shifter via the SHIFTCTL[TIMSEL] register. The Shifters are designed to support either DMA, interrupt or polled operation. The following block diagram provides a detailed view of the Shifter microarchitecture.

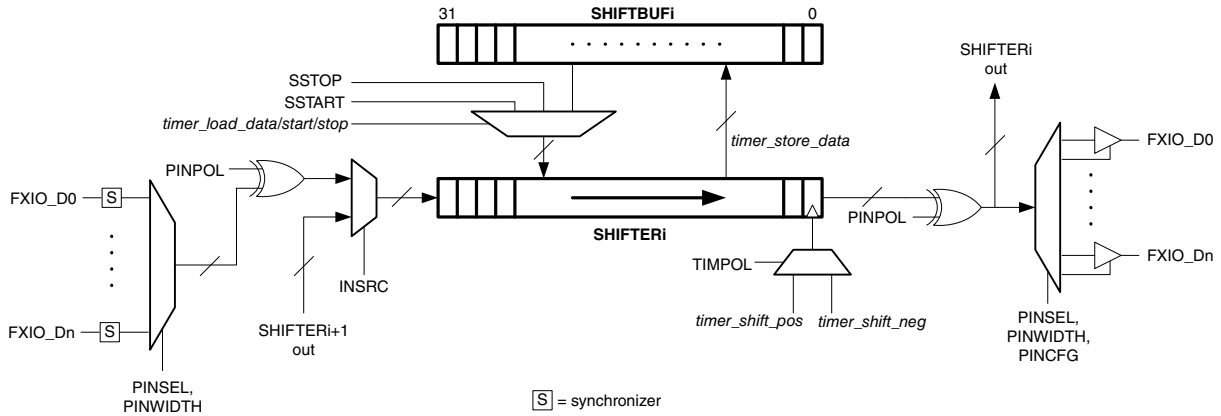


Figure 50-2. Shifter Microarchitecture

50.4.2.1 Transmit Mode

When configured for Transmit mode (SHIFTCTL[SMOD]=Transmit), the shifter will load data from the SHIFTBUF register and shift data out when a load event is signalled by the assigned Timer. An optional start/stop bit can also be automatically loaded before/after SHIFTBUF data by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Note that the shifter will immediately load a stop bit when the Shifter is initially configured for Transmit mode if a stop bit is enabled.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when data has been loaded from the SHIFTBUF register into the Shifter or when the Shifter is initially configured into Transmit mode. The flag will clear when new data has been written into the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to load data from an empty SHIFTBUF register occurs (buffer underrun). The flag can be cleared by writing it with logic 1.

50.4.2.2 Receive Mode

When configured for Receive mode (SHIFTCTL[SMOD]=Receive), the shifter will shift data in and store data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when data has been stored into the SHIFTBUF register from the Shifter. The flag will clear when the data has been read from the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to store data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

50.4.2.3 Match Store Mode

When configured for Match Store mode (SHIFTCTL[SMOD]=Match Store), the shifter will shift data in, check for a match result and store matched data into the SHIFTBUF register when a store event is signalled by the assigned Timer. Checking for a start/stop bit can be enabled before/after shifter data is sampled by configuring the SHIFTCFG[SSTART], TIMCFG[TSTART] or SHIFTCFG[SSTOP], TIMCFG[TSTOP] registers in the Shifter and Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when a match occurs and matched data has been stored into the SHIFTBUF register from the Shifter. The flag will clear when the matched data has been read from the SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when an attempt to store matched data into a full SHIFTBUF register occurs (buffer overrun) or when a mismatch occurs on a start/stop bit check. The flag can be cleared by writing it with logic 1.

50.4.2.4 Match Continuous Mode

When configured for Match Continuous mode (SHIFTCTL[SMOD]=Match Continuous), the shifter will shift data in and continuously check for a match result whenever a shift event is signalled by the assigned Timer. Up to 16-bits of data can be compared using SHIFTBUF[31:16] to configure the data to be matched and SHIFTBUF[15:0] to mask the match result.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set when a match occurs. The flag will clear automatically as soon as there is no longer a match between Shifter data and SHIFTBUF register. The flag cannot be cleared by reading SHIFTBUF register.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when a match occurs. The flag will clear when there is a read from the SHIFTBUF register or it written with logic 1.

50.4.2.5 State Mode

Using State mode enables the user to implement any state machine with up to 8 states, 8 outputs and 3 selectable inputs per state. This feature allows basic control functions to be offloaded from the CPU, which could potentially remain in a STOP/VLPS low power mode.

When configured for State mode (SHIFTCTL[SMOD]=State), the SHIFTBUF register is used to drive the output and compute next state values when the Shifter has been selected by the current state pointer (SHIFTSTATE[STATE]). The following diagram provides a detailed view of Shifter microarchitecture when configured for State mode.

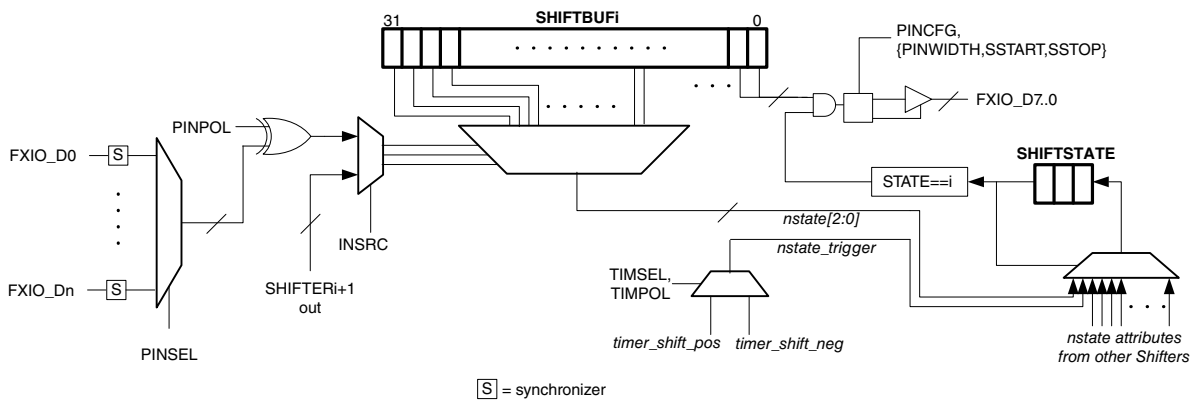


Figure 50-3. State Microarchitecture

When Shifter i has been selected by the current state pointer, output pins FXIO_D[7:0] will be driven by SHIFTBUFi[31:24] using the configuration set by SHIFTCTLi[PINCFG]. When set, SHIFTCFGi{PWIDTH[3:0],SSTOP[1:0],SSTART[1:0]} are respectively used to disable the output drive on pins FXIO_D[7:0] for state machine applications which require less than 8 output pins.

The next state value is computed using the 3 input pins selected by SHIFTCTLi[PINSEL] together with SHIFTBUFi[23:0]. Note that each state could potentially use a different set of 3 input pins. The following table details how the next state value is computed when the current state pointer is pointing to Shifter i.

Table 50-4. Next State computation for SHIFTSTATE[STATE]=i

FXIO_D[PINSEL+2]	FXIO_D[PINSEL+1]	FXIO_D[PINSEL]	Next State Value

Table continues on the next page...

Table 50-4. Next State computation for SHIFTSTATE[STATE]=i (continued)

0	0	0	SHIFTBUFi[2:0]
0	0	1	SHIFTBUFi[5:3]
0	1	0	SHIFTBUFi[8:6]
0	1	1	SHIFTBUFi[11:9]
...	...	...	...
1	1	1	SHIFTBUFi[23:21]

Note that other shifters/timers could potentially be configured to drive the input pins of a given state, allowing the user to create complex combinations of shifters/timers as desired e.g. the output of a Shifter configured for logic mode could potentially be used to drive a state machine input.

The next state transition is triggered using the Timer output selected by SHIFTCTLi[TIMSEL] with polarity controlled by SHIFTCTLi[TIMPOL]. Note that each state could potentially use a different Timer to trigger each next state transition, allowing a variety of internal/external trigger sources and clocking configurations to be used (see Timer section for more detail).

The current state pointer defaults to Shifter 0 at reset, however it can be written by the user to select a different Shifter for the initial state. If the current state pointer selects a Shifter which is not configured for State mode, then outputs will not be driven and a next state transition is never triggered.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set whenever the Shifter has been selected by the current state pointer. The flag will clear when the current state pointer is updated to a different Shifter.

50.4.2.6 Logic Mode

Using logic mode enables the user to implement a small amount of programmable digital logic within a FlexIO Shifter.

When configured for Logic mode (SHIFTCTL[SMOD]=Logic), the SHIFTBUF register is used to implement a 5-input, 32-bit programmable logic look-up table. The following diagram provides a detailed view of Shifter microarchitecture when configured for Logic mode.

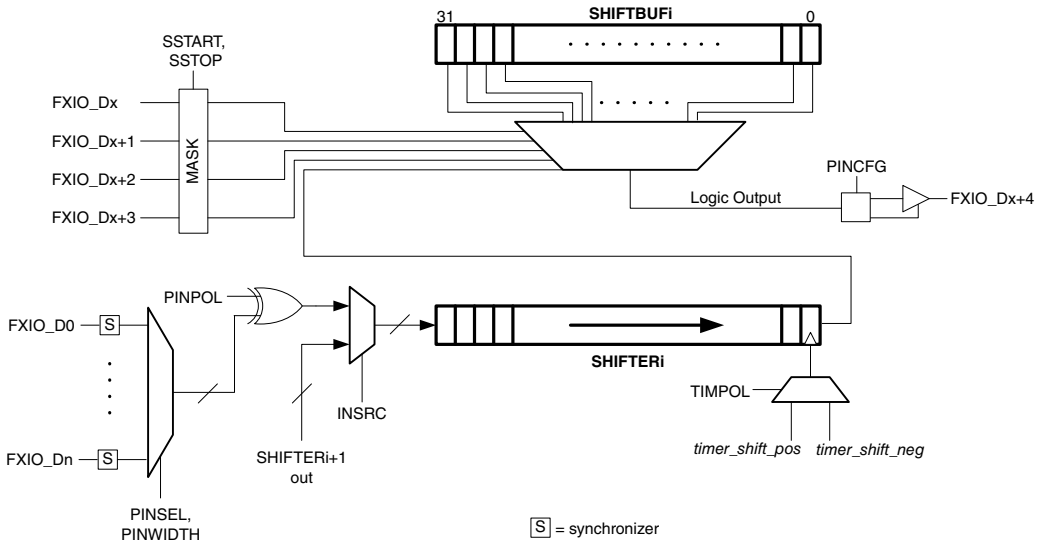


Figure 50-4. Logic Microarchitecture

The look-up table is driven using 4 pin inputs (maskable using SHIFTCFG[SSTOP] and SHFITCFG[SSTART]) plus 1 input from the internal shifter and can be configured to drive an output pin using the SHIFTCTL[PINCFG] field. Pin inputs and outputs are fixed for each logic look-up table and are not selectable. The following table lists the logic output value selected by the look-up table for Shifter 'i'.

Table 50-5. Logic Look-up table for Shifter 'i'

SHIFTERi[0]	FXIO_D[x+3][1]	FXIO_D[x+2]	FXIO_D[x+1]	FXIO_D[x]	Logic Output to FXIO_D[x+4]
0	0	0	0	0	SHIFTBUFi[0]
0	0	0	0	1	SHIFTBUFi[1]
0	0	0	1	0	SHIFTBUFi[2]
0	0	0	1	1	SHIFTBUFi[3]
...	...	...	...	...	...
1	1	1	1	1	SHIFTBUFi[31]

1. for Shifter i=0...3, x=i

   for Shifter i=4...7, x=i+4

To minimize output glitches, SHIFTCFG[SSTOP] and SHIFTCFG[SSTART] can be used to mask unused input pins. When set, {SSTOP[1:0], SSTART[1:0]} will mask FXIO_D[x+3]...FXIO_D[x] inputs respectively, so that any transitions on these pins will not cause the logic output to glitch.

Note that other shifters/timers could potentially be configured to drive the input pins of a given look-up table, allowing the user to concatenate look-up tables or create complex combinations of shifters/timers as desired.

SHIFTCFG[PWIDTH] will control the number of delay stages introduced by the internal shifter input (SHIFTERi[0]). For example, when configured for 1-bit shift (PWIDTH=0), the internal shifter will introduce a 32 Shift clock delay before passing its input (selected by SHIFTCTL[PINSEL]) to the look-up table. When configured for 32-bit shift (PWIDTH=16...31), the internal shifter will introduce a 1 Shift clock delay to its input.

The Shifter Status Flag (SHIFTSTAT[SSF]) and any enabled interrupts or DMA requests will set whenever the output pin allocated to the logic look-up table has a value of 1 (after being synchronized to the FlexIO clock). The flag will clear when the output pin has a value of 0. This also allows the SSF flag to be used as a trigger to a Timer if desired.

The Shifter Error Flag (SHIFTERR[SEF]) and any enabled interrupts will set when the output pin allocated to the logic look-up table has a value of 1. The flag can be cleared by writing it with logic 1.

50.4.3 Timer Operation

The FlexIO 16-bit timers control the loading, shifting and storing of the shift registers, the counters load the contents of the compare register and decrement down to zero on the FlexIO clock. They can perform generic timer functions such as generating a clock or select output or a PWM waveform. Timers can be configured to enable in response to a trigger, pin or shifter condition; decrement always or only on a trigger or pin edge; reset in response to a trigger or pin condition; and disable on a trigger or pin condition or on a timer compare. Timers can optionally include a start condition and/or stop condition.

Each timer operates independently, although a timer can be configured to enable or disable at the same time as the previous timer (eg: timer1 can enable or disable at the same time as timer 0) and a timer output can be used to trigger any other timer. The trigger used by each timer is configured independently and can be configured to be a timer output, shifter status flag, pin input or an external trigger input (refer to the chip-specific FlexIO information for details on the external trigger connections). The trigger configuration is separate from the pin configuration, which can be configured for input, output data or output enable.

The Timer Configuration Register (TIMCFGn) should be configured before setting the Timer Mode (TIMOD).

50.4.3.1 Timer 8-bit Baud Counter Mode

In 8-bit Baud Counter Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the baud rate of the shift clock and the upper 8-bits are used to configure the number of shift clock edges in the transfer. When the lower 8-bits

decrement to zero, the timer output is toggled and the lower 8-bits reload from the compare register. The upper 8-bits decrement when the lower 8-bits equal zero and decrement.

Note that a timer reset event in 8-bit Baud Counter Mode will only reset the lower 8-bit counter, the upper 8-bit counter is not affected and can decrement if the timer reset is configured to update the state of the timer output, and the timer output toggles as a result of the timer reset event.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

50.4.3.2 Timer 8-bit High PWM Mode

In 8-bit High PWM Mode, the 16-bit counter is divided into two 8-bit counters. The lower 8-bits are used to configure the timer output high period and the upper 8-bits are used to configure the timer output low period. The lower 8-bits decrement when the output is high. When the lower 8-bits equal zero and decrement, the timer output is cleared and the lower 8-bits are reloaded from the compare register. The upper 8-bits decrement when the output is low. When the upper 8-bits equal zero and decrement, the timer output is set and the upper 8-bits are reloaded from the compare register.

A timer compare event occurs when the upper 8-bits equal zero and decrements. The timer status flag is set on a timer compare event.

50.4.3.3 Timer 16-bit Counter Mode

In 16-bit Counter Mode, the 16-bit counter can be used to configure either the baud rate of the shift clock (eg: TIMDEC[1:0] != 10 or 11) or the number of shift clock edges in the transfer (eg: TIMDEC[1:0] = 10 or 11). When the 16-bit counter equals zero and decrements, the timer output toggles and the counter reloads from the compare register.

A timer compare event occurs when the 16-bit counter equals zero and decrements. The timer status flag is set on a timer compare event.

50.4.3.4 Timer Enable and Start Bit

When the TIMOD is configured for the desired mode, and the condition configured by timer enable (TIMENA) is detected then the following events occur.

- Timer counter will load the current value of the compare Register and start decrementing as configured by TIMDEC.
- Timer output may update to its initial state depending on the TIMOUT configuration. Shifters that are controlled by this timer do not see this as a rising edge on the timer shift clock.
- Transmit shifters controlled by this timer will either output their start bit value, or load the shift register from the shift buffer and output the first bit, as configured by SSTART.

If the Timer start bit is enabled, the timer counter will reload with the compare register on the first rising edge of the shift clock after the timer starts decrementing. If there is no falling edge on the shift clock before the first rising edge (for example, when TIMOUT=1), a shifter that is configured to shift on falling edge and load on the first shift will not load correctly.

50.4.3.5 Timer Decrement and Reset

The Timer will then generate the timer output and timer shift clock depending on the TIMOD and TIMDEC fields. The shifter clock is either equal to the timer output (when TIMDEC != 10 or 11) or equal to the decrement clock (when TIMDEC = 10 or 11). When TIMDEC is configured to decrement from a pin or trigger, the timer will decrement on both rising and falling edges.

When the Timer is configured to reset as configured in the TIMRST field then the Timer counter will load the current value of the compare register again. The timer output and timer shift clock can be configured to update on timer reset, as configured by TIMOUT. This can result in a timer shift clock edge if the timer output toggles as a result of the timer reset. In 8-bit Baud Counter mode this would also decrement the upper 8-bits of the counter.

In general, when the timer counter decrements to zero a timer compare event is triggered. The timer compare event will cause the timer counter to load the contents of the timer compare register, the timer output to toggle, any configured transmit shift registers to load, and any configured receive shift registers to store. Depending on the timer mode, the timer status flag may also be set.

50.4.3.6 Timer Disable and Stop Bit

When the is Timer is configured to add a stop bit on each compare, the following additional events will occur.

- Transmit shifters controlled by this timer will output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer will store the contents of the shift register in their shift buffer, as configured by SSTOP.
- On the first rising edge of the shifter clock after the compare, the timer counter will reload the current value of the Compare Register.

Transmit shifters must be configured to load on the first shift when the timer is configured to insert a stop bit on each compare.

When the condition configured by timer disable (TIMDIS) is detected, the following events occur.

- Timer counter will reload the current value of the Compare Register and start decrementing as configured by TIMDEC.
- Timer output will clear. Shifters that are controlled by this timer do not see this as a falling edge on the timer shift clock, but can generate a shift event if the timer shift clock would otherwise generate one.
- Transmit shifters controlled by this timer will output their stop bit value (if configured by SSTOP).
- Receive shifters controlled by this timer will store the contents of the shift register in their shift buffer, as configured by SSTOP.

If the timer stop bit is enabled, the timer counter will continue decrementing until the next rising edge of the shift clock is detected, at which point it will finish. Although the timer output is forced low during the stop bit, the timer shift clock can toggle during the stop bit, but does not generate shift events.

A timer enable condition can be detected in the same cycle as a timer disable condition (if timer stop bit is disabled), or on the first rising edge of the shift clock after the disable condition (if stop bit is enabled). Receive shift registers with stop bit enabled will store the contents of the shift register into the shift buffer and verify the state of the input data on the configured shift edge while the timer is in the stop state condition. If there is no configured edge between the timer disable and the next rising edge of the shift clock then the final store and verify do not occur.

50.4.4 Pin operation

The pin configuration for each timer and shifter can be configured to use any FlexIO pin with either polarity. Each timer and shifter can be configured as an input, output data, output enable or bidirectional output. A pin configured for output enable can be used as an open drain (with inverted polarity, since the output enable assertion would cause logic

zero to be output on the pin) or to control the enable on the bidirectional output. Any timer or shifter could be configured to control the output enable for a pin where the bidirectional output data is driven by another timer or shifter.

50.4.4.1 Parallel Interface

Shifters can be configured to use multiple FlexIO pins in parallel using the SHIFTCFG[PWIDTH] field. PWIDTH is used to configure the following settings of a shifter:

1. Number of bits shifted per Shift clock.
2. Number of pins driven by the shifter per Shift clock (only on shifters supporting parallel transmit i.e. SHIFTER0, SHIFTER4).
3. Number of pins sampled by the shifter per Shift clock (only on Shifter supporting parallel receive i.e. SHIFTER3, SHIFTER7).

When configured for parallel shift, either 4, 8, 16 or 32-bits can be shifted on every Shift clock. If an adjacent shifter is selected as the input source (SHIFTCFG[INSRC]=1), the least significant 4, 8, 16 or 32-bits from the adjacent shifter will be sampled on each Shift clock.

For shifters supporting parallel receive (SHIFTER3, SHIFTER7), the shifter can be configured to sample multiple pins (SHIFTCFG[INSRC]=0), with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant bits will be masked with 0 e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[31:24] will sample {0,0,FXIO_D[12:7]} on each Shift clock.

For shifters supporting parallel transmit (SHIFTER0, SHIFTER4), the shifter can be configured to drive multiple pins using SHIFTCTL[PINCFG], with PWIDTH and PINSEL selecting the pins as follows: FXIO_D[PINSEL+PWIDTH]:FXIO_D[PINSEL]. Note that if PWIDTH is less than the number of bits being shifted on each Shift clock, then the most significant pins will not be driven e.g. if PINSEL=7 and PWIDTH=6, then SHIFTER[5:0] will drive only FXIO_D[12:7] on each Shift clock.

50.4.4.2 Pin Synchronization

When configuring a pin as an input (this includes a timer trigger configured as a pin input), the input signal is first synchronized to the FlexIO clock before the signal is used by a timer or shifter. This introduces a small latency of between 0.5 to 1.5 FlexIO clock cycles when using an external pin input to generate an output or control a shifter. This sets the maximum setup time at 1.5 FlexIO clock cycles.

If an input is used by more than one timer or shifter then the synchronization occurs once to ensure any edge is seen on the same cycle by all timers and shifters using that input.

Note that FlexIO pins are also connected internally, configuring a FlexIO shifter or timer to output data on an unused pin will make an internal connection that allows other shifters and timer to use this pin as an input. This allows a shifter output to be used to trigger a timer or a timer output to be shifted into a shifter. This path is also synchronized to the FlexIO clock and therefore incurs a 1 cycle latency.

So when using a Pin input as a Timer Trigger, Timer Clock or Shifter Data Input, the following synchronization delays occur:

1. 0.5 to 1.5 FlexIO clock cycles for external pin
2. 1 FlexIO clock cycle for an internally driven pin

For timing considerations such as output valid time and input setup time for specific applications (SPI Master, SPI Slave, I2C Master, I2S Master, I2S Slave) please refer to the FlexIO Application Information Section.

50.4.5 Interrupts and DMA Requests

The following table illustrates the status flags that can generate the FlexIO interrupt and DMA requests.

Table 50-6. FlexIO Interrupts and DMA Requests

Flag	Description	Interrupt	DMA Request	Low Power Wakeup
SSF	Shifter Status Flag.	Y	Y	Y
SEF	Shifter Error Flag.	Y	N	Y
TSF	Timer Status Flag.	Y	N	Y

50.4.6 Peripheral Triggers

The connection of the FlexIO peripheral triggers with other peripherals are device specific.

50.4.6.1 Output Triggers

Each FlexIO Timer generates an output trigger equal to the timer output. The output trigger is not affected by the timer pin polarity configuration.

50.4.6.2 Input Trigger

FlexIO supports multiple external trigger inputs that can be used to trigger one or more FlexIO timers. The external triggers are synchronized to the FlexIO functional clock and must assert for at least two cycles of the FlexIO functional clock to be sampled correctly.

50.5 Application Information

This section provides examples for a variety of FlexIO module applications.

50.5.1 UART Transmit

UART transmit can be supported using one Timer, one Shifter and one Pin (two Pins if supporting CTS). The start and stop bit insertion is handled automatically and multiple transfers can be supported using DMA controller. The timer status flag can be used to indicate when the stop bit of each word is transmitted.

Break and idle characters require software intervention, before transmitting a break or idle character the SSTART and SSTOP fields should be altered to transmit the required state and the data to transmit must equal 0xFF or 0x00. Supporting a second stop bit requires the stop bit to be inserted into the data stream using software (and increasing the number of bits to transmit). Note that when performing byte writes to SHIFTBUFn (or SHIFTBUFBIS for transmitting MSB first), the rest of the register remains unaltered allowing an address mark bit or additional stop bit to remain undisturbed.

FlexIO does not support automatic insertion of parity bits.

Table 50-7. UART Transmit Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0. Can invert output data by setting PINPOL, or can support open drain by setting PINPOL=0x1 and PINCFG=0x1.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0000_2222	Configure start bit, stop bit, enable on trigger asserted and disable on

Table continues on the next page...

Table 50-7. UART Transmit Configuration (continued)

Register	Value	Comments
		compare. Can support CTS by configuring TIMEN=0x3.
TIMCTLn	0x01C0_0001	Configure dual 8-bit counter using Shifter 0 status flag as inverted internal trigger source. Can support CTS by configuring PINSEL=0x1 (for Pin 1) and PINPOL=0x1.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF[7:0] to initiate an 8-bit transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS[7:0] register instead.

50.5.2 UART Receive

UART receive can be supported using one Timer, one Shifter and one Pin (two Timers and two Pins if supporting RTS). The start and stop bit verification is handled automatically and multiple transfers can be supported using the DMA controller. The timer status flag can be used to indicate when the stop bit of each word is received.

Triple voting of the received data is not supported by FlexIO, data is sampled only once in the middle of each bit. Another timer can be used to implement a glitch filter on the incoming data, another Timer can also be used to detect an idle line of programmable length. Break characters will cause the error flag to set and the shifter buffer register will return 0x00.

FlexIO does not support automatic verification of parity bits.

Table 50-8. UART Receiver Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.

Table continues on the next page...

Table 50-8. UART Receiver Configuration (continued)

Register	Value	Comments
TIMCFGn	0x0204_2422	Configure start bit, stop bit, enable on pin posedge and disable on compare. Enable resynchronization to received data with TIMOUT=0x2 and TIMRST=0x4.
TIMCTLn	0x0000_0081	Configure dual 8-bit counter using inverted Pin 0 input.
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

The UART Receiver with RTS configuration uses a 2nd Timer to generate the RTS output. The RTS will assert when the start bit is detected and negate when the data is read from the shifter buffer register. No start bit will be detected while the RTS is asserted, the received data is simply ignored.

Table 50-9. UART Receiver with RTS Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Configure start bit of 0 and stop bit of 1.
SHIFTCTLn	0x0080_0001	Configure receive using Timer 0 on negedge of clock with input data on Pin 0. Can invert input data by setting PINPOL.
TIMCMPn	0x0000_0F01	Configure 8-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0204_2522	Configure start bit, stop bit, enable on pin posedge with trigger asserted and disable on compare. Enable resynchronization to received data with TIMOUT=0x2 and TIMRST=0x4.
TIMCTLn	0x02C0_0081	Configure dual 8-bit counter using inverted Pin 0 input. Trigger is internal using inverted Pin 1 input.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0030_6100	Enable on Timer N enable and disable on trigger falling edge. Decrement on trigger to ensure no compare.
TIMCTL(n+1)	0x0143_0003	Configure 16-bit counter and output on Pin 1. Trigger is internal using Shifter 0 flag.

Table continues on the next page...

Table 50-9. UART Receiver with RTS Configuration (continued)

Register	Value	Comments
SHIFTBUFn	Data to receive	Received data can be read from SHIFTBUFBYS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS[7:0] register instead.

50.5.3 SPI Master

SPI master mode can be supported using two Timers, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The stop bit is used to guarantee a minimum of 1 clock cycle between the slave select negating and before the next transfer. Writing to the transmit buffer by either core or DMA is used to initiate each transfer.

Due to synchronization delays, the setup time for the serial input data is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 50-10. SPI Master (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on negedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on posedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as

Table continues on the next page...

Table 50-10. SPI Master (CPHA=0) Configuration (continued)

Register	Value	Comments
		the inverted trigger. Set PINPOL to invert the output shift clock.
TIMCMP(n+1)	0x0000_FFFF	Never compare.
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 50-11. SPI Master (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0021	Start bit loads data on first shift.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on negedge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0100_2222	Configure start bit, stop bit, enable on trigger high and disable on compare, initial clock state is logic 0.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (shift clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock. Set TIMDIS=3 to keep slave select asserted for as long as there is data in the transmit buffer.
TIMCMP(n+1)	0x0000_FFFF	Never compare.

Table continues on the next page...

Table 50-11. SPI Master (CPHA=1) Configuration (continued)

Register	Value	Comments
TIMCFG(n+1)	0x0000_1100	Enable when Timer 0 is enabled and disable when Timer 0 is disabled.
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter (never compare) using inverted Pin 3 output (as slave select).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

50.5.4 SPI Slave

SPI slave mode can be supported using one Timer, two Shifters and four Pins. Either CPHA=0 or CPHA=1 can be supported and transfers can be supported using the DMA controller. For CPHA=1, the select can remain asserted for multiple transfers and the timer status flag can be used to indicate the end of the transfer.

The transmit data must be written to the transmit buffer register before the external slave select asserts, otherwise the shifter error flag will be set.

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

Table 50-12. SPI Slave (CPHA=0) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.
SHIFTCTLn	0x0083_0002	Configure transmit using Timer 0 on falling edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0000_0101	Configure receive using Timer 0 on rising edge of shift clock with input data on Pin 1.

Table continues on the next page...

Table 50-12. SPI Slave (CPHA=0) Configuration (continued)

Register	Value	Comments
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0120_0600	Configure enable on trigger rising edge, initial clock state is logic 0 and decrement on pin input.
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

Table 50-13. SPI Slave (CPHA=1) Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Shifter configured to load on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_003F	Configure 32-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFGn	0x0120_6602	Configure start bit, enable on trigger rising edge, disable on trigger falling edge, initial clock state is logic 0 and decrement on pin input.
TIMCTLn	0x06C0_0203	Configure 16-bit counter using Pin 2 input (shift clock), with Pin 3 input (slave select) as the inverted trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUF, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support MSB first transfer by writing to SHIFTBUFBBS register instead.

Table continues on the next page...

Table 50-13. SPI Slave (CPHA=1) Configuration (continued)

Register	Value	Comments
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBYS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support MSB first transfer by reading from SHIFTBUFBIS register instead.

50.5.5 I2C Master

I2C master mode can be supported using two Timers, two Shifters and two Pins. One timer is used to generate the SCL output and one timer is used to control the shifters. The two shifters are used to transmit and receive for every word, when receiving the transmitter must transmit 0xFF to tristate the output. FlexIO inserts a stop bit after every word to generate/verify the ACK/NACK. FlexIO waits for the first write to the transmit data buffer before enabling SCL generation. Data transfers can be supported using the DMA controller and the shifter error flag will set on transmit underrun or receive overflow.

The first timer generates the bit clock for the entire packet (START to Repeated START/ STOP), so the compare register needs to be programmed with the total number of clock edges in the packet (minus one). The timer supports clock stretching using the reset counter when pin equal to output (although this increases both the clock high and clock low periods by at least 1 FlexIO clock cycle each). The second timer uses the SCL input pin to control the transmit/receive shift registers, this enforces an SDA data hold time by an extra 2 FlexIO clock cycles.

Both the transmit and receive shifters need to be serviced for each word in the transfer, the transmit shifter must transmit 0xFF when receiving and the receive shifter returns the data actually present on the SDA pin. The transmit shifter will load 1 additional word on the last falling edge of SCL pin, this word should be 0x00 if generating a STOP condition or 0xFF if generating a repeated START condition. During the last word of a master-receiver transfer, the transmit SSTOP bit should be set by software to generate a NACK.

The receive shift register will assert an error interrupt if a NACK is detected, but software is responsible for generating the STOP or repeated START condition. If a NACK is detected during master-transmit, the interrupt routine should immediately write the transmit shifter register with 0x00 (if generating STOP) or 0xFF (if generating repeated START). Software should then wait for the next rising edge on SCL and then disable both timers. The transmit shifter should then be disabled after waiting the setup delay for a repeated START or STOP condition.

Due to synchronization delays, the data valid time for the transmit output is 2 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The I2C master data valid is delayed 2 cycles because the clock output is passed through a synchronizer before clocking the transmit/receive shifter (to guarantee some SDA hold time). Since the SCL output is synchronous with FlexIO clock, the synchronization delay is 1 cycle and then 1 cycle to generate the output.

Table 50-14. I2C Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0032	Start bit enabled (logic 0) and stop bit enabled (logic 1).
SHIFTCTLn	0x0101_0082	Configure transmit using Timer 1 on rising edge of clock with inverted output enable (open drain output) on Pin 0.
SHIFTCFG(n+1)	0x0000_0020	Start bit disabled and stop bit enabled (logic 0) for ACK/NACK detection.
SHIFTCTL(n+1)	0x0180_0001	Configure receive using Timer 1 on falling edge of clock with input data on Pin 0.
TIMCMPn	0x0000_2501	Configure 2 word transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of words x 18) + 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0102_2222	Configure start bit, stop bit, enable on trigger high, disable on compare, reset if output equals pin. Initial clock state is logic 0 and is not affected by reset.
TIMCTLn	0x01C1_0101	Configure dual 8-bit counter using Pin 1 output enable (SCL open drain), with Shifter 0 flag as the inverted trigger.
TIMCMP(n+1)	0x0000_000F	Configure 8-bit transfer. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_1112	Enable when Timer 0 is enabled, disable when Timer 0 is disabled, enable start bit and stop bit at end of each word, decrement on pin input.
TIMCTL(n+1)	0x01C0_0183	Configure 16-bit counter using inverted Pin 1 input (SCL).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBBS[7:0], use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS[7:0], use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

50.5.6 I2S Master

I2S master mode can be supported using two Timers, two Shifters and four Pins. One timer is used to generate the bit clock and control the shifters and one timer is used to generate the frame sync. FlexIO waits for the first write to the transmit data buffer before enabling bit clock and frame sync generation. Data transfers can be supported using the DMA controller and the shifter error flag will set on transmit underrun or receive overflow.

The bit clock frequency is an even integer divide of the FlexIO clock frequency, and the initial frame sync assertion occurs at the same time as the first bit clock edge. The timer uses the start bit to ensure the frame sync is generated one clock cycle before the first output data.

Due to synchronization delays, the setup time for the receiver input is 1.5 FlexIO clock cycles, so the maximum baud rate is divide by 4 of the FlexIO clock frequency.

Table 50-15. I2S Master Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0001	Load transmit data on first shift and stop bit disabled.
SHIFTCTLn	0x0003_0002	Configure transmit using Timer 0 on rising edge of clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0080_0101	Configure receive using Timer 0 on falling edge of clock with input data on Pin 1.
TIMCMPn	0x0000_3F01	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of bits x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFGn	0x0000_0202	Configure start bit, enable on trigger high and never disable. Initial clock state is logic 1.
TIMCTLn	0x01C3_0201	Configure dual 8-bit counter using Pin 2 output (bit clock), with Shifter 0 flag as the inverted trigger. Set PINPOL to invert the output shift clock.
TIMCMP(n+1)	0x0000_007F	Configure 32-bit transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:0] = (number of bits x baud rate divider) - 1.
TIMCFG(n+1)	0x0000_0100	Enable when Timer 0 is enabled and never disable.

Table continues on the next page...

Table 50-15. I2S Master Configuration (continued)

Register	Value	Comments
TIMCTL(n+1)	0x0003_0383	Configure 16-bit counter using inverted Pin 3 output (as frame sync).
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

50.5.7 I2S Slave

I2S slave mode can be supported using three Timers, two Shifters and four Pins (for single transmit and single receive, other combinations of transmit and receive are possible).

The transmit data must be written to the transmit buffer register before the external frame sync asserts, otherwise the shifter error flag will be set.

Due to synchronization delays, the output valid time for the serial output data is 2.5 FlexIO clock cycles, so the maximum baud rate is divide by 6 of the FlexIO clock frequency.

The output valid time of I2S slave is max 2.5 cycles because there is a maximum 1.5 cycle delay on the clock synchronization plus 1 cycle to output the data

Timer 2 detects the falling edge of frame sync (start of new frame) and asserts output until falling edge of bit clock (when frame sync is normally sampled). Timer 0 detects falling edge of bit clock with Timer 2 output asserted and asserts output for length of frame. Timer 1 detects rising edge of bit clock with Timer 0 output asserted and controls shift registers for 32-bit transfers.

Table 50-16. I2S Slave Configuration

Register	Value	Comments
SHIFTCFGn	0x0000_0000	Start and stop bit disabled.

Table continues on the next page...

Table 50-16. I2S Slave Configuration (continued)

Register	Value	Comments
SHIFTCTLn	0x0103_0002	Configure transmit using Timer 1 on rising edge of shift clock with output data on Pin 0.
SHIFTCFG(n+1)	0x0000_0000	Start and stop bit disabled.
SHIFTCTL(n+1)	0x0180_0101	Configure receive using Timer 1 on falling edge of shift clock with input data on Pin 1.
TIMCMPn	0x0000_007F	Configure two 32-bit transfers per frame. Set TIMCMP[15:0] = (number of bits x 4) - 1.
TIMCFGn	0x0020_2500	Configure enable on pin rising edge (inverted bit clock) with trigger high (Timer 2) and disable on compare, initial clock state is logic 1 and decrement on pin input (bit clock).
TIMCTLn	0x0B40_0283	Configure 16-bit counter using inverted Pin 2 input (bit clock), with Timer 2 output as the trigger.
TIMCMP(n+1)	0x0000_003F	Configure 32-bit transfers. Set TIMCMP[15:0] = (number of bits x 2) - 1.
TIMCFG(n+1)	0x0020_2500	Configure enable on pin (bit clock) rising edge with trigger (Timer 0) high and disable on compare, initial clock state is logic 1 and decrement on pin input (bit clock).
TIMCTL(n+1)	0x0340_0283	Configure 16-bit counter using inverted Pin 2 input (bit clock), with Timer 0 output as the trigger.
TIMCMP(n+2)	0x0000_0000	Compare on zero (first edge).
TIMCFG(n+2)	0x0020_6400	Configure enable on inverted pin (frame sync) rising edge and disable on trigger falling edge (bit clock), initial clock state is logic 1 and decrement on inverted pin input (frame sync).
TIMCTL(n+2)	0x0440_0383	Configure 16-bit counter using inverted Pin 3 input (frame sync), with Pin 2 input (bit clock) as the trigger.
SHIFTBUFn	Data to transmit	Transmit data can be written to SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request. Can support LSB first transfer by writing to SHIFTBUF register instead.
SHIFTBUF(n+1)	Data to receive	Received data can be read from SHIFTBUFBIS, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request. Can support LSB first transfer by reading from SHIFTBUF register instead.

50.5.8 Camera Interface

Camera Interface can be supported using one Timer, one or more Shifters and multiple Pins. Multiple transfers can be supported using DMA controller.

The example below describes FlexIO configuration for interfacing to an 8-bit CMOS sensor with PCLK, VSYNC, HREF and D[7:0] outputs. The example uses a 128-bit buffer to capture 16-pixels of image data before interrupt or DMA transfer, however a bigger or smaller buffer may be used depending on system DMA performance and FlexIO resource usage by other applications. Note that additional timers may be used to track number of pixels per row and number of rows per frame or HREF/VSYNC may be assigned as GPIO interrupts for software tracking.

Table 50-17. Camera Interface Configuration for 8-bit CMOS sensor

Register	Value	Comments
SHIFTCFGn...n+2[1]	0x0007_0100	Configure 8-bit parallel shift in from adjacent shifter.
SHIFTCFGn+3	0x0007_0000	Configure 8-bit parallel shift in from pins FXIO_D[7:0] (D[7:0]).
SHIFTCTLn...n+3	0x0080_0001	Configure receive using Timer 0 on negedge of clock.
TIMCMPn	0x0000_001F	Configure 16-pixel (8 bits/pixel x 16 pixels = 128-bits) transfer. Set TIMCMP[15:0] = (number of pixels x 2) - 1.
TIMCFGn	0x0120_6600	Configure enable on trigger (HREF) rising edge and disable on trigger falling edge, initial Shift clock state is logic 0 and decrement on PCLK input.
TIMCTLn	0x12C0_0803	Configure 16-bit counter using FXIO_D[8] input (PCLK), with FXIO_D[9] input (HREF) as the inverted trigger.
SHIFTBUFn...n+3	Data to receive	Received data can be read from SHIFTBUFn...n+3, use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

1. n=0 or 4

50.5.9 Motorola 68K/Intel 8080 Bus Interface

The Motorola 68K and Intel 8080 bus are two parallel interfaces commonly used by Smart/Asynchronous LCD controllers. In conjunction with GPIO, FlexIO is able to drive these interfaces using one Timer and one Shifter, although additional Shifters could be used to support large transfers via the DMA controller.

The configuration below provides an example of how to drive a 16-bit 68K or 8080 bus. For a 8080 bus, two GPIO are used to drive the nCS and RS pins. For a 68K bus, an additional GPIO is required to drive the RDWR pin.

Table 50-18. Motorola 68K/Intel 8080 Write Configuration

Register	Value	Comments
SHIFTCFG0...7	0x000F_0100	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCTL0	0x0003_0002	Configure transmit using Timer 0 on posedge of clock with data output to FXIO_D[15:0].
SHIFTCTL1...7	0x0000_0002	Configure transmit using Timer 0 on posedge of clock.
TIMCMP0	0x0000_0101 (1-beat) 0x0000_1F01 (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_2200	Configure enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	0x01C3_1001 (Motorola 68K, 1-beat) 0x1DC3_1001 (Motorola 68K, 16-beats) 0x01C3_1081 (Intel 8080, 1-beat) 0x1DC3_1081 (Intel 8080, 16-beats)	Configure dual 8-bit counter using FXIO_D[16] output (EN pin for 68K, nWR pin for 8080), with Shifter 0 (1-beat) or Shifter 7 (16-beats) flag as the inverted trigger.
SHIFTBUF0...7	Data to transmit	Transmit data can be written to SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats) to initiate a transfer, use the Shifter Status Flag to indicate when data can be written using interrupt or DMA request.

Table 50-19. Motorola 68K/Intel 8080 Read Configuration

Register	Value	Comments
SHIFTCFG0...6	0x000F_0100	Configure 16-bit parallel shift in from adjacent shifter.
SHIFTCFG7	0x000F_0000	Configure 16-bit parallel shift in from pin.

Table continues on the next page...

Table 50-19. Motorola 68K/Intel 8080 Read Configuration (continued)

Register	Value	Comments
SHIFTCTL0...7	0x0080_0001	Configure receive using Timer 0 on negedge of clock with data input from FXIO_D[15:0].
TIMCMP0	0x0000_0101 (1-beat) 0x0000_1F01 (16-beats)	Configure 1 or 16-beat transfer with baud rate of divide by 4 of the FlexIO clock. Set TIMCMP[15:8] = (number of beats x 2) - 1. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_2220	Configure stop_bit, enable on trigger high and disable on compare. Timer output high on enable.
TIMCTL0	0x1DC3_1001 (Motorola 68K, 1 beat) 0x01C3_1001 (Motorola 68K, 16 beats) 0x1DC3_1181 (Intel 8080, 1 beat) 0x01C3_1181 (Intel 8080, 16 beats)	Configure dual 8-bit counter using either FXIO_D[16] output (EN pin for 68K) or FXIO_D[17] output (nRD pin for 8080), with Shifter 7 flag (1-beat) or Shifter 0 flag (16-beats) as the inverted trigger.
SHIFTBUF0...7	Data received	Received data can be read from SHIFTBUF0 (1-beat) or SHIFTBUF0...7 (16-beats), use the Shifter Status Flag to indicate when data can be read using interrupt or DMA request.

In general, any operation to a 68K/8080 bus slave will begin with a register write cycle followed by one or more data read or write cycles. To accomplish this, the following program flow should be used:

1. Configure FlexIO with 1-beat write configuration
2. Configure GPIO to assert nCS, RS pins (and deassert RDWR pin for 68K)
3. Write register index data to SHIFTBUF0[15:0]
4. Configure GPIO to deassert RS pin (and assert RDWR pin for 68K data read)
5. Configure FlexIO with desired read or write configuration (e.g. 1 or 16-beats)
6. Use the Shifter Status Flag to trigger interrupt or DMA driven data transfers to/from SHIFTBUF registers
7. Configure GPIO to deassert nCS pin

50.5.10 Low Power State Machine

The configuration below details a hypothetical state machine example to illustrate the flexibility allowed when using Shifter state mode.

In this example, FlexIO waits for the FXIO_D[2] pin to assert and then drives a complementary square wave output at a frequency of FLEXIO_CLK/131072 on the FXIO_D[1:0] pins while the comparator output is asserted (This assumes comparator is

connected to external trigger 15. See the chip-specific FlexIO information for actual FlexIO trigger mappings). Throughout this operation, the CPU can be kept in a STOP/ VLPS mode, by clearing the CTRL[DOZEN] bit and ensuring the FLEXIO_CLK is enabled. The state diagram below shows the states and transitions implemented by this example.

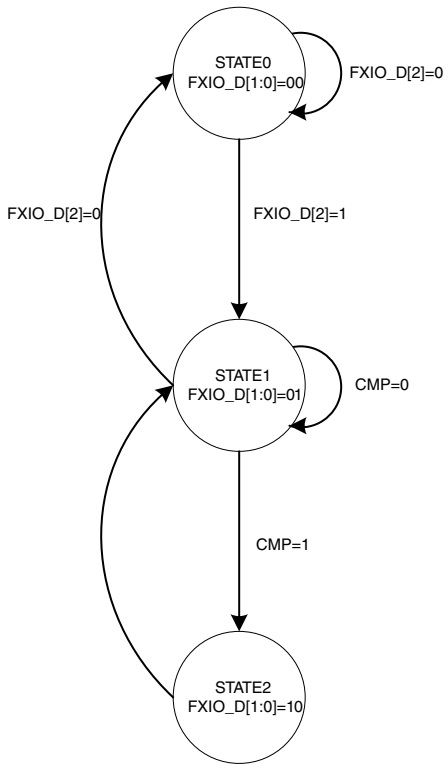


Figure 50-5. State Diagram

Table 50-20. State Machine Configuration

Register	Value	Comments
SHIFTCFG0...2	0x0000_0003	Enable FXIO_D[1:0] as outputs.
SHIFTCTL0	0x0080_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF0	0x0020_8208	State0: Drive FXIO_D[1:0]=00, transition to State0 if FXIO_D[2]=0, State1 if FXIO_D[2]=1.
SHIFTCTL1	0x0000_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output high to trigger next state.

Table continues on the next page...

Table 50-20. State Machine Configuration (continued)

Register	Value	Comments
SHIFTBUF1	0x0140_8408	State1: Drive FXIO_D[1:0]=01, transition to State0 if FXIO_D[2]=0, State1 if CMP=0, State2 if CMP=1 (FXIO_D[3]=1)
SHIFTCTL2	0x0080_0206	Configure for State mode using FXIO_D[4:2] as inputs to select next state and Timer0 output low to trigger next state.
SHIFTBUF2	0x0224_9249	State2: Drive FXIO_D[1:0]=10, transition to State1 when Timer0 output low
TIMCMP0	0x0000_FFFF	Configure baud rate of divide by 131072 of the FlexIO clock. Set TIMCMP[7:0] = (baud rate divider / 2) - 1.
TIMCFG0	0x0000_0000	Configure timer always enabled.
TIMCTL0	0x0000_0003	Configure single 16-bit counter.
TIMCFG1	0x0010_7600	Configure timer enabled on trigger rising edge, disabled on trigger falling edge, decrement on trigger edge.
TIMCTL1	0x0F03_0303	Configure timer output enabled on FXIO_D[3], external trigger 15 (comparator output) selected.

Chapter 51
Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

51.1 Chip-specific LPUART information

Table 51-1. Reference links to related information

Topic	Related module	Reference
Full description	LPUART	LPUART
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

51.1.1 LPUART

The LPUART provides asynchronous, serial communication capability with external devices.

- Supports non-return-to-zero (NRZ) encoding format and IrDA-compatible infrared (low-speed) SIR format
- Can continue operating while the chip is in stop modes, if an appropriate clock is available
- Designed for low CPU overhead, with DMA offloading of FIFO register accesses

Table 51-2. LPUART configuration

Parameter	Description
Name	LPUART
Instances	8
Configurable features	• LPUART0-1: 4-word TX FIFO and RX FIFO • LPUART2-7: 8-word TX FIFO and RX FIFO • LPUART 0-7: Hardware Flow Control, IrDA

Table continues on the next page...

Table 51-2. LPUART configuration (continued)

Parameter	Description
Interface speed	LPUART0-7: maximum 12 Mbps
External I/O pins	LPUART0-7: TX, RX, CTS_b, RTS_b. See the attached IOMUXC spreadsheet for pinout details.

51.2 Introduction

51.2.1 Features

Features of the LPUART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Programmable baud rates (13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- Transmit and receive baud rate can operate asynchronous to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency
  - Supports operation in Stop modes
- Interrupt, DMA or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
  - Receive data match
- Hardware parity generation and checking
- Programmable 7-bit, 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Three receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
  - Receive data match
- Automatic address matching to reduce ISR overhead:
  - Address mark matching
  - Idle line address matching
  - Address match start, address match end
- Optional 13-bit break character generation / 11-bit break character detection

- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64 or 128 idle characters
- Selectable transmitter output and receiver input polarity
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse width
- Independent FIFO structure for transmit and receive
  - Separate configurable watermark for receive and transmit requests
  - Option for receiver to assert request after a configurable number of idle characters if receive FIFO is not empty

51.2.2 Modes of operation

51.2.2.1 Stop mode

The LPUART will remain functional during Stop mode, provided the CTRL[DOZEEN] bit is clear and the asynchronous transmit and receive clock remain enabled. The LPUART can generate an interrupt or DMA request to cause a wakeup from Stop mode.

If the LPUART is disabled in Stop mode, then it can generate a wakeup via the STAT[RXEDGIF] flag if the receiver detects an active edge.

51.2.2.2 Wait mode

The LPUART can be configured to Stop in Wait modes, when the CTRL[DOZEEN] bit is set. The transmitter and receiver will finish transmitting/receiving the current word.

51.2.2.3 Debug mode

The LPUART remains functional in debug mode.

51.2.3 Signal Descriptions

Signal	Description	I/O
TXD	Transmit data. This pin is normally an output, but is an input (tristated) in single wire mode whenever the transmitter is	I/O

Table continues on the next page...

Signal	Description	I/O
	disabled or transmit direction is configured for receive data.	
RXD	Receive data.	I
CTS_B	Clear to send.	I
RTS_B	Request to send.	O

51.2.4 Block diagram

The following figure shows the transmitter portion of the LPUART.

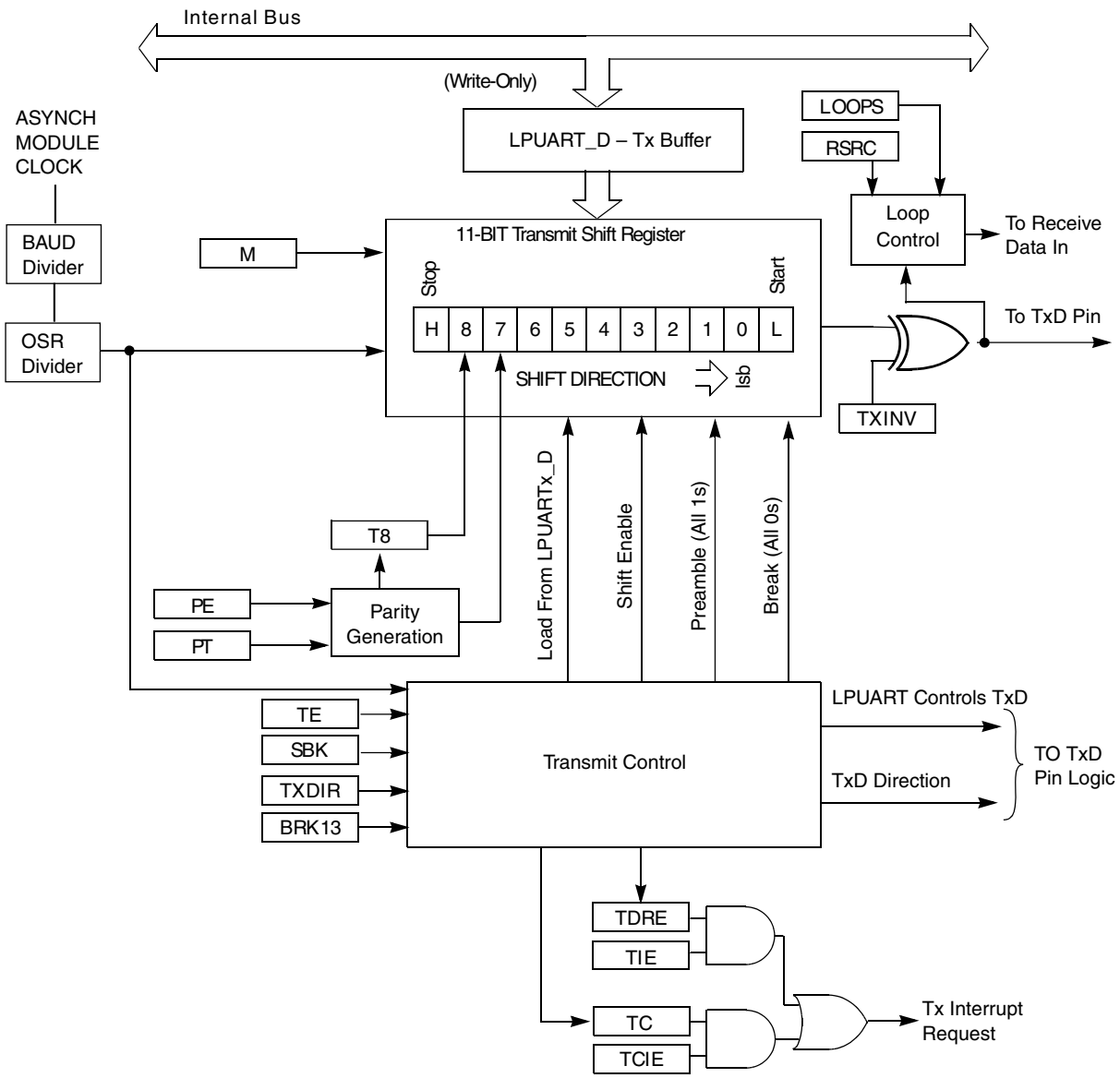


Figure 51-1. LPUART transmitter block diagram

The following figure shows the receiver portion of the LPUART.

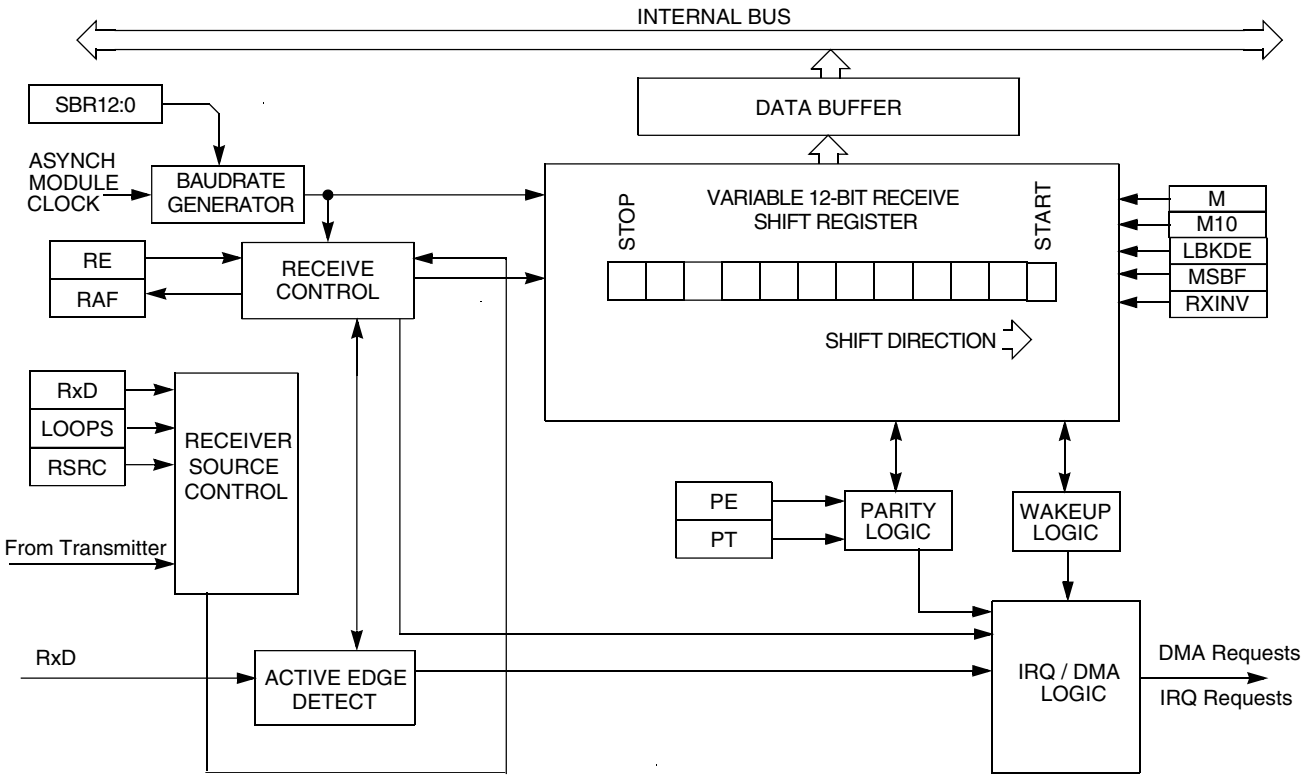


Figure 51-2. LPUART receiver block diagram

51.3 Register definition

The LPUART includes registers to control baud rate, select options, report status, and store transmit/receive data. Access to an address outside the valid memory map will generate a bus error.

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. This module will not check if programmed values in the registers are correct; the application software must ensure that valid programmed values are being written.

51.3.1 LPUART register descriptions

51.3.1.1 LPUART Memory map

LPUART0 base address: 4103_A000h

LPUART1 base address: 4103_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0401_0003h
4h	Parameter Register (PARAM)	32	RO	0000_0202h
8h	LPUART Global Register (GLOBAL)	32	RW	0000_0000h
Ch	LPUART Pin Configuration Register (PINCFG)	32	RW	0000_0000h
10h	LPUART Baud Rate Register (BAUD)	32	RW	0F00_0004h
14h	LPUART Status Register (STAT)	32	RW	00C0_0000h
18h	LPUART Control Register (CTRL)	32	RW	0000_0000h
1Ch	LPUART Data Register (DATA)	32	RW	0000_1000h
20h	LPUART Match Address Register (MATCH)	32	RW	0000_0000h
24h	LPUART Modem IrDA Register (MODIR)	32	RW	0000_0000h
28h	LPUART FIFO Register (FIFO)	32	RW	00C0_0011h
2Ch	LPUART Watermark Register (WATER)	32	RW	0000_0000h

51.3.1.2 Version ID Register (VERID)

51.3.1.2.1 Offset

Register	Offset
VERID	0h

51.3.1.2.2 Function

The Version ID register indicates the version integrated for this instance on the device and also indicates inclusion/exclusion of several optional features.

51.3.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

51.3.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number This read only field returns the feature set number. 0000000000000001b - Standard feature set. 0000000000000011b - Standard feature set with MODEM/IrDA support.

51.3.1.3 Parameter Register (PARAM)

51.3.1.3.1 Offset

Register	Offset
PARAM	4h

51.3.1.3.2 Function

The Parameter register indicates the parameter configuration for this instance on the device

51.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

51.3.1.3.4 Fields

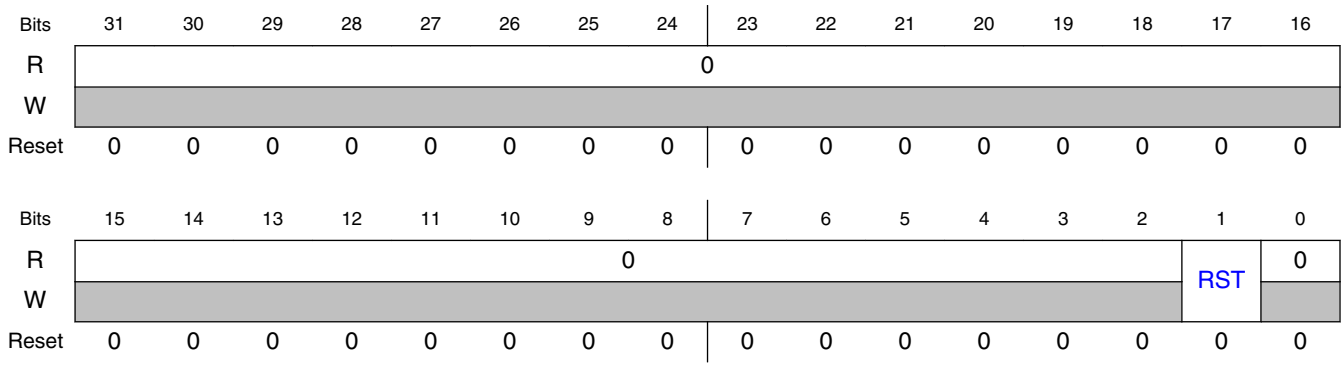
Field	Function
31-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size The number of words in the receive FIFO is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size The number of words in the transmit FIFO is 2^TXFIFO.

51.3.1.4 LPUART Global Register (GLOBAL)

51.3.1.4.1 Offset

Register	Offset
GLOBAL	8h

51.3.1.4.2 Diagram



51.3.1.4.3 Fields

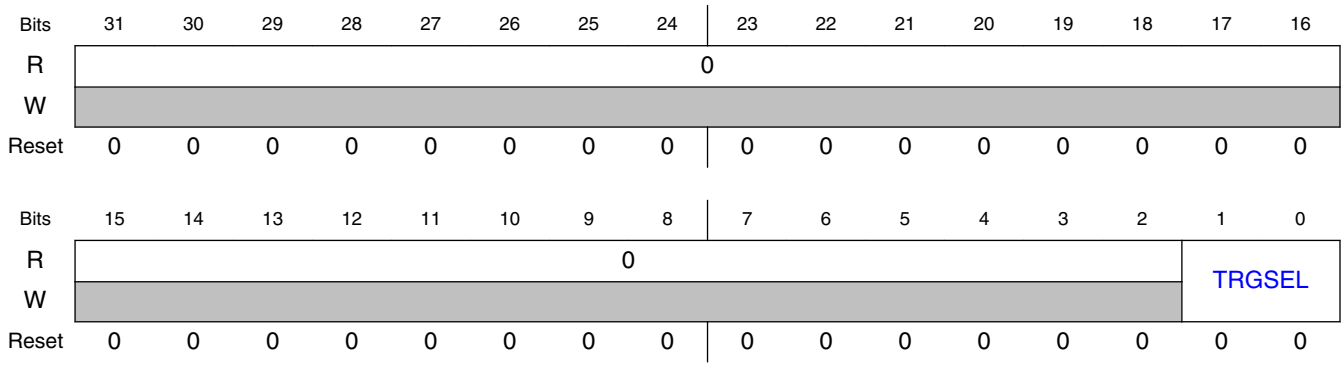
Field	Function
31-2 —	Reserved
1 RST	Software Reset Resets all internal logic and registers, except the Global Register. Remains set until cleared by software. 0b - Module is not reset. 1b - Module is reset.
0 —	Reserved

51.3.1.5 LPUART Pin Configuration Register (PINCFG)

51.3.1.5.1 Offset

Register	Offset
PINCFG	Ch

51.3.1.5.2 Diagram



51.3.1.5.3 Fields

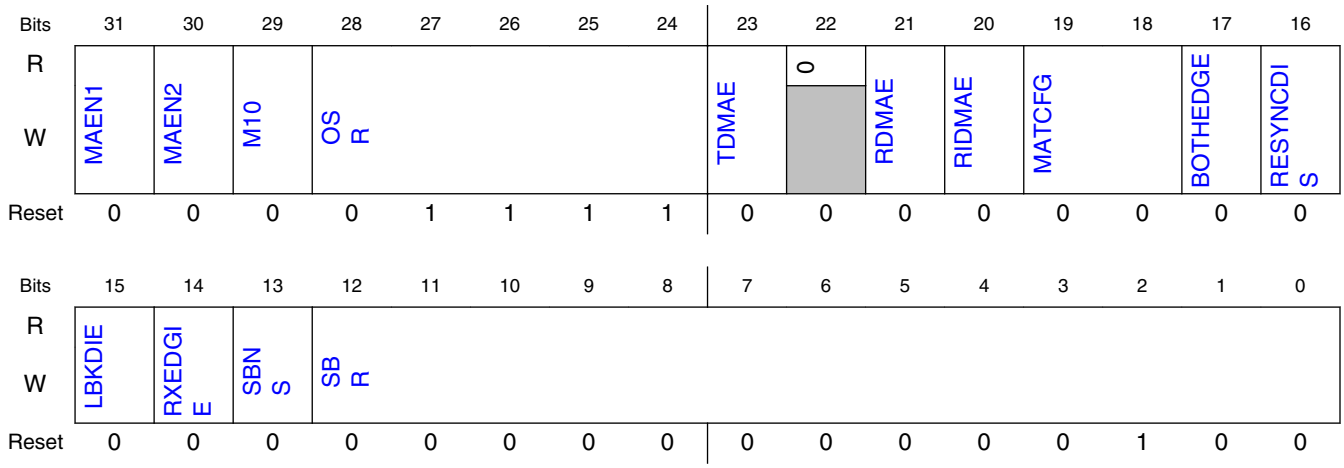
Field	Function
31-2 —	Reserved
1-0 TRGSEL	Trigger Select Configures the input trigger usage. This field should only be changed when the transmitter and receiver are both disabled. 00b - Input trigger is disabled. 01b - Input trigger is used instead of RXD pin input. 10b - Input trigger is used instead of CTS_B pin input. 11b - Input trigger is used to modulate the TXD pin output. The TXD pin output (after TXINV configuration) is ANDed with the input trigger.

51.3.1.6 LPUART Baud Rate Register (BAUD)

51.3.1.6.1 Offset

Register	Offset
BAUD	10h

51.3.1.6.2 Diagram



51.3.1.6.3 Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 0b - Normal operation. 1b - Enables automatic address matching or data matching mode for MATCH[MA1].
30 MAEN2	Match Address Mode Enable 2 0b - Normal operation. 1b - Enables automatic address matching or data matching mode for MATCH[MA2].
29 M10	10-bit Mode select The M10 bit causes a tenth bit to be part of the serial transmission. This bit should only be changed when the transmitter and receiver are both disabled. 0b - Receiver and transmitter use 7-bit to 9-bit data characters. 1b - Receiver and transmitter use 10-bit data characters.
28-24 OSR	Oversampling Ratio This field configures the oversampling ratio for the receiver. This field should only be changed when the transmitter and receiver are both disabled. 00000b - Writing 0 to this field will result in an oversampling ratio of 16 00001b - Reserved 00010b - Reserved 00011b - Oversampling ratio of 4, requires BOTHEDGE to be set. 00100b - Oversampling ratio of 5, requires BOTHEDGE to be set. 00101b - Oversampling ratio of 6, requires BOTHEDGE to be set. 00110b - Oversampling ratio of 7, requires BOTHEDGE to be set. 00111b - Oversampling ratio of 8. 01000b - Oversampling ratio of 9. 01001b - Oversampling ratio of 10. 01010b - Oversampling ratio of 11. 01011b - Oversampling ratio of 12. 01100b - Oversampling ratio of 13. 01101b - Oversampling ratio of 14. 01110b - Oversampling ratio of 15. 01111b - Oversampling ratio of 16. 10000b - Oversampling ratio of 17.

Table continues on the next page...

Field	Function
	10001b - Oversampling ratio of 18. 10010b - Oversampling ratio of 19. 10011b - Oversampling ratio of 20. 10100b - Oversampling ratio of 21. 10101b - Oversampling ratio of 22. 10110b - Oversampling ratio of 23. 10111b - Oversampling ratio of 24. 11000b - Oversampling ratio of 25. 11001b - Oversampling ratio of 26. 11010b - Oversampling ratio of 27. 11011b - Oversampling ratio of 28. 11100b - Oversampling ratio of 29. 11101b - Oversampling ratio of 30. 11110b - Oversampling ratio of 31. 11111b - Oversampling ratio of 32.
23 TDMAE	Transmitter DMA Enable TDMAE configures the transmit data register empty flag, STAT[TDRE], to generate a DMA request. 0b - DMA request disabled. 1b - DMA request enabled.
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable RDMAE configures the receiver data register full flag, STAT[RDRF], to generate a DMA request. 0b - DMA request disabled. 1b - DMA request enabled.
20 RIDMAE	Receiver Idle DMA Enable RIDMAE configures the receiver idle flag, STAT[IDLE], to generate a DMA request. When this bit is set, reading the DATA register when either DATA[RXEMPT] or DATA[IDLINE] bit is set, will generate an End Of Packet response until the completion of the existing DMA transfer. During an End of Packet response, reading the DATA register will return 0x0000_33FF and does not pull data from the FIFO. 0b - DMA request disabled. 1b - DMA request enabled.
19-18 MATCFG	Match Configuration Configures the match addressing mode used. This field should only be changed when the transmitter and receiver are both disabled. 00b - Address Match Wakeup 01b - Idle Match Wakeup 10b - Match On and Match Off 11b - Enables RWU on Data Match and Match On/Off for transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given oversampling ratio. This bit must be set for oversampling ratios between x4 and x7 and is optional for higher oversampling ratios. This bit should only be changed when the receiver is disabled. 0b - Receiver samples input data using the rising edge of the baud rate clock. 1b - Receiver samples input data using the rising and falling edge of the baud rate clock.
16 RESYNCDIS	Resynchronization Disable When set, disables the resynchronization of the received data word when a data one followed by data zero transition is detected. This bit should only be changed when the receiver is disabled. 0b - Resynchronization during received data word is supported 1b - Resynchronization during received data word is disabled

Table continues on the next page...

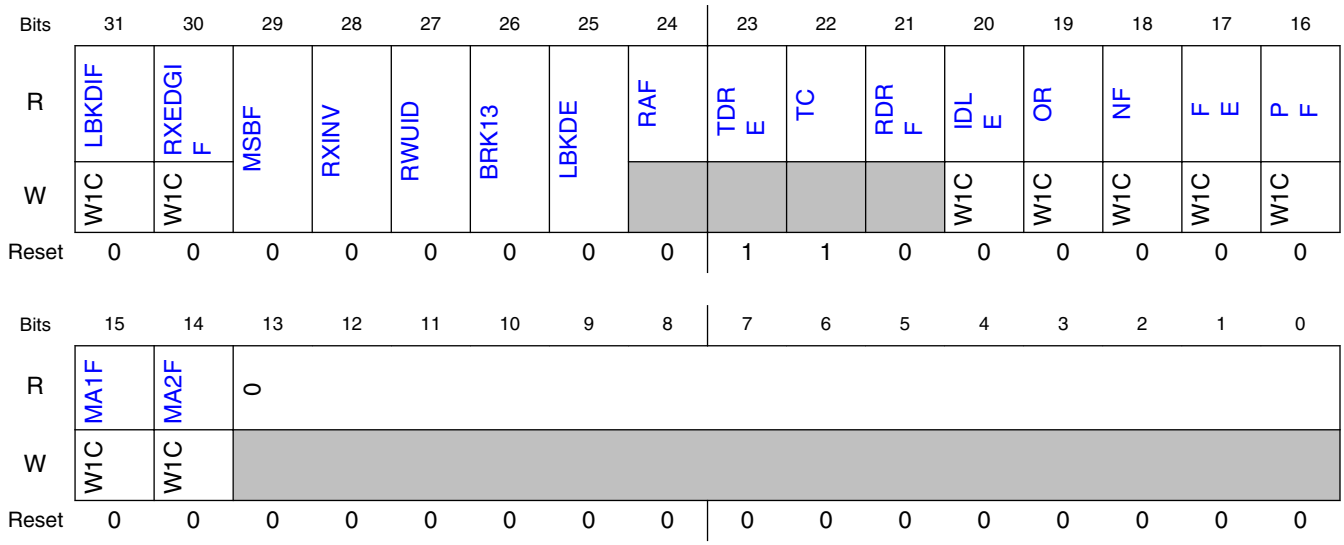
Field	Function
15 LBKDIE	LIN Break Detect Interrupt Enable LBKDIE enables the LIN break detect flag, STAT[LBKDIF], to generate interrupt requests. 0b - Hardware interrupts from STAT[LBKDIF] flag are disabled (use polling). 1b - Hardware interrupt requested when STAT[LBKDIF] flag is 1.
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables the receive input active edge, STAT[RXEDGIF], to generate interrupt requests. Changing CTRL[LOOP] or CTRL[RSRC] when RXEDGIE is set can cause the STAT[RXEDGIF] flag to set. 0b - Hardware interrupts from STAT[RXEDGIF] are disabled. 1b - Hardware interrupt is requested when STAT[RXEDGIF] flag is 1.
13 SBNS	Stop Bit Number Select SBNS determines whether data characters have one or two stop bits. This bit should only be changed when the transmitter and receiver are both disabled. 0b - One stop bit. 1b - Two stop bits.
12-0 SBR	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) × SBR)". The 13-bit baud rate setting [SBR12:SBR0] must be updated only when the transmitter and receiver are both disabled (CTRL[RE] and CTRL[TE] are both 0).

51.3.1.7 LPUART Status Register (STAT)

51.3.1.7.1 Offset

Register	Offset
STAT	14h

51.3.1.7.2 Diagram



51.3.1.7.3 Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it. 0b - No LIN break character has been detected. 1b - LIN break character has been detected.
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag RXEDGIF is set whenever the receiver is enabled and an active edge, falling if RXINV = 0, rising if RXINV=1, on the RXD pin occurs. RXEDGIF is cleared by writing a 1 to it. 0b - No active edge on the receive pin has occurred. 1b - An active edge on the receive pin has occurred.
29 MSBF	MSB First Setting this bit reverses the order of the bits that are transmitted and received on the wire. This bit does not affect the polarity of the bits, the location of the parity bit or the location of the start or stop bits. This bit should only be changed when the transmitter and receiver are both disabled. 0b - LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1b - MSB (bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of CTRL[M], CTRL[PE] and BAUD[M10]. Further, the first bit received after the start bit is identified as bit9, bit8, bit7 or bit6 depending on the setting of CTRL[M] and CTRL[PE].
28 RXINV	Receive Data Inversion Setting this bit reverses the polarity of the received data input. This bit should only be changed when the receiver is disabled. **NOTE:** Setting RXINV inverts the RXD input for all cases: data bits, start and stop bits, break, and idle. 0b - Receive data not inverted. 1b - Receive data inverted.
27	Receive Wake Up Idle Detect

Table continues on the next page...

Field	Function
RWUID	For RWU on idle character, RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. For address match wakeup, RWUID controls if the IDLE bit is set when the address does not match. This bit should only be changed when the receiver is disabled. 0b - During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. During address match wakeup, the IDLE bit does not set when an address does not match. 1b - During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. During address match wakeup, the IDLE bit does set when an address does not match.
26 BRK13	Break Character Generation Length BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. This bit should only be changed when the transmitter is disabled. A break character can be sent by setting CTRL[SBK] or by writing the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] clear. 0b - Break character is transmitted with length of 9 to 13 bit times. 1b - Break character is transmitted with length of 12 to 15 bit times.
25 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive data buffer. 0b - LIN break detect is disabled, normal break character can be detected. 1b - LIN break detect is enabled. LIN break character is detected at length of 11 bit times (if M = 0) or 12 (if M = 1) or 13 (M10 = 1).
24 RAF	Receiver Active Flag RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. 0b - LPUART receiver idle waiting for a start bit. 1b - LPUART receiver active (RXD input not idle).
23 TDRE	Transmit Data Register Empty Flag When the transmit FIFO is enabled, TDRE will set when the number of datawords in the transmit FIFO (DATA register) is equal to or less than the number indicated by WATER[TXWATER]). To clear TDRE, write to the DATA register until the number of words in the transmit FIFO is greater than the number indicated by WATER[TXWATER]. When the transmit FIFO is disabled,TDRE will set when the transmit DATA register is empty. To clear TDRE, write to the DATA register. TDRE is not affected by a character that is in the process of being transmitted; it is updated at the start of each transmitted character. 0b - Transmit data buffer full. 1b - Transmit data buffer empty.
22 TC	Transmission Complete Flag TC is cleared when there is a transmission in progress or when a preamble or break character is loaded. TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by writing to the DATA register to transmit new data, queuing a preamble by clearing and then setting CTRL[TE], queuing a break character by writing 1 to CTRL[SBK]. 0b - Transmitter active (sending data, a preamble, or a break). 1b - Transmitter idle (transmission activity complete).
21 RDRF	Receive Data Register Full Flag When the receive FIFO is enabled, RDRF is set when the number of datawords in the receive buffer is greater than the number indicated by WATER[RXWATER]. To clear RDRF, read the DATA register until the number of datawords in the receive data buffer is equal to or less than the number indicated by WATER[RXWATER]. When the receive FIFO is disabled, RDRF is set when the receive buffer (the DATA register) is full. To clear RDRF, read the DATA register.

Table continues on the next page...

Field	Function
	A character that is in the process of being received does not cause a change in RDRF until the entire character is received. Even if RDRF is set, the character will continue to be received until an overrun condition occurs once the entire character is received. 0b - Receive data buffer empty. 1b - Receive data buffer full.
20 IDLE	Idle Line Flag IDLE is set when the LPUART receive line becomes idle for a full character time after a period of activity. When CTRL[ILT] is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. When CTRL[ILT] is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line. To clear IDLE, write logic 1 to the IDLE flag. After IDLE has been cleared, it cannot be set again until after a new character has been stored in the receive buffer or a LIN break character has set the LBKDIF flag . IDLE is set only once even if the receive line remains idle for an extended period. 0b - No idle line detected. 1b - Idle line was detected.
19 OR	Receiver Overrun Flag OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If LBKDE is enabled and a LIN Break is detected, the OR field asserts if LBKDIF is not cleared before the next data character is received. While the OR flag is set, no additional data is stored in the data buffer even if sufficient room exists. To clear OR, write logic 1 to the OR flag. 0b - No overrun. 1b - Receive overrun (new LPUART data lost).
18 NF	Noise Flag The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame then noise is detected for that character. NF is set whenever the next character to be read from the DATA register was received with noise detected within the character. To clear NF, write logic 1 to the NF field. 0b - No noise detected. 1b - Noise detected in the received character in the DATA register.
17 FE	Framing Error Flag FE is set whenever the next character to be read from the DATA register was received with logic 0 detected where a stop bit was expected. To clear FE, write logic 1 to the FE field. 0b - No framing error detected. This does not guarantee the framing is correct. 1b - Framing error.
16 PF	Parity Error Flag PF is set whenever the next character to be read from the DATA register was received when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, write a logic 1 to the PF field. 0b - No parity error. 1b - Parity error.
15 MA1F	Match 1 Flag

Table continues on the next page...

Field	Function
	MA1F is set whenever the next character to be read from the DATA register matches MA1. To clear MA1F, write a logic 1 to the MA1F field. 0b - Received data is not equal to MA1 1b - Received data is equal to MA1
14 MA2F	Match 2 Flag MA2F is set whenever the next character to be read from the DATA register matches MA2. To clear MA2F, write a logic 1 to the MA2F field. 0b - Received data is not equal to MA2 1b - Received data is equal to MA2
13-0 —	Reserved

51.3.1.8 LPUART Control Register (CTRL)

51.3.1.8.1 Offset

Register	Offset
CTRL	18h

51.3.1.8.2 Function

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

51.3.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MA1IE	MA2IE	0		M7	IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

51.3.1.8.4 Fields

Field	Function
31 R8T9	Receive Bit 8 / Transmit Bit 9 R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading the DATA register. T9 is the tenth data bit transmitted when the LPUART is configured for 10-bit data formats. When writing 10-bit data, write T9 before writing the DATA register. If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time the DATA register is written. **NOTE:** R8 is a read-only bit and T9 is a write-only bit, the value read is different from the value written.
30 R9T8	Receive Bit 9 / Transmit Bit 8 R9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When reading 10-bit data, read R9 before reading the DATA register T8 is the ninth data bit transmitted when the LPUART is configured for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing the DATA register. If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, then it need not be written each time the DATA register is written. **NOTE:** R9 is a read-only bit and T8 is a write-only bit, the value read is different from the value written.
29 TXDIR	TXD Pin Direction in Single-Wire Mode When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the TXD pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the TXD pin. 0b - TXD pin is an input in single-wire mode. 1b - TXD pin is an output in single-wire mode.
28 TXINV	Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output. **NOTE:** Setting TXINV inverts the TXD output for all cases: data bits, start and stop bits, break, and idle. 0b - Transmit data not inverted. 1b - Transmit data inverted.
27 ORIE	Overrun Interrupt Enable This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0b - OR interrupts disabled; use polling. 1b - Hardware interrupt requested when OR is set.
26 NEIE	Noise Error Interrupt Enable This bit enables the noise flag (NF) to generate hardware interrupt requests. 0b - NF interrupts disabled; use polling. 1b - Hardware interrupt requested when NF is set.
25 FEIE	Framing Error Interrupt Enable This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0b - FE interrupts disabled; use polling. 1b - Hardware interrupt requested when FE is set.
24 PEIE	Parity Error Interrupt Enable This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0b - PF interrupts disabled; use polling). 1b - Hardware interrupt requested when PF is set.
23	Transmit Interrupt Enable

Table continues on the next page...

Field	Function
TIE	Enables STAT[TDRE] to generate interrupt requests. 0b - Hardware interrupts from TDRE disabled; use polling. 1b - Hardware interrupt requested when TDRE flag is 1.
22 TCIE	Transmission Complete Interrupt Enable for TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0b - Hardware interrupts from TC disabled; use polling. 1b - Hardware interrupt requested when TC flag is 1.
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate interrupt requests. 0b - Hardware interrupts from RDRF disabled; use polling. 1b - Hardware interrupt requested when RDRF flag is 1.
20 ILIE	Idle Line Interrupt Enable ILIE enables the idle line flag, STAT[IDLE], to generate interrupt requests. 0b - Hardware interrupts from IDLE disabled; use polling. 1b - Hardware interrupt requested when IDLE flag is 1.
19 TE	Transmitter Enable Enables the LPUART transmitter. TE can also be used to queue an idle preamble by clearing and then setting TE. When TE is cleared, this register bit will read as 1 until the transmitter has completed the current character and the TXD pin is tristated. A single idle character can also be queued by writing to the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] set. 0b - Transmitter disabled. 1b - Transmitter enabled.
18 RE	Receiver Enable Enables the LPUART receiver. When RE is written to 0, this register bit will read as 1 until the receiver finishes receiving the current character (if any). 0b - Receiver disabled. 1b - Receiver enabled.
17 RWU	Receiver Wakeup Control This field can be set to place the LPUART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when CTRL[WAKE] is clear or an address match when CTRL[WAKE] is set with STAT[RWUID] is clear. **NOTE:** RWU must be set only with CTRL[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by STAT[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the LPUART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to be reasserted. 0b - Normal receiver operation. 1b - LPUART receiver in standby waiting for wakeup condition.
16 SBK	Send Break Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 9 to 13 bits, or 12 to 15 bits if STAT[BRK13] is set, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the character currently being transmitted, a second break character may be queued before software clears SBK. A single break character can also be queued by writing to the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] clear. 0b - Normal transmitter operation. 1b - Queue break character(s) to be sent.

Table continues on the next page...

Register definition

Field	Function
15 MA1IE	Match 1 Interrupt Enable 0b - MA1F interrupt disabled 1b - MA1F interrupt enabled
14 MA2IE	Match 2 Interrupt Enable 0b - MA2F interrupt disabled 1b - MA2F interrupt enabled
13-12 —	Reserved
11 M7	7-Bit Mode Select This bit should only be changed when the transmitter and receiver are both disabled. 0b - Receiver and transmitter use 8-bit to 10-bit data characters. 1b - Receiver and transmitter use 7-bit data characters.
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before the IDLE flag is set. 000b - 1 idle character 001b - 2 idle characters 010b - 4 idle characters 011b - 8 idle characters 100b - 16 idle characters 101b - 32 idle characters 110b - 64 idle characters 111b - 128 idle characters
7 LOOPS	Loop Mode Select When LOOPS is set, the RXD pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0b - Normal operation - RXD and TXD use separate pins. 1b - Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input (see RSRC bit).
6 DOZEEN	Doze Enable 0b - LPUART is enabled in Doze mode. 1b - LPUART is disabled in Doze mode.
5 RSRC	Receiver Source Select This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0b - Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the LPUART does not use the RXD pin. 1b - Single-wire LPUART mode where the TXD pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 0b - Receiver and transmitter use 8-bit data characters. 1b - Receiver and transmitter use 9-bit data characters.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the LPUART when RWU=1: • Address mark in the bit preceding the stop bit (or bit preceding the parity bit when parity is enabled) of the received data character, or • An idle condition on the receive pin input signal. 0b - Configures RWU for idle-line wakeup. 1b - Configures RWU with address-mark wakeup.

Table continues on the next page...

Field	Function
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE:** In case the LPUART is programmed with ILT = 1, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0b - Idle character bit count starts after start bit. 1b - Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables hardware parity generation and checking. When parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0b - No hardware parity generation or checking. 1b - Parity enabled.
0 PT	Parity Type Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even. 0b - Even parity. 1b - Odd parity.

51.3.1.9 LPUART Data Register (DATA)

51.3.1.9.1 Offset

Register	Offset
DATA	1Ch

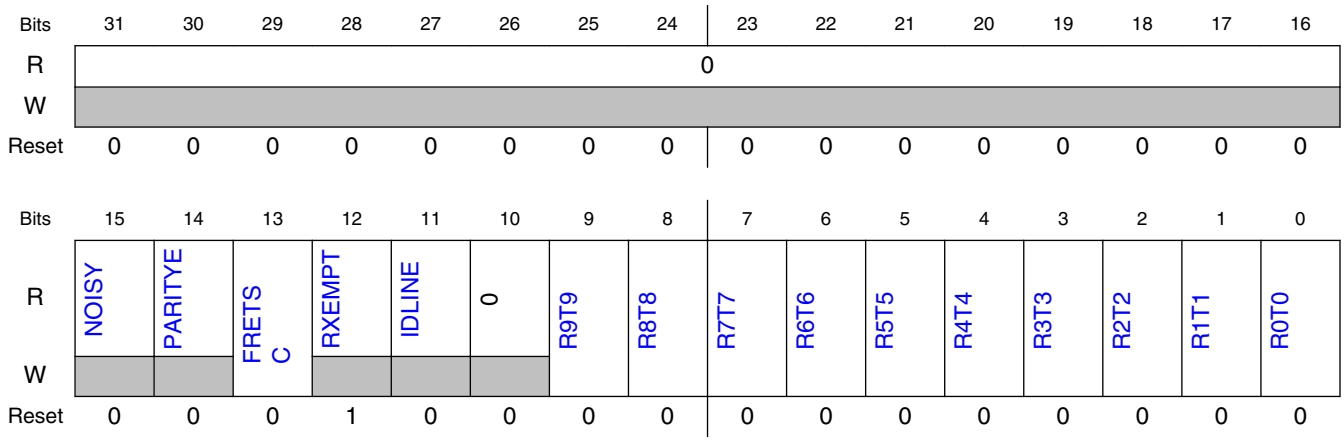
51.3.1.9.2 Function

NOTE

This register is two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer.

Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status flags.

51.3.1.9.3 Diagram



51.3.1.9.4 Fields

Field	Function
31-16 —	Reserved
15 NOISY	NOISY The current received dataword contained in DATA[R9:R0] was received with noise. 0b - The dataword was received without noise. 1b - The data was received with noise.
14 PARITYE	PARITYE The current received dataword contained in DATA[R9:R0] was received with a parity error. 0b - The dataword was received without a parity error. 1b - The dataword was received with a parity error.
13 FRETSC	Frame Error / Transmit Special Character For reads, indicates the current received dataword contained in DATA[R9:R0] was received with a frame error. For writes, indicates a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 is used to indicate a break character when 0 and a idle character when 1, the contents of DATA[T8:T0] should be zero. 0b - The dataword was received without a frame error on read, or transmit a normal character on write. 1b - The dataword was received with a frame error, or transmit an idle or break character on transmit.
12 RXEMPT	Receive Buffer Empty Asserts when there is no data in the receive buffer. This field does not take into account data that is in the receive shift register. 0b - Receive buffer contains valid data. 1b - Receive buffer is empty, data returned on read is not valid.
11 IDLINE	Idle Line Indicates the receiver line was idle before receiving the character in DATA[9:0]. Unlike the IDLE flag, this bit can set for the first character received when the receiver is first enabled. 0b - Receiver was not idle before receiving this character. 1b - Receiver was idle before receiving this character.

Table continues on the next page...

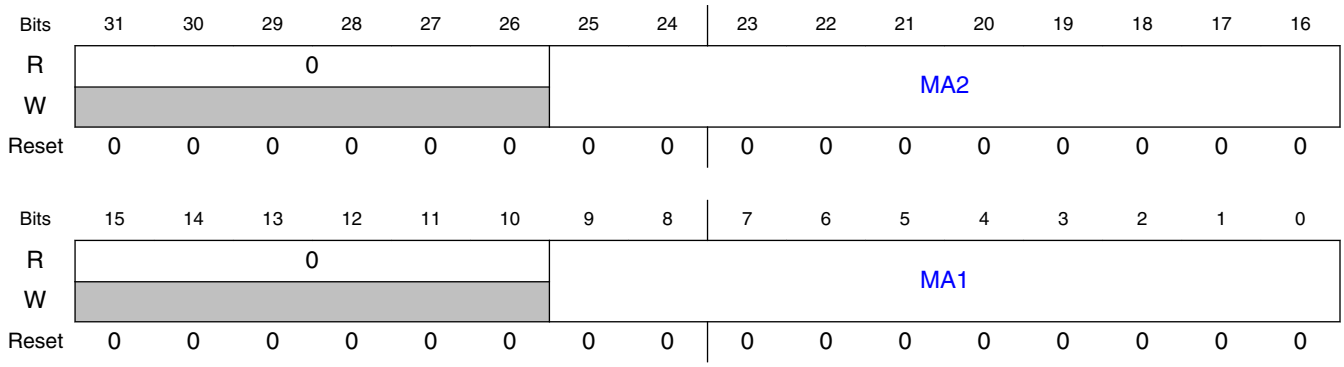
Field	Function
10 —	Reserved
9 R9T9	R9T9 Read receive data buffer 9 or write transmit data buffer 9.
8 R8T8	R8T8 Read receive data buffer 8 or write transmit data buffer 8.
7 R7T7	R7T7 Read receive data buffer 7 or write transmit data buffer 7.
6 R6T6	R6T6 Read receive data buffer 6 or write transmit data buffer 6.
5 R5T5	R5T5 Read receive data buffer 5 or write transmit data buffer 5.
4 R4T4	R4T4 Read receive data buffer 4 or write transmit data buffer 4.
3 R3T3	R3T3 Read receive data buffer 3 or write transmit data buffer 3.
2 R2T2	R2T2 Read receive data buffer 2 or write transmit data buffer 2.
1 R1T1	R1T1 Read receive data buffer 1 or write transmit data buffer 1.
0 R0T0	R0T0 Read receive data buffer 0 or write transmit data buffer 0.

51.3.1.10 LPUART Match Address Register (MATCH)

51.3.1.10.1 Offset

Register	Offset
MATCH	20h

51.3.1.10.2 Diagram



51.3.1.10.3 Fields

Field	Function
31-26 —	Reserved
25-16 MA2	Match Address 2 The MA1 and MA2 fields are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the DATA register. If a match fails, the following data is discarded. Software should only write a MAx field when the associated BAUD[MAEN] bit is clear.
15-10 —	Reserved
9-0 MA1	Match Address 1 The MA1 and MA2 fields are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the DATA register. If a match fails, the following data is discarded. Software should only write a MAx field when the associated BAUD[MAEN] bit is clear.

51.3.1.11 LPUART Modem IrDA Register (MODIR)

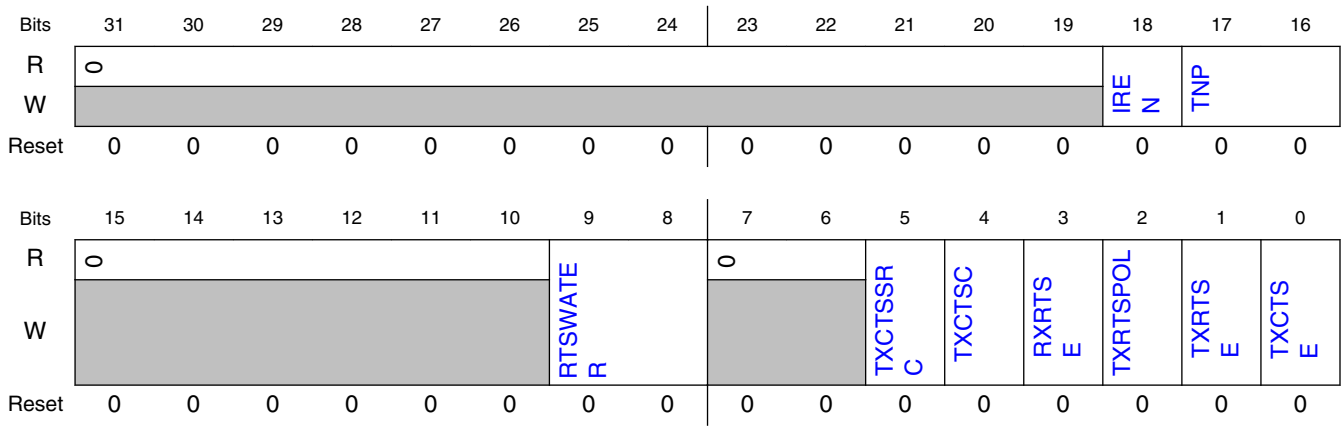
51.3.1.11.1 Offset

Register	Offset
MODIR	24h

51.3.1.11.2 Function

The MODEM register controls options for setting the modem configuration.

51.3.1.11.3 Diagram



51.3.1.11.4 Fields

Field	Function
31-19 —	Reserved
18 IREN	Infrared enable Enables/disables the infrared modulation/demodulation. This bit should only be changed when the transmitter and receiver are both disabled. 0b - IR disabled. 1b - IR enabled.
17-16 TNP	Transmitter narrow pulse Configures whether the LPUART transmits a 1/OSR, 2/OSR, 3/OSR or 4/OSR narrow pulse when IR is enabled. This bit should only be changed when the transmitter and receiver are both disabled. The IR pulse width should be configured to less than half of the oversampling ratio. Common pulse widths are 3/16, 1/16, 1/32 or 1/4 of the bit length. These can be configured by selecting the appropriate oversample ratio and pulse width. 00b - 1/OSR. 01b - 2/OSR. 10b - 3/OSR. 11b - 4/OSR.
15-10 —	Reserved
9-8 RTSWATER	Receive RTS Configuration Configures the assertion and negation of the RX RTS_B output. The RX RTS_B output negates when the number of empty words in the receive FIFO is greater or equal to RTSWATER. If RTSWATER is configured to zero, RTS_B negates when the receive FIFO is full. For the purpose of RX RTS_B generation, the number of words in the receive FIFO updates when a start bit is detected. This supports additional latency between RTS_B negation and the external transmitter ceasing transmission. If both RX RTS_B and address/data matching is enabled, then RTS_B could assert at the end of a character if there is not a match.

Table continues on the next page...

Field	Function
	This field should only be changed when the receiver is disabled.
7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0b - CTS input is the CTS_B pin. 1b - CTS input is the inverted Receiver Match result.
4 TXCTSC	Transmit CTS Configuration Configures if the CTS state is checked at the start of each character or only when the transmitter is idle. 0b - CTS input is sampled at the start of each character. 1b - CTS input is sampled when the transmitter is idle.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. This bit should only be changed when the receiver is disabled. **NOTE:** Do not set both RXRTSE and TXRTSE. 0b - The receiver has no effect on RTS. 1b - RTS is deasserted if the receiver data register is full or a start bit has been detected that would cause the receiver data register to become full. RTS is asserted if the receiver data register is not full and has not detected a start bit that would cause the receiver data register to become full.
2 TXRTSPOL	Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS will remain negated in the active low state unless TXRTSE is set. This bit should only be changed when the transmitter is disabled. 0b - Transmitter RTS is active low. 1b - Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission. This bit should only be changed when the transmitter is disabled. 0b - The transmitter has no effect on RTS. 1b - When a character is placed into an empty transmitter data buffer , RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit.
0 TXCTSE	Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE. 0b - CTS has no effect on the transmitter. 1b - Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.

51.3.1.12 LPUART FIFO Register (FIFO)

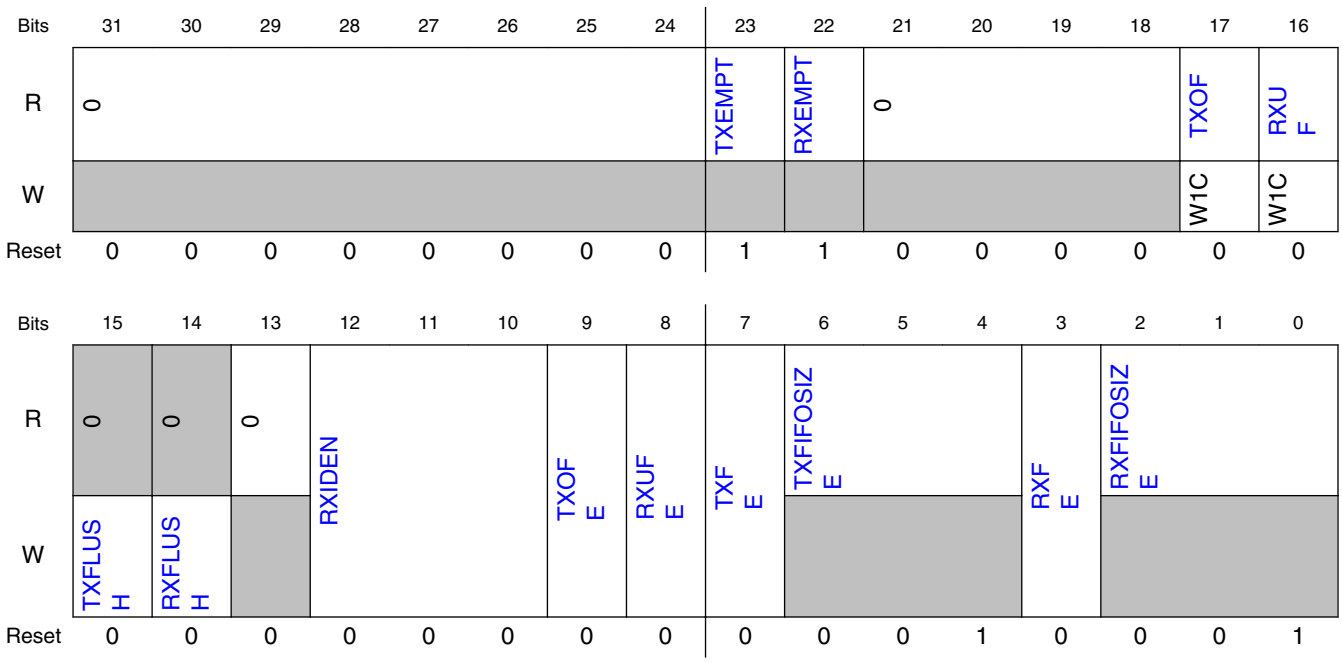
51.3.1.12.1 Offset

Register	Offset
FIFO	28h

51.3.1.12.2 Function

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when CTRL[RE] and CTRL[TE] are cleared/not set and when the data buffer/FIFO is empty.

51.3.1.12.3 Diagram



51.3.1.12.4 Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit Buffer/FIFO Empty Asserts when there is no data in the Transmit FIFO/buffer. This field does not take into account data that is in the transmit shift register. 0b - Transmit buffer is not empty.

Table continues on the next page...

Field	Function
	1b - Transmit buffer is empty.
22 RXEMPT	Receive Buffer/FIFO Empty Asserts when there is no data in the receive FIFO/Buffer. This field does not take into account data that is in the receive shift register. 0b - Receive buffer is not empty. 1b - Receive buffer is empty.
21-18 —	Reserved
17 TXOF	Transmitter Buffer Overflow Flag Indicates that more data has been written to the transmit buffer than it can hold. This field will assert regardless of the value of TXOFE. However, an interrupt will be issued to the host only if TXOFE is set. This flag is cleared by writing a 1. 0b - No transmit buffer overflow has occurred since the last time the flag was cleared. 1b - At least one transmit buffer overflow has occurred since the last time the flag was cleared.
16 RXUF	Receiver Buffer Underflow Flag Indicates that more data has been read from the receive buffer than was present. This field will assert regardless of the value of RXUFE. However, an interrupt will be issued to the host only if RXUFE is set. This flag is cleared by writing a 1. 0b - No receive buffer underflow has occurred since the last time the flag was cleared. 1b - At least one receive buffer underflow has occurred since the last time the flag was cleared.
15 TXFLUSH	Transmit FIFO/Buffer Flush Writing to this field causes all data that is stored in the transmit FIFO/buffer to be flushed. This does not affect data that is in the transmit shift register. 0b - No flush operation occurs. 1b - All data in the transmit FIFO/Buffer is cleared out.
14 RXFLUSH	Receive FIFO/Buffer Flush Writing to this field causes all data that is stored in the receive FIFO/buffer to be flushed. This does not affect data that is in the receive shift register. 0b - No flush operation occurs. 1b - All data in the receive FIFO/buffer is cleared out.
13 —	Reserved
12-10 RXIDEN	Receiver Idle Empty Enable When set, enables the assertion of RDRF when the receiver is idle for a number of idle characters and the FIFO is not empty. 000b - Disable RDRF assertion due to partially filled FIFO when receiver is idle. 001b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 1 character. 010b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 2 characters. 011b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 4 characters. 100b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 8 characters. 101b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 16 characters. 110b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 32 characters. 111b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 64 characters.
9 TXOFE	Transmit FIFO Overflow Interrupt Enable When this field is set, the TXOF flag generates an interrupt to the host. 0b - TXOF flag does not generate an interrupt to the host. 1b - TXOF flag generates an interrupt to the host.
8	Receive FIFO Underflow Interrupt Enable

Table continues on the next page...

Field	Function
RXUFE	When this field is set, the RXUF flag generates an interrupt to the host. 0b - RXUF flag does not generate an interrupt to the host. 1b - RXUF flag generates an interrupt to the host.
7 TXFE	Transmit FIFO Enable When this field is set, the built-in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0b - Transmit FIFO is not enabled. Buffer is depth 1. 1b - Transmit FIFO is enabled. Buffer is depth indicated by TXFIFOSIZE.
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth The maximum number of transmit datawords that can be stored in the transmit buffer. This field is read only. 000b - Transmit FIFO/Buffer depth = 1 dataword. 001b - Transmit FIFO/Buffer depth = 4 datawords. 010b - Transmit FIFO/Buffer depth = 8 datawords. 011b - Transmit FIFO/Buffer depth = 16 datawords. 100b - Transmit FIFO/Buffer depth = 32 datawords. 101b - Transmit FIFO/Buffer depth = 64 datawords. 110b - Transmit FIFO/Buffer depth = 128 datawords. 111b - Transmit FIFO/Buffer depth = 256 datawords
3 RXFE	Receive FIFO Enable When this field is set, the built-in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0b - Receive FIFO is not enabled. Buffer is depth 1. 1b - Receive FIFO is enabled. Buffer is depth indicted by RXFIFOSIZE.
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read only. 000b - Receive FIFO/Buffer depth = 1 dataword. 001b - Receive FIFO/Buffer depth = 4 datawords. 010b - Receive FIFO/Buffer depth = 8 datawords. 011b - Receive FIFO/Buffer depth = 16 datawords. 100b - Receive FIFO/Buffer depth = 32 datawords. 101b - Receive FIFO/Buffer depth = 64 datawords. 110b - Receive FIFO/Buffer depth = 128 datawords. 111b - Receive FIFO/Buffer depth = 256 datawords.

51.3.1.13 LPUART Watermark Register (WATER)

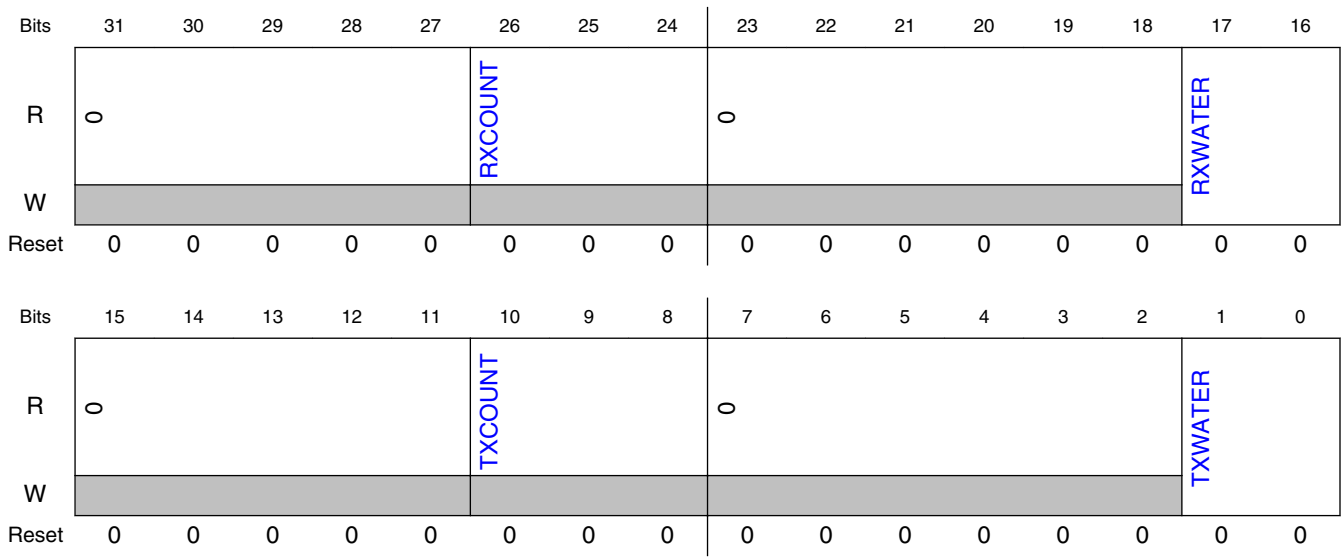
51.3.1.13.1 Offset

Register	Offset
WATER	2Ch

51.3.1.13.2 Function

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when CTRL[TE] is not set.

51.3.1.13.3 Diagram



51.3.1.13.4 Fields

Field	Function
31-27 —	Reserved
26-24 RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.
23-18 —	Reserved
17-16 RXWATER	Receive Watermark When the number of datawords in the receive FIFO/buffer is greater than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE] and must be greater than 0.
15-11 —	Reserved

Table continues on the next page...

Field	Function
10-8 TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with FIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.
7-2 —	Reserved
1-0 TXWATER	Transmit Watermark When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

51.3.2 LPUART register descriptions

51.3.2.1 LPUART Memory map

LPUART2 base address: 410A_B000h

LPUART3 base address: 410A_C000h

LPUART4 base address: 402D_0000h

LPUART5 base address: 402E_0000h

LPUART6 base address: 40A6_0000h

LPUART7 base address: 40A7_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0401_0003h
4h	Parameter Register (PARAM)	32	RO	0000_0303h
8h	LPUART Global Register (GLOBAL)	32	RW	0000_0000h
Ch	LPUART Pin Configuration Register (PINCFG)	32	RW	0000_0000h
10h	LPUART Baud Rate Register (BAUD)	32	RW	0F00_0004h
14h	LPUART Status Register (STAT)	32	RW	00C0_0000h
18h	LPUART Control Register (CTRL)	32	RW	0000_0000h
1Ch	LPUART Data Register (DATA)	32	RW	0000_1000h
20h	LPUART Match Address Register (MATCH)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
24h	LPUART Modem IrDA Register (MODIR)	32	RW	0000_0000h
28h	LPUART FIFO Register (FIFO)	32	RW	00C0_0022h
2Ch	LPUART Watermark Register (WATER)	32	RW	0000_0000h

51.3.2.2 Version ID Register (VERID)

51.3.2.2.1 Offset

Register	Offset
VERID	0h

51.3.2.2.2 Function

The Version ID register indicates the version integrated for this instance on the device and also indicates inclusion/exclusion of several optional features.

51.3.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

51.3.2.2.4 Fields

Field	Function
31-24	Major Version Number
MAJOR	This read only field returns the major version number for the module specification.
23-16	Minor Version Number

Table continues on the next page...

Field	Function
MINOR	This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number This read only field returns the feature set number. 0000000000000001b - Standard feature set. 0000000000000011b - Standard feature set with MODEM/IrDA support.

51.3.2.3 Parameter Register (PARAM)

51.3.2.3.1 Offset

Register	Offset
PARAM	4h

51.3.2.3.2 Function

The Parameter register indicates the parameter configuration for this instance on the device

51.3.2.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

51.3.2.3.4 Fields

Field	Function
31-16 —	Reserved
15-8	Receive FIFO Size

Table continues on the next page...

Field	Function
RXFIFO	The number of words in the receive FIFO is 2^RXFIFO.
7-0 TXFIFO	Transmit FIFO Size The number of words in the transmit FIFO is 2^TXFIFO.

51.3.2.4 LPUART Global Register (GLOBAL)

51.3.2.4.1 Offset

Register	Offset
GLOBAL	8h

51.3.2.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														RST	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

51.3.2.4.3 Fields

Field	Function
31-2 —	Reserved
1 RST	Software Reset Resets all internal logic and registers, except the Global Register. Remains set until cleared by software. 0b - Module is not reset. 1b - Module is reset.
0 —	Reserved

51.3.2.5 LPUART Pin Configuration Register (PINCFG)

51.3.2.5.1 Offset

Register	Offset
PINCFG	Ch

51.3.2.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														TRGSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

51.3.2.5.3 Fields

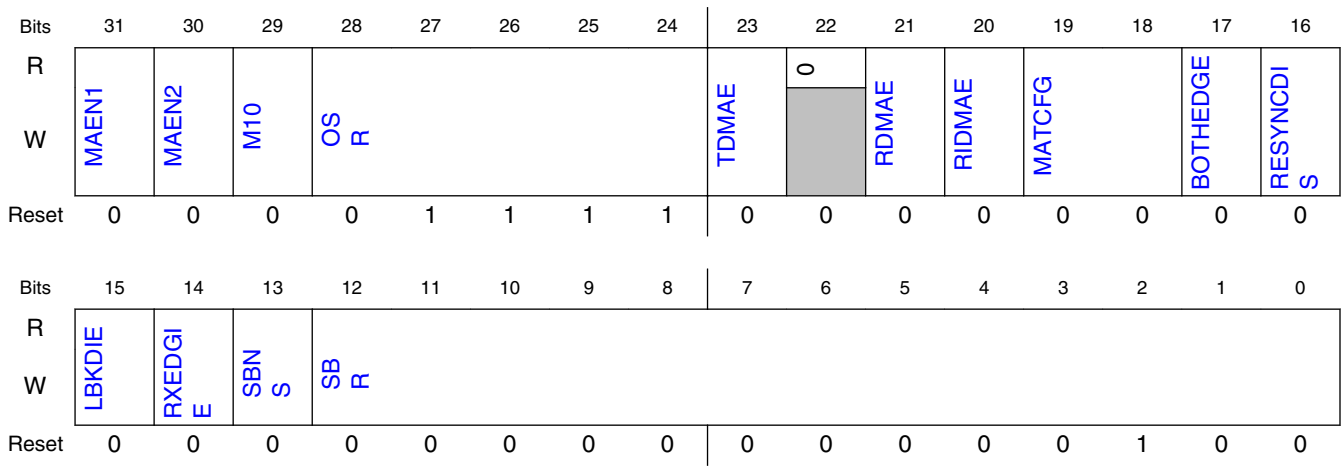
Field	Function
31-2 —	Reserved
1-0 TRGSEL	Trigger Select Configures the input trigger usage. This field should only be changed when the transmitter and receiver are both disabled. 00b - Input trigger is disabled. 01b - Input trigger is used instead of RXD pin input. 10b - Input trigger is used instead of CTS_B pin input. 11b - Input trigger is used to modulate the TXD pin output. The TXD pin output (after TXINV configuration) is ANDed with the input trigger.

51.3.2.6 LPUART Baud Rate Register (BAUD)

51.3.2.6.1 Offset

Register	Offset
BAUD	10h

51.3.2.6.2 Diagram



51.3.2.6.3 Fields

Field	Function
31 MAEN1	Match Address Mode Enable 1 0b - Normal operation. 1b - Enables automatic address matching or data matching mode for MATCH[MA1].
30 MAEN2	Match Address Mode Enable 2 0b - Normal operation. 1b - Enables automatic address matching or data matching mode for MATCH[MA2].
29 M10	10-bit Mode select The M10 bit causes a tenth bit to be part of the serial transmission. This bit should only be changed when the transmitter and receiver are both disabled. 0b - Receiver and transmitter use 7-bit to 9-bit data characters. 1b - Receiver and transmitter use 10-bit data characters.
28-24 OSR	Oversampling Ratio This field configures the oversampling ratio for the receiver. This field should only be changed when the transmitter and receiver are both disabled. 00000b - Writing 0 to this field will result in an oversampling ratio of 16 00001b - Reserved 00010b - Reserved 00011b - Oversampling ratio of 4, requires BOTHEDGE to be set. 00100b - Oversampling ratio of 5, requires BOTHEDGE to be set. 00101b - Oversampling ratio of 6, requires BOTHEDGE to be set. 00110b - Oversampling ratio of 7, requires BOTHEDGE to be set. 00111b - Oversampling ratio of 8.

Table continues on the next page...

Field	Function
	01000b - Oversampling ratio of 9. 01001b - Oversampling ratio of 10. 01010b - Oversampling ratio of 11. 01011b - Oversampling ratio of 12. 01100b - Oversampling ratio of 13. 01101b - Oversampling ratio of 14. 01110b - Oversampling ratio of 15. 01111b - Oversampling ratio of 16. 10000b - Oversampling ratio of 17. 10001b - Oversampling ratio of 18. 10010b - Oversampling ratio of 19. 10011b - Oversampling ratio of 20. 10100b - Oversampling ratio of 21. 10101b - Oversampling ratio of 22. 10110b - Oversampling ratio of 23. 10111b - Oversampling ratio of 24. 11000b - Oversampling ratio of 25. 11001b - Oversampling ratio of 26. 11010b - Oversampling ratio of 27. 11011b - Oversampling ratio of 28. 11100b - Oversampling ratio of 29. 11101b - Oversampling ratio of 30. 11110b - Oversampling ratio of 31. 11111b - Oversampling ratio of 32.
23 TDMAE	Transmitter DMA Enable TDMAE configures the transmit data register empty flag, STAT[TDRE], to generate a DMA request. 0b - DMA request disabled. 1b - DMA request enabled.
22 —	Reserved
21 RDMAE	Receiver Full DMA Enable RDMAE configures the receiver data register full flag, STAT[RDRF], to generate a DMA request. 0b - DMA request disabled. 1b - DMA request enabled.
20 RIDMAE	Receiver Idle DMA Enable RIDMAE configures the receiver idle flag, STAT[IDLE], to generate a DMA request. When this bit is set, reading the DATA register when either DATA[RXEMPT] or DATA[IDLINE] bit is set, will generate an End Of Packet response until the completion of the existing DMA transfer. During an End of Packet response, reading the DATA register will return 0x0000_33FF and does not pull data from the FIFO. 0b - DMA request disabled. 1b - DMA request enabled.
19-18 MATCFG	Match Configuration Configures the match addressing mode used. This field should only be changed when the transmitter and receiver are both disabled. 00b - Address Match Wakeup 01b - Idle Match Wakeup 10b - Match On and Match Off 11b - Enables RWU on Data Match and Match On/Off for transmitter CTS input
17 BOTHEDGE	Both Edge Sampling Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given oversampling ratio. This bit must be set

Table continues on the next page...

Field	Function
	for oversampling ratios between x4 and x7 and is optional for higher oversampling ratios. This bit should only be changed when the receiver is disabled. 0b - Receiver samples input data using the rising edge of the baud rate clock. 1b - Receiver samples input data using the rising and falling edge of the baud rate clock.
16 RESYNCDIS	Resynchronization Disable When set, disables the resynchronization of the received data word when a data one followed by data zero transition is detected. This bit should only be changed when the receiver is disabled. 0b - Resynchronization during received data word is supported 1b - Resynchronization during received data word is disabled
15 LBKDIE	LIN Break Detect Interrupt Enable LBKDIE enables the LIN break detect flag, STAT[LBKDIF], to generate interrupt requests. 0b - Hardware interrupts from STAT[LBKDIF] flag are disabled (use polling). 1b - Hardware interrupt requested when STAT[LBKDIF] flag is 1.
14 RXEDGIE	RX Input Active Edge Interrupt Enable Enables the receive input active edge, STAT[RXEDGIF], to generate interrupt requests. Changing CTRL[LOOP] or CTRL[RSRC] when RXEDGIE is set can cause the STAT[RXEDGIF] flag to set. 0b - Hardware interrupts from STAT[RXEDGIF] are disabled. 1b - Hardware interrupt is requested when STAT[RXEDGIF] flag is 1.
13 SBNS	Stop Bit Number Select SBNS determines whether data characters have one or two stop bits. This bit should only be changed when the transmitter and receiver are both disabled. 0b - One stop bit. 1b - Two stop bits.
12-0 SBR	Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) × SBR)". The 13-bit baud rate setting [SBR12:SBR0] must be updated only when the transmitter and receiver are both disabled (CTRL[RE] and CTRL[TE] are both 0).

51.3.2.7 LPUART Status Register (STAT)

51.3.2.7.1 Offset

Register	Offset
STAT	14h

51.3.2.7.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W	W1C	W1C										W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MA1F	MA2F	0													
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

51.3.2.7.3 Fields

Field	Function
31 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it. 0b - No LIN break character has been detected. 1b - LIN break character has been detected.
30 RXEDGIF	RXD Pin Active Edge Interrupt Flag RXEDGIF is set whenever the receiver is enabled and an active edge, falling if RXINV = 0, rising if RXINV=1, on the RXD pin occurs. RXEDGIF is cleared by writing a 1 to it. 0b - No active edge on the receive pin has occurred. 1b - An active edge on the receive pin has occurred.
29 MSBF	MSB First Setting this bit reverses the order of the bits that are transmitted and received on the wire. This bit does not affect the polarity of the bits, the location of the parity bit or the location of the start or stop bits. This bit should only be changed when the transmitter and receiver are both disabled. 0b - LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0. 1b - MSB (bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of CTRL[M], CTRL[PE] and BAUD[M10]. Further, the first bit received after the start bit is identified as bit9, bit8, bit7 or bit6 depending on the setting of CTRL[M] and CTRL[PE].
28 RXINV	Receive Data Inversion Setting this bit reverses the polarity of the received data input. This bit should only be changed when the receiver is disabled. **NOTE:** Setting RXINV inverts the RXD input for all cases: data bits, start and stop bits, break, and idle. 0b - Receive data not inverted. 1b - Receive data inverted.
27	Receive Wake Up Idle Detect

Table continues on the next page...

Field	Function
RWUID	For RWU on idle character, RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. For address match wakeup, RWUID controls if the IDLE bit is set when the address does not match. This bit should only be changed when the receiver is disabled. 0b - During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. During address match wakeup, the IDLE bit does not set when an address does not match. 1b - During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. During address match wakeup, the IDLE bit does set when an address does not match.
26 BRK13	Break Character Generation Length BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. This bit should only be changed when the transmitter is disabled. A break character can be sent by setting CTRL[SBK] or by writing the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] clear. 0b - Break character is transmitted with length of 9 to 13 bit times. 1b - Break character is transmitted with length of 12 to 15 bit times.
25 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive data buffer. 0b - LIN break detect is disabled, normal break character can be detected. 1b - LIN break detect is enabled. LIN break character is detected at length of 11 bit times (if M = 0) or 12 (if M = 1) or 13 (M10 = 1).
24 RAF	Receiver Active Flag RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. 0b - LPUART receiver idle waiting for a start bit. 1b - LPUART receiver active (RXD input not idle).
23 TDRE	Transmit Data Register Empty Flag When the transmit FIFO is enabled, TDRE will set when the number of datawords in the transmit FIFO (DATA register) is equal to or less than the number indicated by WATER[TXWATER]). To clear TDRE, write to the DATA register until the number of words in the transmit FIFO is greater than the number indicated by WATER[TXWATER]. When the transmit FIFO is disabled,TDRE will set when the transmit DATA register is empty. To clear TDRE, write to the DATA register. TDRE is not affected by a character that is in the process of being transmitted; it is updated at the start of each transmitted character. 0b - Transmit data buffer full. 1b - Transmit data buffer empty.
22 TC	Transmission Complete Flag TC is cleared when there is a transmission in progress or when a preamble or break character is loaded. TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by writing to the DATA register to transmit new data, queuing a preamble by clearing and then setting CTRL[TE], queuing a break character by writing 1 to CTRL[SBK]. 0b - Transmitter active (sending data, a preamble, or a break). 1b - Transmitter idle (transmission activity complete).
21 RDRF	Receive Data Register Full Flag When the receive FIFO is enabled, RDRF is set when the number of datawords in the receive buffer is greater than the number indicated by WATER[RXWATER]. To clear RDRF, read the DATA register until the number of datawords in the receive data buffer is equal to or less than the number indicated by WATER[RXWATER]. When the receive FIFO is disabled, RDRF is set when the receive buffer (the DATA register) is full. To clear RDRF, read the DATA register.

Table continues on the next page...

Field	Function
	A character that is in the process of being received does not cause a change in RDRF until the entire character is received. Even if RDRF is set, the character will continue to be received until an overrun condition occurs once the entire character is received. 0b - Receive data buffer empty. 1b - Receive data buffer full.
20 IDLE	Idle Line Flag IDLE is set when the LPUART receive line becomes idle for a full character time after a period of activity. When CTRL[ILT] is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. When CTRL[ILT] is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line. To clear IDLE, write logic 1 to the IDLE flag. After IDLE has been cleared, it cannot be set again until after a new character has been stored in the receive buffer or a LIN break character has set the LBKDIF flag . IDLE is set only once even if the receive line remains idle for an extended period. 0b - No idle line detected. 1b - Idle line was detected.
19 OR	Receiver Overrun Flag OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is lost, but the data already in the LPUART data registers is not affected. If LBKDE is enabled and a LIN Break is detected, the OR field asserts if LBKDIF is not cleared before the next data character is received. While the OR flag is set, no additional data is stored in the data buffer even if sufficient room exists. To clear OR, write logic 1 to the OR flag. 0b - No overrun. 1b - Receive overrun (new LPUART data lost).
18 NF	Noise Flag The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame then noise is detected for that character. NF is set whenever the next character to be read from the DATA register was received with noise detected within the character. To clear NF, write logic 1 to the NF field. 0b - No noise detected. 1b - Noise detected in the received character in the DATA register.
17 FE	Framing Error Flag FE is set whenever the next character to be read from the DATA register was received with logic 0 detected where a stop bit was expected. To clear FE, write logic 1 to the FE field. 0b - No framing error detected. This does not guarantee the framing is correct. 1b - Framing error.
16 PF	Parity Error Flag PF is set whenever the next character to be read from the DATA register was received when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, write a logic 1 to the PF field. 0b - No parity error. 1b - Parity error.
15 MA1F	Match 1 Flag

Table continues on the next page...

Field	Function
	MA1F is set whenever the next character to be read from the DATA register matches MA1. To clear MA1F, write a logic 1 to the MA1F field. 0b - Received data is not equal to MA1 1b - Received data is equal to MA1
14 MA2F	Match 2 Flag MA2F is set whenever the next character to be read from the DATA register matches MA2. To clear MA2F, write a logic 1 to the MA2F field. 0b - Received data is not equal to MA2 1b - Received data is equal to MA2
13-0 —	Reserved

51.3.2.8 LPUART Control Register (CTRL)

51.3.2.8.1 Offset

Register	Offset
CTRL	18h

51.3.2.8.2 Function

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

51.3.2.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R8T9	R9T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MA1IE	MA2IE	0		M7	IDLECFG			LOOPS	DOZEEN	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

51.3.2.8.4 Fields

Field	Function
31 R8T9	Receive Bit 8 / Transmit Bit 9 R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading the DATA register. T9 is the tenth data bit transmitted when the LPUART is configured for 10-bit data formats. When writing 10-bit data, write T9 before writing the DATA register. If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time the DATA register is written. **NOTE:** R8 is a read-only bit and T9 is a write-only bit, the value read is different from the value written.
30 R9T8	Receive Bit 9 / Transmit Bit 8 R9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When reading 10-bit data, read R9 before reading the DATA register T8 is the ninth data bit transmitted when the LPUART is configured for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing the DATA register. If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, then it need not be written each time the DATA register is written. **NOTE:** R9 is a read-only bit and T8 is a write-only bit, the value read is different from the value written.
29 TXDIR	TXD Pin Direction in Single-Wire Mode When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the TXD pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the TXD pin. 0b - TXD pin is an input in single-wire mode. 1b - TXD pin is an output in single-wire mode.
28 TXINV	Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output. **NOTE:** Setting TXINV inverts the TXD output for all cases: data bits, start and stop bits, break, and idle. 0b - Transmit data not inverted. 1b - Transmit data inverted.
27 ORIE	Overrun Interrupt Enable This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0b - OR interrupts disabled; use polling. 1b - Hardware interrupt requested when OR is set.
26 NEIE	Noise Error Interrupt Enable This bit enables the noise flag (NF) to generate hardware interrupt requests. 0b - NF interrupts disabled; use polling. 1b - Hardware interrupt requested when NF is set.
25 FEIE	Framing Error Interrupt Enable This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0b - FE interrupts disabled; use polling. 1b - Hardware interrupt requested when FE is set.
24 PEIE	Parity Error Interrupt Enable This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0b - PF interrupts disabled; use polling). 1b - Hardware interrupt requested when PF is set.
23	Transmit Interrupt Enable

Table continues on the next page...

Field	Function
TIE	Enables STAT[TDRE] to generate interrupt requests. 0b - Hardware interrupts from TDRE disabled; use polling. 1b - Hardware interrupt requested when TDRE flag is 1.
22 TCIE	Transmission Complete Interrupt Enable for TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0b - Hardware interrupts from TC disabled; use polling. 1b - Hardware interrupt requested when TC flag is 1.
21 RIE	Receiver Interrupt Enable Enables STAT[RDRF] to generate interrupt requests. 0b - Hardware interrupts from RDRF disabled; use polling. 1b - Hardware interrupt requested when RDRF flag is 1.
20 ILIE	Idle Line Interrupt Enable ILIE enables the idle line flag, STAT[IDLE], to generate interrupt requests. 0b - Hardware interrupts from IDLE disabled; use polling. 1b - Hardware interrupt requested when IDLE flag is 1.
19 TE	Transmitter Enable Enables the LPUART transmitter. TE can also be used to queue an idle preamble by clearing and then setting TE. When TE is cleared, this register bit will read as 1 until the transmitter has completed the current character and the TXD pin is tristated. A single idle character can also be queued by writing to the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] set. 0b - Transmitter disabled. 1b - Transmitter enabled.
18 RE	Receiver Enable Enables the LPUART receiver. When RE is written to 0, this register bit will read as 1 until the receiver finishes receiving the current character (if any). 0b - Receiver disabled. 1b - Receiver enabled.
17 RWU	Receiver Wakeup Control This field can be set to place the LPUART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when CTRL[WAKE] is clear or an address match when CTRL[WAKE] is set with STAT[RWUID] is clear. **NOTE:** RWU must be set only with CTRL[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by STAT[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the LPUART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to be reasserted. 0b - Normal receiver operation. 1b - LPUART receiver in standby waiting for wakeup condition.
16 SBK	Send Break Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 9 to 13 bits, or 12 to 15 bits if STAT[BRK13] is set, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the character currently being transmitted, a second break character may be queued before software clears SBK. A single break character can also be queued by writing to the transmit FIFO with DATA[FRETSC] set and DATA[R9T9] clear. 0b - Normal transmitter operation. 1b - Queue break character(s) to be sent.

Table continues on the next page...

Field	Function
15 MA1IE	Match 1 Interrupt Enable 0b - MA1F interrupt disabled 1b - MA1F interrupt enabled
14 MA2IE	Match 2 Interrupt Enable 0b - MA2F interrupt disabled 1b - MA2F interrupt enabled
13-12 —	Reserved
11 M7	7-Bit Mode Select This bit should only be changed when the transmitter and receiver are both disabled. 0b - Receiver and transmitter use 8-bit to 10-bit data characters. 1b - Receiver and transmitter use 7-bit data characters.
10-8 IDLECFG	Idle Configuration Configures the number of idle characters that must be received before the IDLE flag is set. 000b - 1 idle character 001b - 2 idle characters 010b - 4 idle characters 011b - 8 idle characters 100b - 16 idle characters 101b - 32 idle characters 110b - 64 idle characters 111b - 128 idle characters
7 LOOPS	Loop Mode Select When LOOPS is set, the RXD pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0b - Normal operation - RXD and TXD use separate pins. 1b - Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input (see RSRC bit).
6 DOZEEN	Doze Enable 0b - LPUART is enabled in Doze mode. 1b - LPUART is disabled in Doze mode.
5 RSRC	Receiver Source Select This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0b - Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the LPUART does not use the RXD pin. 1b - Single-wire LPUART mode where the TXD pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 0b - Receiver and transmitter use 8-bit data characters. 1b - Receiver and transmitter use 9-bit data characters.
3 WAKE	Receiver Wakeup Method Select Determines which condition wakes the LPUART when RWU=1: • Address mark in the bit preceding the stop bit (or bit preceding the parity bit when parity is enabled) of the received data character, or • An idle condition on the receive pin input signal. 0b - Configures RWU for idle-line wakeup. 1b - Configures RWU with address-mark wakeup.

Table continues on the next page...

Field	Function
2 ILT	Idle Line Type Select Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. **NOTE:** In case the LPUART is programmed with ILT = 1, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count. 0b - Idle character bit count starts after start bit. 1b - Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables hardware parity generation and checking. When parity is enabled, the bit immediately before the stop bit is treated as the parity bit. 0b - No hardware parity generation or checking. 1b - Parity enabled.
0 PT	Parity Type Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even. 0b - Even parity. 1b - Odd parity.

51.3.2.9 LPUART Data Register (DATA)

51.3.2.9.1 Offset

Register	Offset
DATA	1Ch

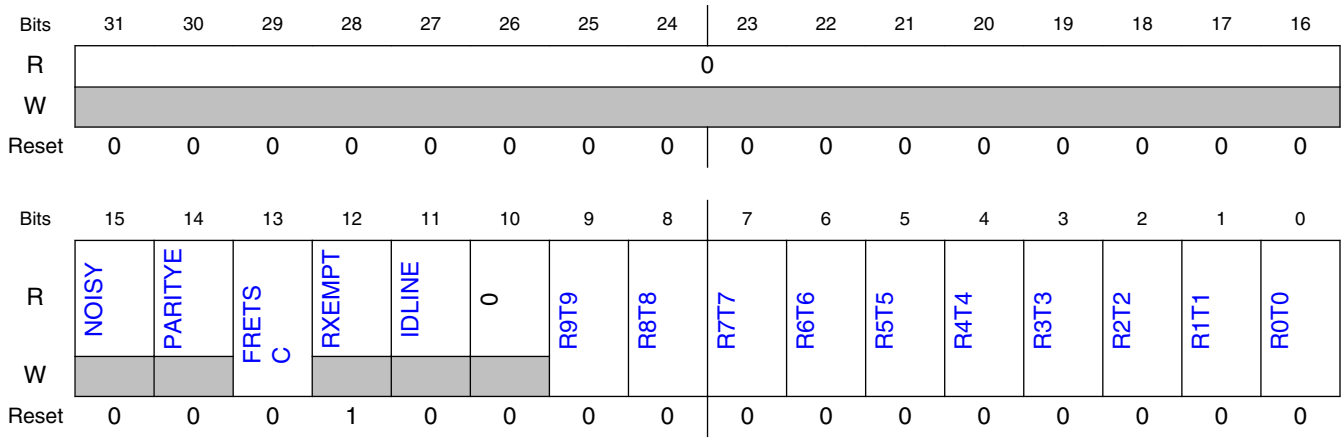
51.3.2.9.2 Function

NOTE

This register is two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer.

Reads and writes of this register are also involved in the automatic flag clearing mechanisms for some of the LPUART status flags.

51.3.2.9.3 Diagram



51.3.2.9.4 Fields

Field	Function
31-16 —	Reserved
15 NOISY	NOISY The current received dataword contained in DATA[R9:R0] was received with noise. 0b - The dataword was received without noise. 1b - The data was received with noise.
14 PARITYE	PARITYE The current received dataword contained in DATA[R9:R0] was received with a parity error. 0b - The dataword was received without a parity error. 1b - The dataword was received with a parity error.
13 FRETSC	Frame Error / Transmit Special Character For reads, indicates the current received dataword contained in DATA[R9:R0] was received with a frame error. For writes, indicates a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 is used to indicate a break character when 0 and a idle character when 1, the contents of DATA[T8:T0] should be zero. 0b - The dataword was received without a frame error on read, or transmit a normal character on write. 1b - The dataword was received with a frame error, or transmit an idle or break character on transmit.
12 RXEMPT	Receive Buffer Empty Asserts when there is no data in the receive buffer. This field does not take into account data that is in the receive shift register. 0b - Receive buffer contains valid data. 1b - Receive buffer is empty, data returned on read is not valid.
11 IDLINE	Idle Line Indicates the receiver line was idle before receiving the character in DATA[9:0]. Unlike the IDLE flag, this bit can set for the first character received when the receiver is first enabled. 0b - Receiver was not idle before receiving this character. 1b - Receiver was idle before receiving this character.

Table continues on the next page...

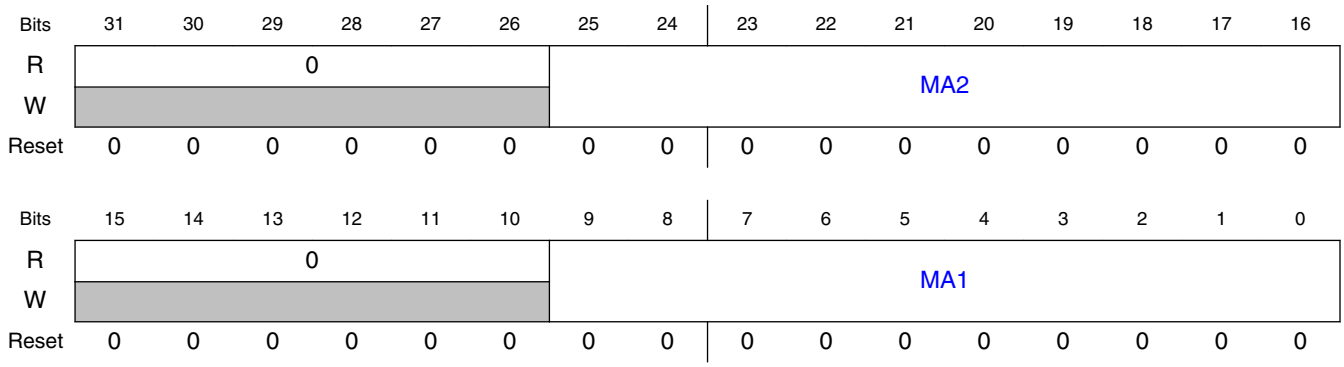
Field	Function
10 —	Reserved
9 R9T9	R9T9 Read receive data buffer 9 or write transmit data buffer 9.
8 R8T8	R8T8 Read receive data buffer 8 or write transmit data buffer 8.
7 R7T7	R7T7 Read receive data buffer 7 or write transmit data buffer 7.
6 R6T6	R6T6 Read receive data buffer 6 or write transmit data buffer 6.
5 R5T5	R5T5 Read receive data buffer 5 or write transmit data buffer 5.
4 R4T4	R4T4 Read receive data buffer 4 or write transmit data buffer 4.
3 R3T3	R3T3 Read receive data buffer 3 or write transmit data buffer 3.
2 R2T2	R2T2 Read receive data buffer 2 or write transmit data buffer 2.
1 R1T1	R1T1 Read receive data buffer 1 or write transmit data buffer 1.
0 R0T0	R0T0 Read receive data buffer 0 or write transmit data buffer 0.

51.3.2.10 LPUART Match Address Register (MATCH)

51.3.2.10.1 Offset

Register	Offset
MATCH	20h

51.3.2.10.2 Diagram



51.3.2.10.3 Fields

Field	Function
31-26 —	Reserved
25-16 MA2	Match Address 2 The MA1 and MA2 fields are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the DATA register. If a match fails, the following data is discarded. Software should only write a MAx field when the associated BAUD[MAEN] bit is clear.
15-10 —	Reserved
9-0 MA1	Match Address 1 The MA1 and MA2 fields are compared to input data addresses when the most significant bit is set and the associated BAUD[MAEN] bit is set. If a match occurs, the following data is transferred to the DATA register. If a match fails, the following data is discarded. Software should only write a MAx field when the associated BAUD[MAEN] bit is clear.

51.3.2.11 LPUART Modem IrDA Register (MODIR)

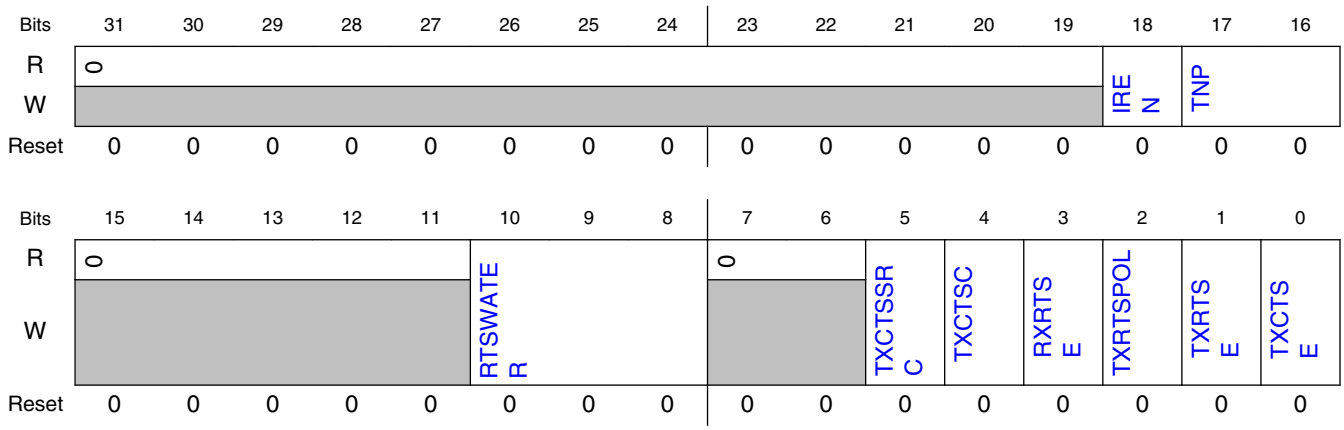
51.3.2.11.1 Offset

Register	Offset
MODIR	24h

51.3.2.11.2 Function

The MODEM register controls options for setting the modem configuration.

51.3.2.11.3 Diagram



51.3.2.11.4 Fields

Field	Function
31-19 —	Reserved
18 IREN	Infrared enable Enables/disables the infrared modulation/demodulation. This bit should only be changed when the transmitter and receiver are both disabled. 0b - IR disabled. 1b - IR enabled.
17-16 TNP	Transmitter narrow pulse Configures whether the LPUART transmits a 1/OSR, 2/OSR, 3/OSR or 4/OSR narrow pulse when IR is enabled. This bit should only be changed when the transmitter and receiver are both disabled. The IR pulse width should be configured to less than half of the oversampling ratio. Common pulse widths are 3/16, 1/16, 1/32 or 1/4 of the bit length. These can be configured by selecting the appropriate oversample ratio and pulse width. 00b - 1/OSR. 01b - 2/OSR. 10b - 3/OSR. 11b - 4/OSR.
15-11 —	Reserved
10-8 RTSWATER	Receive RTS Configuration Configures the assertion and negation of the RX RTS_B output. The RX RTS_B output negates when the number of empty words in the receive FIFO is greater or equal to RTSWATER. If RTSWATER is configured to zero, RTS_B negates when the receive FIFO is full. For the purpose of RX RTS_B generation, the number of words in the receive FIFO updates when a start bit is detected. This supports additional latency between RTS_B negation and the external transmitter ceasing transmission. If both RX RTS_B and address/data matching is enabled, then RTS_B could assert at the end of a character if there is not a match.

Table continues on the next page...

Field	Function
	This field should only be changed when the receiver is disabled.
7-6 —	Reserved
5 TXCTSSRC	Transmit CTS Source Configures the source of the CTS input. 0b - CTS input is the CTS_B pin. 1b - CTS input is the inverted Receiver Match result.
4 TXCTSC	Transmit CTS Configuration Configures if the CTS state is checked at the start of each character or only when the transmitter is idle. 0b - CTS input is sampled at the start of each character. 1b - CTS input is sampled when the transmitter is idle.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. This bit should only be changed when the receiver is disabled. **NOTE:** Do not set both RXRTSE and TXRTSE. 0b - The receiver has no effect on RTS. 1b - RTS is deasserted if the receiver data register is full or a start bit has been detected that would cause the receiver data register to become full. RTS is asserted if the receiver data register is not full and has not detected a start bit that would cause the receiver data register to become full.
2 TXRTSPOL	Transmitter request-to-send polarity Controls the polarity of the transmitter RTS. TXRTSPOL does not affect the polarity of the receiver RTS. RTS will remain negated in the active low state unless TXRTSE is set. This bit should only be changed when the transmitter is disabled. 0b - Transmitter RTS is active low. 1b - Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission. This bit should only be changed when the transmitter is disabled. 0b - The transmitter has no effect on RTS. 1b - When a character is placed into an empty transmitter data buffer , RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit.
0 TXCTSE	Transmitter clear-to-send enable TXCTSE controls the operation of the transmitter. TXCTSE can be set independently from the state of TXRTSE and RXRTSE. 0b - CTS has no effect on the transmitter. 1b - Enables clear-to-send operation. The transmitter checks the state of CTS each time it is ready to send a character. If CTS is asserted, the character is sent. If CTS is deasserted, the signal TXD remains in the mark state and transmission is delayed until CTS is asserted. Changes in CTS as a character is being sent do not affect its transmission.

51.3.2.12 LPUART FIFO Register (FIFO)

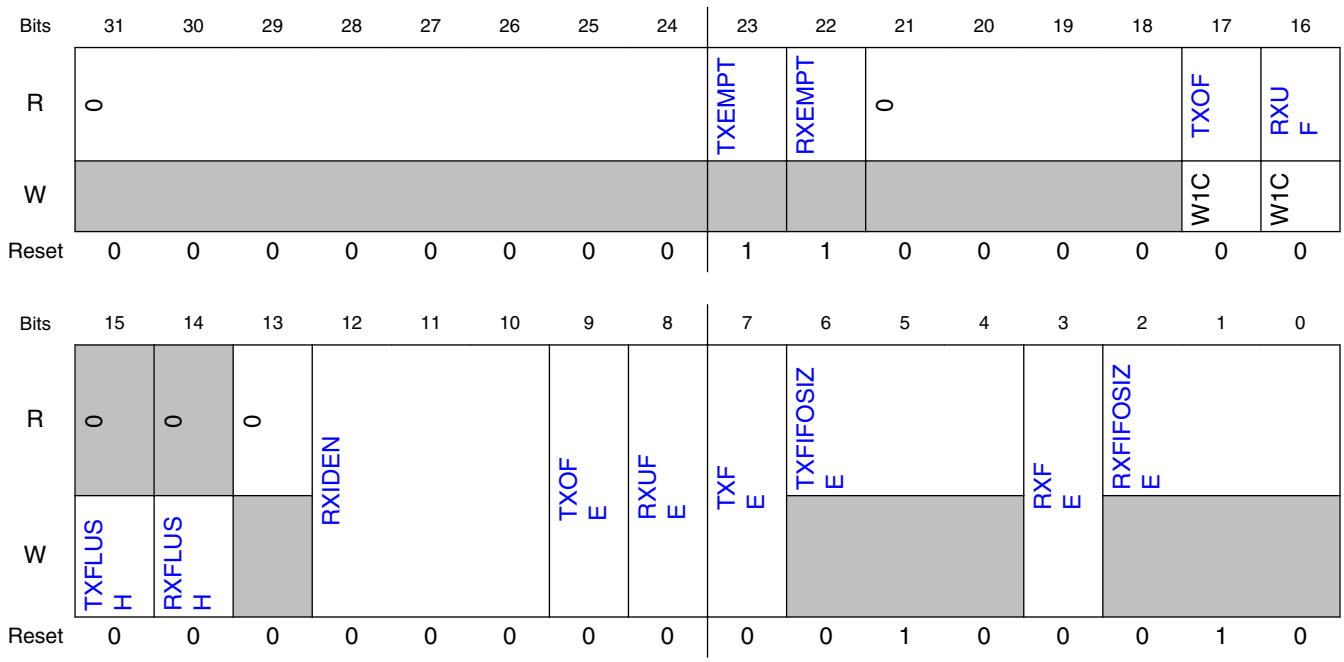
51.3.2.12.1 Offset

Register	Offset
FIFO	28h

51.3.2.12.2 Function

This register provides the ability for the programmer to turn on and off FIFO functionality. It also provides the size of the FIFO that has been implemented. This register may be read at any time. This register must be written only when CTRL[RE] and CTRL[TE] are cleared/not set and when the data buffer/FIFO is empty.

51.3.2.12.3 Diagram



51.3.2.12.4 Fields

Field	Function
31-24 —	Reserved
23 TXEMPT	Transmit Buffer/FIFO Empty Asserts when there is no data in the Transmit FIFO/buffer. This field does not take into account data that is in the transmit shift register. 0b - Transmit buffer is not empty.

Table continues on the next page...

Field	Function
	1b - Transmit buffer is empty.
22 RXEMPT	Receive Buffer/FIFO Empty Asserts when there is no data in the receive FIFO/Buffer. This field does not take into account data that is in the receive shift register. 0b - Receive buffer is not empty. 1b - Receive buffer is empty.
21-18 —	Reserved
17 TXOF	Transmitter Buffer Overflow Flag Indicates that more data has been written to the transmit buffer than it can hold. This field will assert regardless of the value of TXOFE. However, an interrupt will be issued to the host only if TXOFE is set. This flag is cleared by writing a 1. 0b - No transmit buffer overflow has occurred since the last time the flag was cleared. 1b - At least one transmit buffer overflow has occurred since the last time the flag was cleared.
16 RXUF	Receiver Buffer Underflow Flag Indicates that more data has been read from the receive buffer than was present. This field will assert regardless of the value of RXUFE. However, an interrupt will be issued to the host only if RXUFE is set. This flag is cleared by writing a 1. 0b - No receive buffer underflow has occurred since the last time the flag was cleared. 1b - At least one receive buffer underflow has occurred since the last time the flag was cleared.
15 TXFLUSH	Transmit FIFO/Buffer Flush Writing to this field causes all data that is stored in the transmit FIFO/buffer to be flushed. This does not affect data that is in the transmit shift register. 0b - No flush operation occurs. 1b - All data in the transmit FIFO/Buffer is cleared out.
14 RXFLUSH	Receive FIFO/Buffer Flush Writing to this field causes all data that is stored in the receive FIFO/buffer to be flushed. This does not affect data that is in the receive shift register. 0b - No flush operation occurs. 1b - All data in the receive FIFO/buffer is cleared out.
13 —	Reserved
12-10 RXIDEN	Receiver Idle Empty Enable When set, enables the assertion of RDRF when the receiver is idle for a number of idle characters and the FIFO is not empty. 000b - Disable RDRF assertion due to partially filled FIFO when receiver is idle. 001b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 1 character. 010b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 2 characters. 011b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 4 characters. 100b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 8 characters. 101b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 16 characters. 110b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 32 characters. 111b - Enable RDRF assertion due to partially filled FIFO when receiver is idle for 64 characters.
9 TXOFE	Transmit FIFO Overflow Interrupt Enable When this field is set, the TXOF flag generates an interrupt to the host. 0b - TXOF flag does not generate an interrupt to the host. 1b - TXOF flag generates an interrupt to the host.
8	Receive FIFO Underflow Interrupt Enable

Table continues on the next page...

Field	Function
RXUFE	When this field is set, the RXUF flag generates an interrupt to the host. 0b - RXUF flag does not generate an interrupt to the host. 1b - RXUF flag generates an interrupt to the host.
7 TXFE	Transmit FIFO Enable When this field is set, the built-in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by TXFIFOSIZE. If this field is not set, the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0b - Transmit FIFO is not enabled. Buffer is depth 1. 1b - Transmit FIFO is enabled. Buffer is depth indicated by TXFIFOSIZE.
6-4 TXFIFOSIZE	Transmit FIFO Buffer Depth The maximum number of transmit datawords that can be stored in the transmit buffer. This field is read only. 000b - Transmit FIFO/Buffer depth = 1 dataword. 001b - Transmit FIFO/Buffer depth = 4 datawords. 010b - Transmit FIFO/Buffer depth = 8 datawords. 011b - Transmit FIFO/Buffer depth = 16 datawords. 100b - Transmit FIFO/Buffer depth = 32 datawords. 101b - Transmit FIFO/Buffer depth = 64 datawords. 110b - Transmit FIFO/Buffer depth = 128 datawords. 111b - Transmit FIFO/Buffer depth = 256 datawords
3 RXFE	Receive FIFO Enable When this field is set, the built-in FIFO structure for the receive buffer is enabled. The size of the FIFO structure is indicated by the RXFIFOSIZE field. If this field is not set, the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both CTRL[TE] and CTRL[RE] must be cleared prior to changing this field. 0b - Receive FIFO is not enabled. Buffer is depth 1. 1b - Receive FIFO is enabled. Buffer is depth indicted by RXFIFOSIZE.
2-0 RXFIFOSIZE	Receive FIFO Buffer Depth The maximum number of receive datawords that can be stored in the receive buffer before an overrun occurs. This field is read only. 000b - Receive FIFO/Buffer depth = 1 dataword. 001b - Receive FIFO/Buffer depth = 4 datawords. 010b - Receive FIFO/Buffer depth = 8 datawords. 011b - Receive FIFO/Buffer depth = 16 datawords. 100b - Receive FIFO/Buffer depth = 32 datawords. 101b - Receive FIFO/Buffer depth = 64 datawords. 110b - Receive FIFO/Buffer depth = 128 datawords. 111b - Receive FIFO/Buffer depth = 256 datawords.

51.3.2.13 LPUART Watermark Register (WATER)

51.3.2.13.1 Offset

Register	Offset
WATER	2Ch

51.3.2.13.2 Function

This register provides the ability to set a programmable threshold for notification of needing additional transmit data. This register may be read at any time but must be written only when CTRL[TE] is not set.

51.3.2.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				RXCOUNT				0					RXWATER		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				TXCOUNT				0					TXWATER		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

51.3.2.13.4 Fields

Field	Function
31-28 —	Reserved
27-24 RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive FIFO/buffer. If a dataword is being received, that is, in the receive shift register, it is not included in the count. This value may be used in conjunction with FIFO[RXFIFOSIZE] to calculate how much room is left in the receive FIFO/buffer.
23-19 —	Reserved
18-16 RXWATER	Receive Watermark When the number of datawords in the receive FIFO/buffer is greater than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in RXWATER must be set to be less than the receive FIFO/buffer size as indicated by FIFO[RXFIFOSIZE] and FIFO[RXFE] and must be greater than 0.
15-12 —	Reserved
11-8 TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit FIFO/buffer. If a dataword is being transmitted, that is, in the transmit shift register, it is not included in the count. This value may be used in conjunction with FIFO[TXFIFOSIZE] to calculate how much room is left in the transmit FIFO/buffer.
7-3	Reserved

Table continues on the next page...

Field	Function
—	
2-0 TXWATER	Transmit Watermark When the number of datawords in the transmit FIFO/buffer is equal to or less than the value in this register field, an interrupt or a DMA request is generated. For proper operation, the value in TXWATER must be set to be less than the size of the transmit buffer/FIFO size as indicated by FIFO[TXFIFOSIZE] and FIFO[TXFE].

51.4 Functional description

The LPUART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following describes each of the blocks of the LPUART.

51.4.1 Clocking and Resets

Table 51-3. Clocks

LPUART Functional clock	The LPUART functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support transmit and/or receive, including low power wakeups.
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPUART transmit and receive registers, including the FIFOs.

Table 51-4. Resets

Chip reset	The logic and registers for the LPUART transmitter and receiver are reset to their default state on a chip reset.
Software reset	Resets the LPUART logic and registers to their default state, except for the Global Register. The LPUART software reset is in the Global Register GLOBAL[RST].
FIFO reset	The LPUART implements write-only control bits that reset the transmit FIFO (FIFO[TXFLUSH]) and receive FIFO (FIFO[RXFLUSH]). After a FIFO is reset, that FIFO is empty.

51.4.2 Baud rate generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to BAUD[SBR] determines the baud clock divisor for the asynchronous LPUART baud clock. The baud

rate clock drives the receiver, while the transmitter is driven by a bit clock which is generated from baud rate clock divided by the over sampling ratio. Depending on the over sampling ratio, the receiver has an acquisition rate of 4 to 32 samples per bit time.

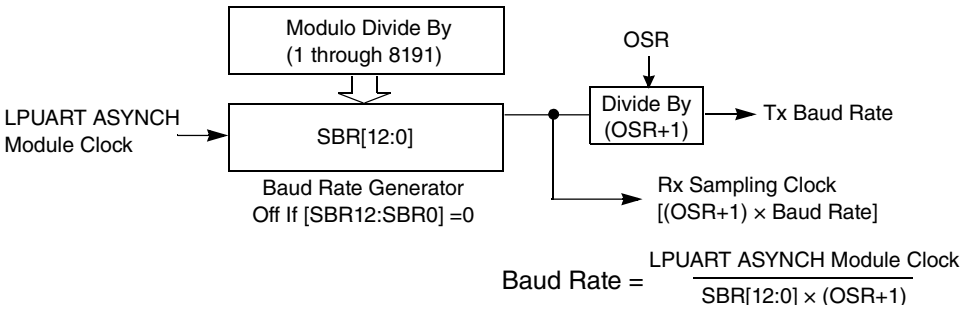


Figure 51-3. LPUART baud rate generation

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can cause phase shift.

The baud rate generation is a free-running counter that continues whenever the transmitter or receiver is enabled. The transmitter bit clock continues whenever the transmitter is enabled, each transmitted character will align to the next edge of the transmit bit clock.

51.4.3 Transmitter functional description

This section describes the overall block diagram for the LPUART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (TXD) idle state defaults to logic high, CTRL[TXINV] is cleared following reset. The transmitter output is inverted by setting CTRL[TXINV]. The transmitter is enabled by setting the CTRL[TE] bit. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the DATA register.

The central element of the LPUART transmitter is the transmit shift register that is 9-bit to 13 bits long depending on the setting in the CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SBNS] control bits. For the remainder of this section, assume CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SBNS] are cleared, selecting the normal 8-bit data

mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (STAT[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at the DATA register.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the TXD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TXD high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character or break character), although the transmitter will not start transmitting another character.

51.4.3.1 Send break and queued idle

The CTRL[SBK] bit sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 9-bit to 12-bit times including the start and stop bits. A longer break of 13-bit times can be enabled by setting STAT[BRK13]. Normally, a program would wait for STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to the CTRL[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If CTRL[SBK] remains 1 when the queued break moves into the shifter, synchronized to the baud rate clock, an additional break character is queued. When the LPUART is the receiving device, a break character is received as 0s in all data bits and a framing error (STAT[FE] = 1) is detected.

A break character can also be transmitted by writing to the DATA register with bit 13 set and the data bits clear. This supports transmitting the break character as part of the normal data stream and also allows the DMA to transmit a break character.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for STAT[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, then write 0 and then write 1 to the CTRL[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while CTRL[TE] is cleared, the LPUART transmitter never actually releases control of the TXD pin.

An idle character can also be transmitted by writing to the DATA register with bit 13 set and the data bits also set. This supports transmitting the idle character as part of the normal data stream and also allows the DMA to transmit a idle character.

The length of the break character is affected by the STAT[BRK13], CTRL[M], CTRL[M7], BAUD[M10] and BAUD[SNBS] bits as shown below.

Table 51-5. Break character length

BRK13	M	M10	M7	SBNS	Break character length
0	0	0	0	0	10 bit times
0	0	0	0	1	11 bit times
0	0	0	1	0	9 bit times
0	0	0	1	1	10 bit times
0	1	0	X	0	11 bit times
0	1	0	X	1	12 bit times
0	X	1	X	0	12 bit times
0	X	1	X	1	13 bit times
1	0	0	0	0	13 bit times
1	0	0	0	1	13 bit times
1	0	0	1	0	12 bit times
1	0	0	1	1	12 bit times
1	1	0	X	0	14 bit times
1	1	0	X	1	14 bit times
1	X	1	X	0	15 bit times
1	X	1	X	1	15 bit times

51.4.3.2 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of CTS_B. If the clear-to-send operation is enabled, the character is transmitted when CTS_B is asserted. If CTS_B is deasserted in the middle of a transmission with characters remaining in the transmitter data buffer, the character in the shift register is sent and TXD remains in the mark state until CTS_B is reasserted. The CTS_B pin must assert for longer than one bit period to guarantee a new transmission is started when the transmitter is idle with data to send.

If the clear-to-send operation is disabled, the transmitter ignores the state of CTS_B.

The transmitter's CTS_B signal can also be enabled even if the same LPUART receiver's RTS_B signal is disabled.

51.4.3.3 Transceiver driver enable

The transmitter can use RTS_B as an enable signal for the driver of an external transceiver. See Transceiver driver enable using RTS_B for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, RTS_B asserts one bit time before the start bit is transmitted. RTS_B remains asserted for the whole time that the transmitter data buffer has any characters. RTS_B deasserts one bit time after all characters in the transmitter data buffer and shift register are completely sent, including the last stop bit. Transmitting a break character also asserts RTS_B, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's RTS_B signal asserts only when the transmitter is enabled. However, the transmitter's RTS_B signal is unaffected by its CTS_B signal. RTS_B will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

51.4.3.4 Transceiver driver enable using RTS_B

RS-485 is a multiple drop communication protocol in which the LPUART transceiver's driver is 3-stated unless the LPUART is driving. The RTS_B signal can be used by the transmitter to enable the driver of a transceiver. The polarity of RTS_B can be matched to the polarity of the transceiver's driver enable signal.

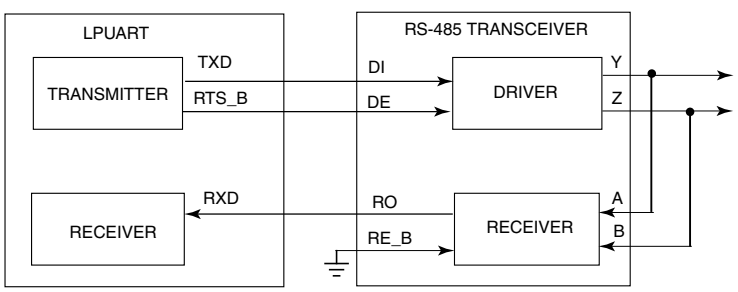


Figure 51-4. Transceiver driver enable using RTS_B

In the figure, the receiver enable signal is asserted. Another option for this connection is to connect RTS_B to both DE and RE_B. The transceiver's receiver is disabled while driving. A pullup can pull RXD to a non-floating value during this time. This option can be refined further by operating the LPUART in single wire mode, freeing the RXD pin for other uses.

51.4.4 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, different variations of the receiver wakeup function are explained.

The receiver input is inverted by setting STAT[RXINV]. The receiver is enabled by setting the CTRL[RE] bit. Character frames consist of a start bit of logic 0, seven to ten data bits (msb or lsb first), and one or two stop bits of logic 1. For information about 7-bit, 9-bit or 10-bit data mode, refer to Data Modes. For the remainder of this discussion, assume the LPUART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (STAT[RDRF]) status flag is set. If [RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost. Because the LPUART receiver is double-buffered, the program has one full character time after [RDRF] is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (STAT[RDRF] = 1), it gets the data from the receive data register by reading the DATA register. Refer to Interrupts and status flags for details about flag clearing.

51.4.4.1 Data sampling technique

The LPUART receiver supports a configurable oversampling rate of between 4× and 32× of the baud rate clock for sampling. The receiver starts by taking logic level samples at the oversampling rate times the baud rate to search for a falling edge on the RXD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a received character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts the sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at (OSR/2), (OSR/2)+1, and (OSR/2)+2 to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any

sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (STAT[NF]) is set when the received character is transferred to the receive data buffer.

When the LPUART receiver is configured to sample on both edges of the baud rate clock, the number of segments in each received bit is effectively doubled (from 1 to OSR×2). The start and data bits are then sampled at OSR, OSR+1 and OSR+2. Sampling on both edges of the clock must be enabled for oversampling rates of 4× to 7× and is optional for higher oversampling rates.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times (unless resynchronization has been disabled). This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

51.4.4.2 Receiver wakeup operation

Receiver wakeup and receiver address matching is a hardware mechanism that allows an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wakeup, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control bit (CTRL[RWU]). When CTRL[RWU] and STAT[RWUID] bit are set, the status flags associated with the receiver, with the exception of the idle bit, STAT[IDLE], are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force CTRL[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver will ignore all characters that do not meet the address match requirements.

Table 51-6. Receiver Wakeup Options

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
0	0	X	X	Normal operation
1	0	00	00	Receiver wakeup on idle line, IDLE flag not set
1	0	00	01	Receiver wakeup on idle line, IDLE flag set

Table continues on the next page...

Table 51-6. Receiver Wakeup Options (continued)

RWU	MA1 | MA2	MATCFG	WAKE:RWUID	Receiver Wakeup
1	0	00	10	Receiver wakeup on address mark
1	1	11	10	Receiver wakeup on data match
0	1	00	X0	Address mark address match, IDLE flag not set for discarded characters
0	1	00	X1	Address mark address match, IDLE flag set for discarded characters
0	1	01	X0	Idle line address match
0	1	10	X0	Address match on and address match off, IDLE flag not set for discarded characters
0	1	10	X1	Address match on and address match off, IDLE flag set for discarded characters

51.4.4.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, CTRL[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The CTRL[M], CTRL[M7] and BAUD[M10] control bit selects 7-bit to 10-bit data mode and the BAUD[SBNS] bit selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 9 to 13 bit times because of the start and stop bits.

When CTRL[RWU] is one and STAT[RWUID] is 0, the idle condition that wakes up the receiver does not set the STAT[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the STAT[RDRF] flag and generates an interrupt if enabled. When STAT[RWUID] is 1, any idle condition sets the STAT[IDLE] flag and generates an interrupt if enabled, regardless of whether CTRL[RWU] is 0 or 1.

The idle-line type (CTRL[ILT]) control bit selects one of two ways to detect an idle line. When CTRL[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle. When CTRL[ILT] is set, the idle bit counter does not start until after the stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

51.4.4.2.2 Address-mark wakeup

When CTRL[WAKE] is set, the receiver is configured for address-mark wakeup. In this mode, CTRL[RWU] is cleared automatically when the receiver detects a logic 1 in the most significant bit of the received character. When parity is enabled, the second most significant bit is used for address-mark wakeup.

Address-mark wakeup allows messages to contain idle characters, but requires one bit be reserved for use in address frames. The logic 1 in the most significant bit (or second most significant bit when parity is enabled) of an address frame clears the CTRL[RWU] bit and sets the STAT[RDRF] flag. In this case, the character with the address-mark bit is received even though the receiver was sleeping during most of this character time.

51.4.4.2.3 Data match wakeup

When CTRL[RWU] is set, CTRL[WAKE] is set and BAUD[MATCFG] equals 11, the receiver is configured for data match wakeup. In this mode, CTRL[RWU] is cleared automatically when the receiver detects a character that matches MATCH[MA1] field when BAUD[MAEN1] is set, or that matches MATCH[MA2] when BAUD[MAEN2] is set.

51.4.4.2.4 Address Match operation

Address match operation is enabled when the BAUD[MAEN1] or BAUD[MAEN2] bit is set and BAUD[MATCFG] is equal to 00. In this function, a character received by the RXD pin with a logic 1 in the most significant bit (or second most significant bit when parity is enabled) is considered an address and is compared with the associated MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and STAT[RDRF] is set, if the comparison matches. All subsequent characters received with a logic 0 in the most significant bit (or second most significant bit when parity is enabled) are considered to be data associated with the address and are transferred to the receive data buffer. If no marked address match occurs then no transfer is made to the receive data buffer, and all following characters with logic zero in the most significant bit (or second most significant bit when parity is enabled) are also discarded. If both the BAUD[MAEN1] and BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] fields.

- If only one of BAUD[MAEN1] and BAUD[MAEN2] is asserted, a marked address is compared only with the associated match register field and data is transferred to the receive data buffer only on a match.
- If BAUD[MAEN1] and BAUD[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either of the MATCH[MA1] or MATCH[MA2] fields.

51.4.4.2.5 Idle Match operation

Idle match operation is enabled when the BAUD[MAEN1] or BAUD[MAEN2] bit is set and BAUD[MATCFG] is equal to 01. In this function, the first character received by the RXD pin after an idle line condition is considered an address and is compared with the associated MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and STAT[RDRF] is set, if the comparison matches. All subsequent characters are considered to be data associated with the address and are transferred to the receive data buffer until the next idle line condition is detected. If no address match occurs then no transfer is made to the receive data buffer, and all following frames until the next idle condition are also discarded. If both the BAUD[MAEN1] and BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Idle match operation functions in the same way for both MATCH[MA1] or MATCH[MA2] fields.

- If only one of BAUD[MAEN1] and BAUD[MAEN2] is asserted, the first character after an idle line is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If BAUD[MAEN1] and BAUD[MAEN2] are asserted, the first character after an idle line is compared with both MATCH[MA1] or MATCH[MA2] fields and data is transferred only on a match with either of the fields.

51.4.4.2.6 Match On Match Off operation

Match on, match off operation is enabled when both BAUD[MAEN1] and BAUD[MAEN2] are set and BAUD[MATCFG] is equal to 10. In this function, a character received by the RXD pin that matches MATCH[MA1] is received and transferred to the receive buffer, and STAT[RDRF] is set. All subsequent characters are considered to be data and are also transferred to the receive data buffer, until a character is received that matches MATCH[MA2] field. The character that matches MATCH[MA2] and all following characters are discarded; this continues until another

character that matches MATCH[MA1] is received. If both the BAUD[MAEN1] and BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

NOTE

Match on, match off operation requires both BAUD[MAEN1] and BAUD[MAEN2] to be asserted.

51.4.4.3 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert RTS_B.

- RTS_B remains asserted until the transfer is complete, even if the transmitter is disabled midway through a data transfer. See Transceiver driver enable using RTS_B for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts RTS_B if the number of characters in the receiver data register is full or a start bit is detected that will cause the receiver data register to be full.
- The receiver asserts RTS_B when the number of characters in the receiver data register is not full and has not detected a start bit that will cause the receiver data register to be full. It is not affected if STAT[RDRF] is asserted.
- Even if RTS_B is deasserted, the receiver continues to receive characters until the receiver data buffer is overrun.
- If the receiver request-to-send functionality is disabled, the receiver RTS_B remains deasserted.

51.4.4.4 Infrared decoder

The infrared decoder converts the received character from the IrDA format to the NRZ format used by the receiver. It also has a OSR oversampling baud rate clock counter that filters noise and indicates when a 1 is received.

51.4.4.4.1 Start bit detection

When STAT[RXINV] is cleared, the first falling edge of the received character corresponds to the start bit. The infrared decoder resets its counter. At this time, the receiver also begins its start bit detection process. After the start bit is detected, the receiver synchronizes its bit times to this start bit time. For the rest of the character reception, the infrared decoder's counter and the receiver's bit time counter count independently from each other.

51.4.4.4.2 Noise filtering

Any further rising edges detected during the first half of the infrared decoder counter are ignored by the decoder. Any pulses less than one oversampling baud clock can be undetected by it regardless of whether it is seen in the first or second half of the count.

51.4.4.4.3 Low-bit detection

During the second half of the decoder count, a rising edge is decoded as a 0, which is sent to the receiver. The decoder counter is also reset.

51.4.4.4.4 High-bit detection

At OSR oversampling baud rate clocks after the previous rising edge, if a rising edge is not seen, then the decoder sends a 1 to the receiver.

If the next bit is a 0, which arrives late, then a low-bit is detected according to Low-bit detection. The value sent to the receiver is changed from 1 to a 0. Then, if a noise pulse occurs outside the receiver's bit time sampling period, then the delay of a 0 is not recorded as noise.

51.4.5 Additional LPUART functions

The following sections describe additional LPUART functions.

51.4.5.1 Data Modes

The LPUART transmitter and receiver can be configured to operate in 7-bit data mode by setting CTRL[M7], 9-bit data mode by setting the CTRL[M] or 10-bit data mode by setting BAUD[M10]. In 9-bit mode, there is a ninth data bit and in 10-bit mode, there is a tenth data bit. For the transmit data buffer, these bits are stored in CTRL[T8] and CTRL[T9]. For the receiver, these bits are held in CTRL[R8] and CTRL[R9]. They are also accessible via 16-bit or 32-bit accesses to the DATA register.

For coherent 8-bit writes to the transmit data buffer, write to CTRL[T8] and CTRL[T9] before writing to DATA[7:0]. For 16-bit and 32-bit writes to the DATA register, all 10 transmit bits are written to the transmit data buffer at the same time.

If the bit values to be transmitted as the ninth and tenth bit of a new character are the same as for the previous character, it is not necessary to write to CTRL[T8] and CTRL[T9] again. When data is transferred from the transmit data buffer to the transmit shifter, the value in CTRL[T8] and CTRL[T9] is copied at the same time data is transferred from DATA[7:0] to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. The 10-bit data mode is typically used with parity and address-mark wakeup so the ninth data bit can serve as the wakeup bit and the tenth bit as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

51.4.5.2 Idle length

An idle character is a character where the start bit, all data bits and stop bits are in the mark position. The CTRL[ILT] bit can be configured to start detecting an idle character from the previous start bit (any data bits and stop bits count towards the idle character detection) or from the previous stop bit.

The number of idle characters that must be received before an idle line condition is detected can also be configured using the CTRL[IDLECFG] field. This field configures the number of idle characters that must be received before the STAT[IDLE] flag is set, the STAT[RAF] flag is cleared and the DATA[IDLINE] flag is set with the next received character.

Idle-line wakeup and idle match operation are also affected by the CTRL[IDLECFG] field. When address match or match on/off operation is enabled, setting the STAT[RWUID] bit will cause any discarded characters to be treated as if they were idle characters.

51.4.5.3 Loop mode

When CTRL[LOOPS] is set, the CTRL[RSRC] bit chooses between loop mode (CTRL[RSRC] = 0) or single-wire mode (CTRL[RSRC] = 1). Loop mode is sometimes used to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the RXD pin is not used by the LPUART.

51.4.5.4 Single-wire operation

When CTRL[LOOPS] is set, the CTRL[RSRC] bit chooses between loop mode (CTRL[RSRC] = 0) or single-wire mode (CTRL[RSRC] = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TXD pin (the RXD pin is not used).

In single-wire mode, the CTRL[TXDIR] bit controls the direction of serial data on the TXD pin. When CTRL[TXDIR] is cleared, the TXD pin is an input to the receiver and the transmitter is temporarily disconnected from the TXD pin so an external device can send serial data to the receiver. When CTRL[TXDIR] is set, the TXD pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver cannot receive characters that are sent out by the transmitter.

51.4.6 Infrared interface

The LPUART provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the LPUART. The IrDA physical layer specification defines a half-duplex infrared communication link for exchanging data. The full standard includes data rates up to 16 Mbits/s. This module covers data rates only between 2.4 kbits/s and 115.2 kbits/s.

The LPUART has an infrared transmit encoder and receive decoder. The LPUART transmits serial bits of data that are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses are detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder, external from the LPUART. The narrow pulses are then stretched by the infrared receive decoder to get back to a serial bit stream to be received by the LPUART. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active high pulses.

The infrared submodule receives its clock sources from the LPUART. One of these two clocks are selected in the infrared submodule to generate either 1/OSR, 2/OSR, 3/OSR, or 4/OSR narrow pulses during transmission.

51.4.6.1 Infrared transmit encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the TXD signal. A narrow pulse is transmitted for a 0 bit and no pulse for a 1 bit. The narrow pulse is sent at the start of the bit with a duration of 1/OSR, 2/OSR, 3/OSR, or 4/OSR of a bit time. A narrow low pulse is transmitted for a 0 bit when CTRL[TXINV] is cleared, while a narrow high pulse is transmitted for a 0 bit when CTRL[TXINV] is set.

51.4.6.2 Infrared receive decoder

The infrared receive block converts data from the RXD signal to the receive shift register. A narrow pulse is expected for each 0 received and no pulse is expected for each 1 received. A narrow low pulse is expected for a 0 bit when STAT[RXINV] is cleared, while a narrow high pulse is expected for a 0 bit when STAT[RXINV] is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

51.4.7 Interrupts and status flags

The LPUART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty STAT[TDRE]) indicates when there is room in the transmit data buffer to write another transmit character to the DATA register. If the transmit interrupt enable CTRL[TIE]) bit is set, a hardware interrupt is requested when STAT[TDRE] is set. Transmit complete (STAT[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TXD at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (CTRL[TCIE]) bit is set, a hardware interrupt is requested when STAT[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the STAT[TDRE] and STAT[TC] status flags if the corresponding CTRL[TIE] or CTRL[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (STAT[RDRF] = 1), it gets the data from the receive data register by reading the DATA register. The STAT[RDRF] flag is cleared by reading the DATA register.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the RXD line remains idle for an extended period of time. IDLE is cleared by writing 1 to the STAT[IDLE] flag. After STAT[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set STAT[RDRF].

If the associated error was detected in the received character that caused STAT[RDRF] to be set, the error flags - noise flag (STAT[NF]), framing error (STAT[FE]), and parity error flag (STAT[PF]) - are set at the same time as STAT[RDRF]. These flags are not set in overrun cases.

If STAT[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (STAT[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2] then the STAT[MA1F] and/or STAT[MA2F] flags are set at the same time that STAT[RDRF] is set.

At any time, an active edge on the RXD serial data input pin causes the STAT[RXEDGIF] flag to set. The STAT[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (CTRL[RE] = 1).

51.4.8 Peripheral Triggers

The connection of the LPUART peripheral triggers with other peripherals are device specific.

51.4.8.1 Output Triggers

The LPUART generates three output triggers that can be connected to other peripherals on the device.

- The transmit word trigger asserts at the end of each transmitted word, it negates after one bit period.
- The receive word trigger asserts at the end of each received word that is written to the receive FIFO, for one oversampling clock period.
- The receive idle trigger asserts at when the idle flag would set, for one oversampling clock period.

51.4.8.2 Input Trigger

The LPUART supports one peripheral input trigger, that can be configured in one of the following ways.

- The input trigger can be connected in place of the CTS_B pin input. The input trigger must assert for longer than one bit clock period when the transmitter is idle with data to send to guarantee a new transmission.
- The input trigger can modulate the transmit data output (trigger is logically ANDed with the TXD output). The input trigger is expected to be generated from a PWM source with a period that is less than the bit clock frequency.
- The input trigger can be connected in place of the RXD pin input. The input trigger is expected to be generated from a receive data source, such as analog comparator or external pin.

Chapter 52
Low Power Serial Peripheral Interface (LPSPI)

52.1 Chip-specific LPSPI information

Table 52-1. Reference links to related information

Topic	Related module	Reference
Full description	LPSPI	LPSPI
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

52.1.1 LPSPI

The LPSPI is a low power Serial Peripheral Interface (SPI) module that supports an efficient interface to an SPI bus as a master and/or a slave.

- Can continue operating while the chip is in stop modes, if an appropriate clock is available
- Designed for low CPU overhead, with DMA offloading of FIFO register accesses

The following table shows the LPSPI configuration:

Table 52-2. LPSPI configuration

Parameter	Description
Name	LPSPI
Instances	4
Configurable features	• LPSPI0-1: 4TX/RX FIFO, 4 CS • LPSPI2-3: 16TX/RX FIFO, 4 CS
Interface speed	See the i.MX 7ULP Data Sheet for correct IO specification.
External I/O pins	LPSPI0-3: SIN, SOUT, SCK, PCSx. See the attached IOMUXC spreadsheet for details.

52.1.2 Transmit/command FIFO in master mode

When writing a command to the transmit/command FIFO containing bit TCR[TXMSK]=1 in order to initiate a half-duplex receive transmission with LPSPI in master mode, the user should not write another command into the FIFO before the command containing bit TCR[TXMSK]=1 has finished executing and bit TCR[TXMSK] has been cleared.

52.2 Introduction

The LPSPI is a low power Serial Peripheral Interface (SPI) module that supports an efficient interface to an SPI bus, either as a master and/or as a slave.

- The LPSPI is designed to use little CPU overhead, with DMA offloading of FIFO register accesses.
- The LPSPI can continue operating in stop modes, if an appropriate clock is available.
- The LPSPI supports DMA accesses and generates a DMA request.

The Serial Peripheral Interface bus (SPI) is a synchronous serial communication interface used in embedded systems, typically to perform short distance communications between microcontrollers and peripheral devices, on printed circuit boards. Typical applications include interfacing to Secure Digital cards and LCD displays.

- SPI devices communicate in full duplex mode using a master-slave scheme with a single master. This enables communication in both directions to happen simultaneously.
- Multiple slave devices are selected using individual slave select (SS) lines.
- The master device originates the frame for reading and writing.
- SPI accesses are always synchronous to the external clock.
- Each SPI can only assert one chip select at a time, but technically there can be multiple SPI slaves sharing the same chip select (in the form of a larger shift register). This requires:
  - the SDO of the master to connect to the SDI of the first slave,
  - the SDO of first slave to connect to the SDI of the next slave,
  - and the SDO of last slave to connect to the SDI of the LPSPI master.

  Note that some SPI modules use MISO/MOSI for the input/output data.

52.2.1 Features

The LPSPI supports:

- Word size = 32 bits
- Configurable clock polarity and clock phase
- Master operation supporting multiple peripheral chip selects (see chip-specific information)
- Slave operation
- Command/transmit FIFO of 16 words
- Receive FIFO of 16 words
- Flexible timing parameters in master mode, including SCK frequency and delays between PCS and SCK edges
- Support for full duplex transfers supporting 1-bit transmit and receive on each clock edge
- Support for half duplex transfers supporting 1-bit transmit or receive on each clock edge
- Support for half duplex transfers supporting 2-bit or 4-bit transmit or receive on each clock edge (master only)
- Host request input can be used to control the start time of an SPI bus transfer (master only)
- Receive data match logic supporting wakeup on data match

52.2.2 Block Diagram

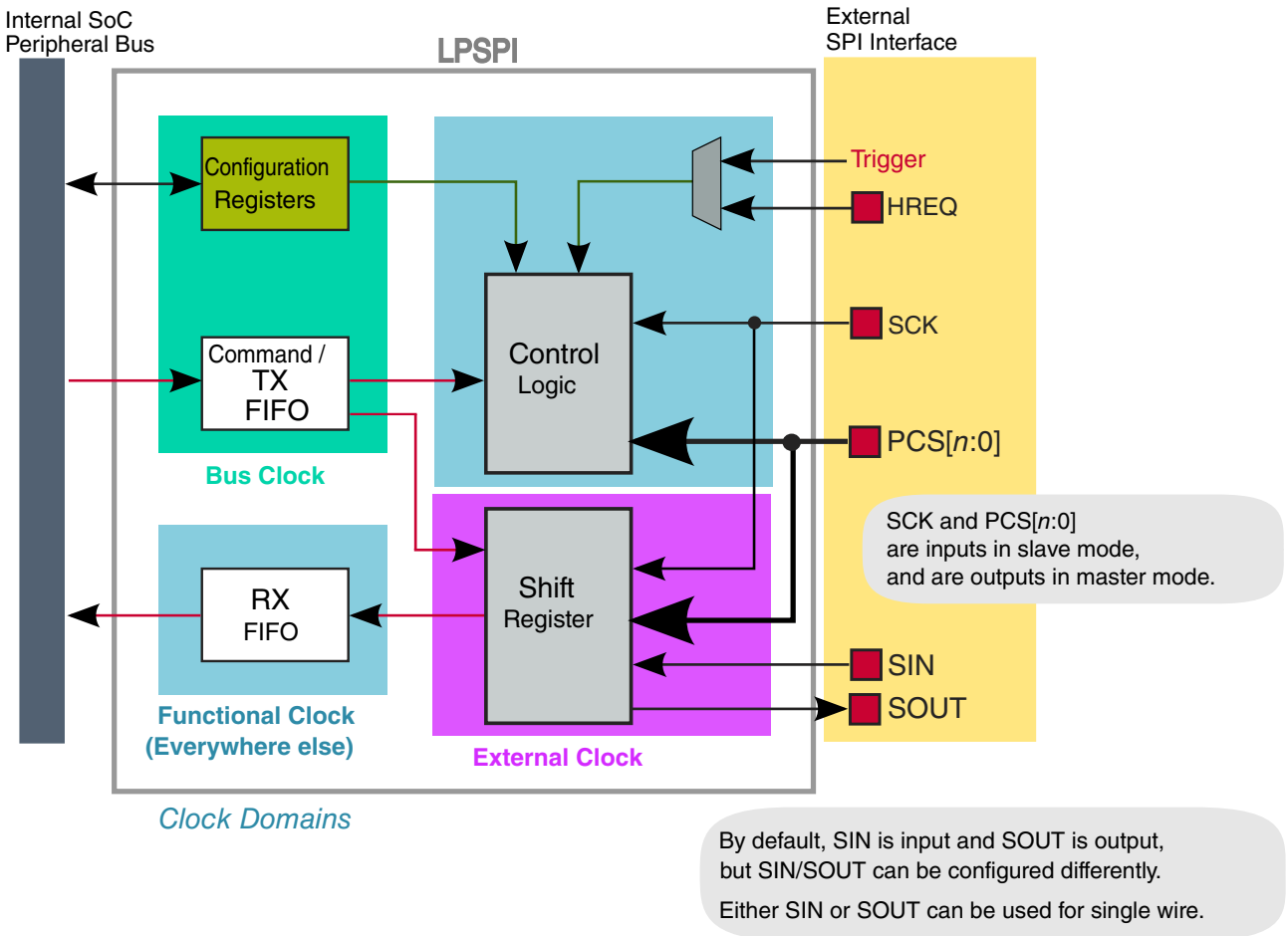


Figure 52-1. Block Diagram

52.2.3 Modes of operation

Table 52-3. Chip modes supported by the LPSPI module

Chip mode	LPSPI Operation
Run	Normal operation
Stop/Wait Stop	Can continue operating in stop mode if the Doze Enable bit (CR[DOZEN]) is clear and the LPSPI is using an external or internal clock source that remains operating during stop/wait modes.stop mode.
Low Power Stop (also called Low Leakage Stop or LLS)	Before entering low power stop mode, the LPSPI will wait for the current transfer to finish any pending operation, temporarily ignoring the Doze Enable (CR[DOZEN]) bit.
Debug (the core is in Debug/Halted mode)	Can continue operating in debug mode, if the Debug Enable bit (CR[DBGEN]) is set.

52.2.4 Signal Descriptions

Table 52-4. Signals

Signal	Name	Description	I/O
SCK	Serial clock	• Input in slave mode • Output in master mode	I/O
PCS[0]	Peripheral Chip Select	• Input in slave mode • Output in master mode	I/O
PCS[1] / HREQ	Peripheral Chip Select or Host Request	Host Request pin is selected when HREN=1 and HRSEL=0 • Input in either slave mode or when used as Host Request • Output in master mode	I/O
PCS[2] / DATA[2]	Peripheral Chip Select or data pin 2 during quad-data transfers	• Input in slave mode • Output in master mode • Input in quad-data receive transfers • Output in quad-data transmit transfers	I/O
PCS[3] / DATA[3]	Peripheral Chip Select or data pin 3 during quad-data transfers	• Input in slave mode • Output in master mode • Input in quad-data receive transfers • Output in quad-data transmit transfers	I/O
SOUT / DATA[0]	Serial Data Output	Can be configured as serial data input signal • Used as data pin 0 in quad-data transfers • Used as data pin 0 in dual-data transfers	I/O
SIN / DATA[1]	Serial Data Input	Can be configured as serial data output signal • Used as data pin 1 in quad-data transfers • Used as data pin 1 in dual-data transfers	I/O

52.3 Functional description

52.3.1 Clocking and resets

For device-specific clocking information, refer to the Clocking chapter.

Table 52-5. Clocks

LPSPI Functional clock	• The LPSPI functional clock is asynchronous to the bus clock. • If the LPSPI functional clock remains enabled in low power modes, then LPSPI can perform SPI bus transfers and low power wakeups, in both master and slave modes. • The LPSPI divides the functional clock by a prescaler; the resulting frequency must be at least 2 times faster than the SPI external clock frequency (LPSPI_SCK).

Table continues on the next page...

Table 52-5. Clocks (continued)

External clock	• The LPSPI shift register is clocked directly by the LPSPI_SCK clock.
	• How the LPSPI_SCK clock is generated or supplied depends upon the mode (master or slave):
	  • in master mode: the LPSPI_SCK clock is generated internally.
	  • in slave mode: the LPSPI_SCK clock is supplied externally.

Bus clock	The bus clock is only used for bus accesses to the LPSPI control and configuration registers. The bus clock frequency must be high enough to support the data bandwidth requirements of the LPSPI registers, including the FIFOs.

Table 52-6. Resets

Chip reset	Resets the LPSPI logic and registers to their default state.
Software reset	• Resets the LPSPI logic and registers to their default state, except for the Control Register. • The LPSPI software reset is in the Control Register CR[RST].
FIFO resets	• Resets the transmit/command FIFO and the receive FIFO. • Control Register CR[RTF](Reset Transmit FIFO) and CR[RRF] (Reset Receive FIFO) are write-only bits. • After being reset,a FIFO is empty.

NOTE

The entire PCSPOL field is not supported in every LPSPI module instance. Refer to the chip-specific information for LPSPI.

Field supported in	Field not supported in
LPSPI0_CFGR1	—
LPSPI1_CFGR1[13–8]	LPSPI1_CFGR1[15–14]
LPSPI2_CFGR1[11–8]	LPSPI2_CFGR1[15–12]
LPSPI3_CFGR1[11–8]	LPSPI3_CFGR1[15–12]
LPSPI4_CFGR1[11–8]	LPSPI4_CFGR1[15–12]
LPSPI5_CFGR1[11–8]	LPSPI5_CFGR1[15–12]
LPSPI6_CFGR1[9–8]	LPSPI6_CFGR1[15–10]
LPSPI7_CFGR1[9–8]	LPSPI7_CFGR1[15–10]
LPSPI8_CFGR1[9–8]	LPSPI8_CFGR1[15–10]
LPSPI9_CFGR1[9–8]	LPSPI9_CFGR1[15–10]

52.3.2 Master Mode

52.3.2.1 Transmit and Command FIFO commands

The transmit and command FIFO is a combined FIFO that includes both transmit data words and command words.

- Transmit data words are stored to the transmit/command FIFO, by writing the Transmit Data Register (TDR).
- Command words are stored to the transmit/command FIFO, by writing the Transmit Command Register (TCR).

When a command word is at the top of the transmit/command FIFO, the actions that can occur depend upon whether the LPSPI module is either busy or between frames, the Continuous Transfer bit (TCR[CONT]), and the Continuing Command bit (TCR[CONTC]).

Table 52-7. Possible actions when a command word is at the top of the transmit/command FIFO

Condition	Action
If the LPSPI is between frames	then the command word is pulled from the FIFO, and that command word controls all subsequent transfers.
If LPSPI is busy and the Continuous Transfer bit (TCR[CONT]) is set or cleared and the Continuing Command bit (TCR[CONTC]) is cleared	then the SPI frame will complete at the end of the existing word, ignoring the FRAMESZ configuration. The command word is then pulled from the FIFO and that command word controls all subsequent transfers (or until the next update to the command word). Note that a command word with CONTC=0 will always terminate (stop) the existing transfer regardless of the previous CONT value.
If the LPSPI is busy and the existing Continuous Transfer bit (TCR[CONT]) is set or cleared and the new Continuing Command bit (TCR[CONTC]) value is set	then the command word must be updated at the frame boundary. The command word is pulled from the FIFO during the last LPSPI_SCK pulse of the existing frame (based on the FRAMESZ configuration), and the frame continues using the new command value for the rest of the frame (or until the next update to the command word). When the Continuing Command bit (TCR[CONTC]) is set, only the lower 24-bits of the command word are updated. If the command word is updated at a word boundary, then the transfer halts (stops) after that word.

NOTE

About the Continuous Transfer bit (CONT) and Continuing Command bit (CONTC):

- Continuous Transfer bit (CONT)=1 is used to keep PCS asserted at end of frame, allowing the transfer to continue.
- Continuing Command bit (CONTC)=1 is used to indicate that this command word should not terminate the existing frame, and the transfer can continue using the new command word.

The Continuing Command bit (CONTC)=1 is restricted in the sense that the new command must load on a frame boundary, and the only way for a transfer to continue from a frame boundary, is when the previous command has the Continuous Transfer bit (CONT)=1.

The current state of the existing command word can be read by reading the Transmit Command Register (TCR). It requires at least 3 LPSPI functional clock cycles for the transmit command register to update after the transmit command register is written (assuming an empty FIFO) and the LPSPI must be enabled (Module Enable CR[MEN] bit is set).

Writing the transmit command register does not initiate a SPI bus transfer, unless the Transmit Data Mask (TCR[TXMSK]) bit is set. When the Transmit Data Mask bit is set, a new command word will not be loaded until the end of the existing frame (based on FRAMESZ configuration); at the end of the transfer, the TXMSK bit will be cleared.

In master mode, the LPSPI command word in the Transmit Command Register (TCR) controls SPI attributes (using bits and fields in registers).

Table 52-8. LPSPI Command Word in Master Mode

Transmit Command Register (TCR)		Description	Can this bit/ field be modified during a data transfer?
Bit/Field	Name		
CPOL	Clock Polarity	Configures the polarity of the LPSPI_SCK pin. Any change of CPOL value will cause a transition on the LPSPI_SCK pin.	N
CPHA	Clock Phase	Configures the clock phase of the transfer.	N
PRESCALE	Prescaler Value	Configures a prescaler used to divide the LPSPI functional clock, to generate the timing parameters of the SPI bus transfer. Changing PRESCALE in conjunction with PCS enables the LPSPI module to connect to different slave devices at different frequencies.	N
PCS	Peripheral Chip Select	Configures which LPSPI_PCS asserts for the transfer; the polarity of LPSPI_PCS is static and configured by PCSPOL. If PCSCFG is set, then PCS[3:2] should not be selected.	N
LSBF	LSB First	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.	Y
BYSW	Byte Swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.	Y
CONT	Continuous Transfer	Configures for a continuous transfer that keeps PCS asserted between frames (as configured by FRAMESZ). A new command word is required to cause PCS to negate. Also supports changing the command word at the frame size boundaries.	Y

Table continues on the next page...

Table 52-8. LPSPI Command Word in Master Mode (continued)

Transmit Command Register (TCR)		Description	Can this bit/ field be modified during a data transfer?
Bit/Field	Name		
CONTC	Continuing Command	Indicates that this is a new command word for the existing continuous transfer. The CONTC bit when set must only be written to the transmit/ command FIFO on a frame bounday.	Y
RXMSK	Receive Data Mask	Masks the receive data and does not store the masked receive data to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.	Y
TXMSK	Transmit Data Mask	Masks the transmit data, so that masked transmit data is not pulled from transmit FIFO, and the output data pin is tristated (unless configured by Output Config CFGR1[OUTCFG]). Useful for half-duplex transfers.	Y
WIDTH	Transfer Width	Configures the number of bits shifted on each LPSPI_SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit transfers are useful for interfacing to QuadSPI memory devices, and only support half-duplex data formats, and at least one bit (Transmit Data Mask (TCR[TXMSK] or Receive Data Mask TCR[RXMSK]) must also be set.	Y
FRAMESZ	Frame Size	Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits. • The minimum word size is 2 bits; a frame size of 33 bits (or similar) is not supported. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32-bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits. For example, a 72-bit transfer will consist of 3 words: the 1st and 2nd words are 32-bit, and the 3rd word is 8-bit.	Y

SPI bus transfers:

- The LPSPI initiates a SPI bus transfer when:
  - data is written to the transmit FIFO
  - and the HREQ pin is asserted (or disabled)
  - and the LPSPI is enabled
- To perform the SPI bus transfer, LPSPI uses the attributes configured in the Transmit Command Register (TCR) and uses the timing parameters in the Clock Configuration Register (CCR).

- The SPI bus transfer ends after the FRAMESZ configuration is reached, or at the end of a word when a new transmit command word is at the top of the transmit/command FIFO.
- The HREQ input is only checked on the next time that the LPSPI goes idle (the LPSPI completes the current transfer and the Transmit Command Register (TCR) is empty).

Circular FIFO: The transmit/command FIFO also supports a Circular FIFO feature, which enables the LPSPI master to (periodically) repeat a short data transfer that can fit within the transmit/command FIFO, without requiring additional FIFO accesses. When the circular FIFO is enabled, the current state of the FIFO read pointer is saved and the status flags do not update. After the transmit/command FIFO is considered empty and the LPSPI is idle, the FIFO read pointer is restored with the saved version, so the contents of the transmit/command FIFO are not permanently pulled from the FIFO while circular FIFO mode is enabled.

52.3.2.2 Receive FIFO and Data Match

The receive FIFO is used to store receive data during SPI bus transfers. When the Receive Data Mask TCR[RXMSK] bit is set, the receive data is discarded instead of being stored in the receive FIFO.

- The receive data is written to the receive FIFO at the end of the frame.
- During a multiple word or continuous transfer, the receive data is also written to the receive FIFO at the same time as the new transmit data is read from the transmit FIFO.
- During a continuous transfer, if the transmit FIFO is empty, then the receive data is only written to the receive FIFO after the transmit FIFO is written or after the Transmit Command Register (TCR) is written to end the frame.

Receive data supports a receive data match function that can match received data against one of two words or against a masked data word. The receive data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded due to the Receive Data Mask TCR[RXMSK] bit cannot cause the data match to set, and will delay the receive data match on the first received data word, until all discarded data is received.
- The receiver data match function can also be configured to discard all received data until a data match is detected, using the Receive Data Match Only CFGR0[RDMO] bit.
- After a receive data match, to allow all subsequent data to be received, first clear the Receive Data Match Only CFGR0[RDMO] bit, then clear the Data Match Flag SR[DMF] bit.

52.3.2.3 Timing Parameters

The timing parameters that are used for all SPI bus transfers are relative to the LPSPI functional clock divided by the PRESCALE configuration. Although the Clock Configuration Register (CCR) cannot be changed when the LPSPI module is busy, to support interfacing to different slave devices at different frequencies, the Prescaler Value TCR[PRESCALE] configuration can be changed between SPI bus transfers using the Transmit Command Register (TCR).

Table 52-9. LPSPI Timing Parameters

Clock Configuration Register (CCR)		Description	Min	Max
Bit/Field	Name			
SCKDIV	SCK Divider	Configures the LPSPI_SCK clock period to (SCKDIV +2) cycles. When SCK Divider is configured to an odd number of cycles, the first half of the LPSPI_SCK cycle is one cycle longer than the second half of the LPSPI_SCK cycle.	0 (2 cycles)	255 (257 cycles)
DBT	Delay Between Transfers	Configures the minimum delay between PCS negation and the next PCS assertion to (DBT + 2) cycles. When the command word is updated between transfers, there is a minimum of (DBT/2)+1 cycles between the command word update and any change on LPSPI_PCS pins.	0 (2 cycles)	255 (257 cycles)
DBT	Delay Between Transfers	Configures the delay during a continuous transfer between the last SCK edge of a frame and the first SCK edge of the continuing frame to (DBT + 1) cycles. This is useful when the external slave requires a large delay between different words of an SPI bus transfer.	0 (1 cycle)	255 (256 cycles)
PCSSCK	PCS-to-SCK Delay	Configures the minimum delay between PCS assertion and the first SCK edge to (PCSSCK + 1) cycles.	0 (1 cycle)	255 (256 cycles)
SCKPCS	SCK-to-PCS Delay	Configures the minimum delay between the last SCK edge and the PCS assertion to (SCKPCS + 1) cycles.	0 (1 cycle)	255 (256 cycles)

52.3.2.4 Pin Configuration

- To swap directions or support half-duplex transfers on the same pin, the LPSPI_SIN and LPSPI_SOUT pins can be configured using the Pin Configuration CFGR1[PINCFG].
- To determine if an output data pin (like LPSPI_SOUT) will tristate when the LPSPI_PCS is negated, or if the output data pin will simply retain the last value, use the Output Config CFGR1[OUTCFG].

- When configuring for half-duplex transfers using the same data pin in single-bit transfer mode, or when configuring any transfer in 2-bit and 4-bit transfer modes, the output data pins must be configured to tristate when LPSPI_PCS is negated; use the Output Config CFGR1[OUTCFG].
- When performing quad-data transfers, the Peripheral Chip Select Configuration CFGR1[PCSCFG] must be enabled. The Peripheral Chip Select Configuration CFGR1[PCSCFG] is also used to disable LPSPI_PCS[3:2] functions.

52.3.2.5 Clock Loopback

The LPSPI master can be configured to use 1 of 2 clocks to sample the input data (for example, LPSPI_SIN):

- either the LPSPI_SCK output clock directly
- or a delayed version of the LPSPI_SCK output clock

The delayed version of the LPSPI_SCK is delayed by the LPSPI_SCK pin output delay, plus the LPSPI_SCK pin input delay, and is configured by setting CFGR1[SAMPLE]. Enabling the loopback version of the LPSPI_SCK can improve the setup time of the input data from the slave.

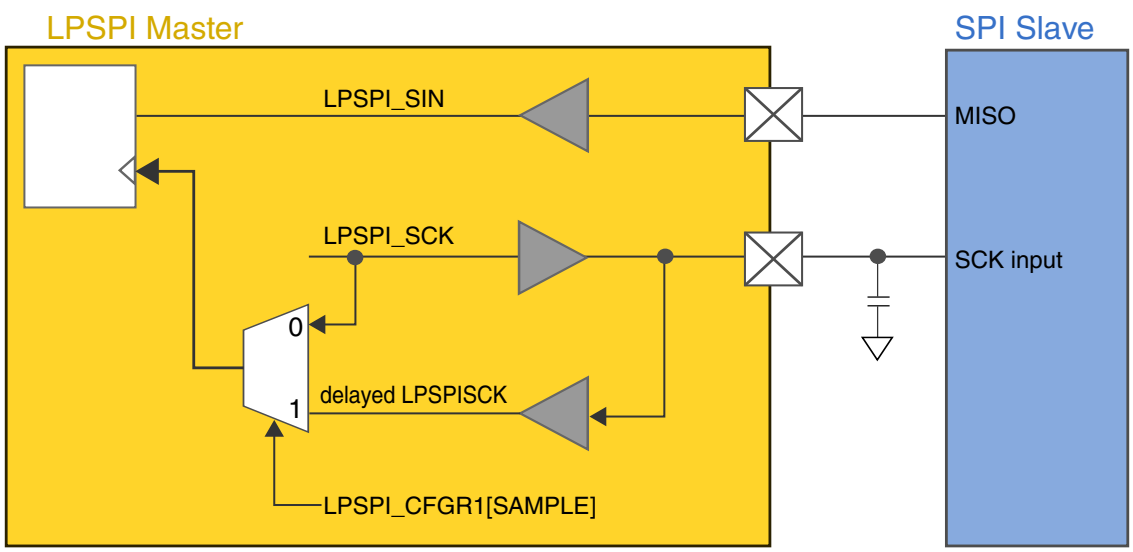


Figure 52-2. Clock Loopback

See the device datasheet for the specific input setup time in master loopback mode.

52.3.3 Slave Mode

LPSPI slave mode:

- uses the same shift register and logic that master mode uses
- does not use the Clock Configuration Register (CCR)
- during SPI bus transfers, requires that the Transmit Command Register (TCR) remain static (unchanging)

52.3.3.1 Transmit and Command FIFO commands

Before enabling the LPSPI in slave mode, the Transmit Command Register (TCR) should be initialized, although the Transmit Command Register register will not update until after the LPSPI is enabled. After being enabled, the Transmit Command Register should only be changed if the LPSPI is idle. In slave mode, the LPSPI command word in the Transmit Command Register (TCR) controls SPI attributes (using bits and fields in registers). Before the LPSPI_PCS input asserts, the transmit FIFO must be filled with transmit data, or the transmit error flag will set.

Table 52-10. LPSPI Command Word in Slave Mode

Transmit Command Register (TCR)		Description
Bit/Field	Name	
CPOL	Clock Polarity	Configures the polarity of the external LPSPI_SCK input.
CPHA	Clock Phase	Configures the clock phase of transfer.
PRESCALE	Prescaler Value	Configures the LPSPI functional clock prescaler.
PCS	Peripheral Chip Select	Configures which LPSPI_PCS is used, the polarity of LPSPI_PCS is static and configured by PCSPOL. If PCSCFG is set, then PCS[3:2] should not be selected.
LSBF	LSB First	Configures if LSB (bit 0) or MSB (bit 31 for a 32-bit word) is transmitted or received first.
BYSW	Byte Swap	Enables byte swap on each 32-bit word when transmitting and receiving data. Byte swapping can be useful when interfacing to devices that organize data as big-endian.
CONT	Continuous Transfer	When Continuous Transfer is set, only the first FRAMSZ bits will be transmitted/ received by the LPSPI module.
CONTC	Continuing Command	CONTC bit is reserved (in slave mode).
RXMSK	Receive Data Mask	Masks the receive data and does not store the masked receive data to the receive FIFO or perform receive data matching. Useful for half-duplex transfers or to configure which fields are compared during receive data matching.
TXMSK	Transmit Data Mask	Masks the transmit data, so that the masked transmit data is not pulled from transmit FIFO, and the output data pin is tristated (unless configured by Output Config CFGR1[OUTCFG]). Useful for half-duplex transfers.
WIDTH	Transfer Width	Configures the number of bits shifted on each LPSPI_SCK pulse. • 1-bit transfers support traditional SPI bus transfers in either half-duplex or full-duplex data formats. • 2-bit and 4-bit transfers are useful for interfacing to QuadSPI memory devices, and only support half-duplex data formats, and at least one bit (Transmit Data Mask TCR[TXMSK] or Receive Data Mask (TCR[RXMSK]) must also be set.
FRAMESZ	Frame Size	Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits.

Table 52-10. LPSPI Command Word in Slave Mode

Transmit Command Register (TCR)		Description
Bit/Field	Name	
		• The minimum word size is 2 bits; a frame size of 33 bits (or similar) is not supported. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32-bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits. For example, a 72-bit transfer will consist of 3 words: the 1st and 2nd words are 32-bit, and the 3rd word is 8-bit.

52.3.3.2 Receive FIFO and Data Match

The receive FIFO is used to store received data during SPI bus transfers. When the Receive Data Mask TCR[RXMSK] is set, the received data is discarded (instead of storing the received data in the receive FIFO).

Receive data supports a receive data match function that can match received data against one of two words or against a masked data word. The data match function can also be configured to compare only the first one or two received data words since the start of the frame.

- Received data that is already discarded (because the Receive Data Mask TCR[RXMSK] is set) cannot cause the data match to set, and will delay the match on the first received data word, until all discarded data is received.
- The receiver match function can also be configured to discard all received data until a data match is detected, using the Receive Data Match Only CFGR0[RDMO] bit.
- After a receive data match, to allow all subsequent data to be received, first clear the Receive Data Match Only CFGR0[RDMO] bit, then clear the Data Match Flag SR[DMF] bit.

52.3.3.3 Clocked Interface

The LPSPI module supports interfacing to external masters that provide only clock and data pins (LPSPI_PCS is not required). This interface requires:

- using Clock Phase TCR[CPHA] = 1 (data is changed on the leading edge of SCK and captured on the following edge)

- configuring the LPSPI_PCS input to be always asserted (configure the Peripheral Chip Select Polarity CFGR1[PCSPOL[n]] = 1). For example, to configure PCS[0] to be always asserted, set PCSPOL[0] = 1, and don't configure PCS[0] in the pin muxing.
- setting the Automatic PCS CFGR1[AUTOPCS] bit = 1 (Automatic PCS generation is enabled). When Automatic PCS generation (AUTOPCS) is set, a minimum of 4 LPSPI functional clock cycles (divided by PRESCALE configuration) is required between the last LPSPI_SCK edge of one word and the first LPSPI_SCK edge of the next word.

52.3.4 Interrupts and DMA Requests

The next table lists the slave mode sources (status flags) that can generate LPSPI interrupts and LPSPI slave transmit/receive DMA requests.

Table 52-11. LPSPI Interrupts and DMA Requests

Status Register (SR)		Description	Can generate		
Status Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
TDF	Transmit Data Flag	Data can be written to transmit FIFO, as configured by the Transmit FIFO Watermark FCR[TXWATER]	Y	TX	Y
RDF	Receive Data Flag	Data can be read from the receive FIFO, as configured by the Receive FIFO Watermark FCR[RXWATER]	Y	RX	Y
WCF	Word Complete Flag	Word is complete, the last bit of the word has been sampled	Y	N	Y
FCF	Frame Complete Flag	Frame is complete, and PCS has negated	Y	RX	Y
TCF	Transfer Complete Flag	Transfer is complete, PCS has negated, and the transmit/command FIFO is empty	Y	N	Y
TEF	Transmit Error Flag	Indicates a transmit/command FIFO underrun. In master mode when the No Stall CFGR1[NOSTALL] bit is clear (0, transfers will stall when transmit FIFO is empty or receive FIFO is full), the Transmit Error Flag bit cannot set.	Y	N	Y
REF	Receive Error Flag	Receive error flag, indicates a receive FIFO overflow. In master mode when the No Stall CFGR1[NOSTALL] bit is clear (0, transfers will stall when transmit FIFO is empty or receive FIFO is full), the Receive Error Flag bit cannot set.	Y	N	Y

Table continues on the next page...

Table 52-11. LPSPI Interrupts and DMA Requests (continued)

Status Register (SR)		Description	Can generate		
Status Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
DMF	Data Match Flag	Indicates that the received data has matched the configured data match value	Y	N	Y
MBF	Module Busy Flag	LPSPI is busy performing a SPI bus transfer	N	N	N

52.3.5 Peripheral Triggers

The connection of the LPSPI peripheral triggers to other peripherals depend upon the specific device being used.

Table 52-12. LPSPI Triggers

Trigger	Description	
Frame Output Trigger	The frame output trigger: • asserts at the end of each frame (when PCS negates) • remains asserted until PCS next asserts	The LPSPI generates 2 output triggers that can be connected to other peripherals on the device.
Word Output Trigger	The word output trigger: • asserts at the end of each received word • remains asserted for 1 LPSPI_SCK period	
Input Trigger	To control the start of a LPSPI bus transfer, the LPSPI input trigger can be selected instead of the LPSPI_HREQ input. • The LPSPI input trigger is synchronized, and must assert for at least 2 cycles of the LPSPI functional clock divided by the PRESCALE configuration, so that the input trigger can be detected. • When the LPSPI module is busy, the LPSPI_HREQ input (and therefore the LPSPI input trigger) is ignored .	

52.4 Memory Map and Registers

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. This module will not check if programmed values in the registers are correct; the application software must ensure that valid programmed values are being written.

52.4.1 LPSPI register descriptions

52.4.1.1 LPSPI Memory map

LPSPI0 base address: 4103_8000h

LPSPI1 base address: 4103_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0101_0004h
4h	Parameter Register (PARAM)	32	RO	0000_0202h
10h	Control Register (CR)	32	RW	0000_0000h
14h	Status Register (SR)	32	W1C	0000_0001h
18h	Interrupt Enable Register (IER)	32	RW	0000_0000h
1Ch	DMA Enable Register (DER)	32	RW	0000_0000h
20h	Configuration Register 0 (CFGR0)	32	RW	0000_0000h
24h	Configuration Register 1 (CFGR1)	32	RW	0000_0000h
30h	Data Match Register 0 (DMR0)	32	RW	0000_0000h
34h	Data Match Register 1 (DMR1)	32	RW	0000_0000h
40h	Clock Configuration Register (CCR)	32	RW	0000_0000h
58h	FIFO Control Register (FCR)	32	RW	0000_0000h
5Ch	FIFO Status Register (FSR)	32	RO	0000_0000h
60h	Transmit Command Register (TCR)	32	RW	0000_001Fh
64h	Transmit Data Register (TDR)	32	WO	0000_0000h
70h	Receive Status Register (RSR)	32	RO	0000_0002h
74h	Receive Data Register (RDR)	32	RO	0000_0000h

52.4.1.2 Version ID Register (VERID)

52.4.1.2.1 Offset

Register	Offset
VERID	0h

52.4.1.2.2 Function

Contains version numbers for the module design and feature set.

52.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

52.4.1.2.4 Fields

Field	Function
31-24	Major Version Number
MAJOR	Returns the major version number for the module specification. Read-only field.
23-16	Minor Version Number
MINOR	Returns the minor version number for the module specification. Read-only field.
15-0	Module Identification Number
FEATURE	Returns the feature set number. Read-only field. 0000000000000100b - Standard feature set supporting a 32-bit shift register.

52.4.1.3 Parameter Register (PARAM)

52.4.1.3.1 Offset

Register	Offset
PARAM	4h

52.4.1.3.2 Function

Contains parameter values that were implemented in the module.

52.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

52.4.1.3.4 Fields

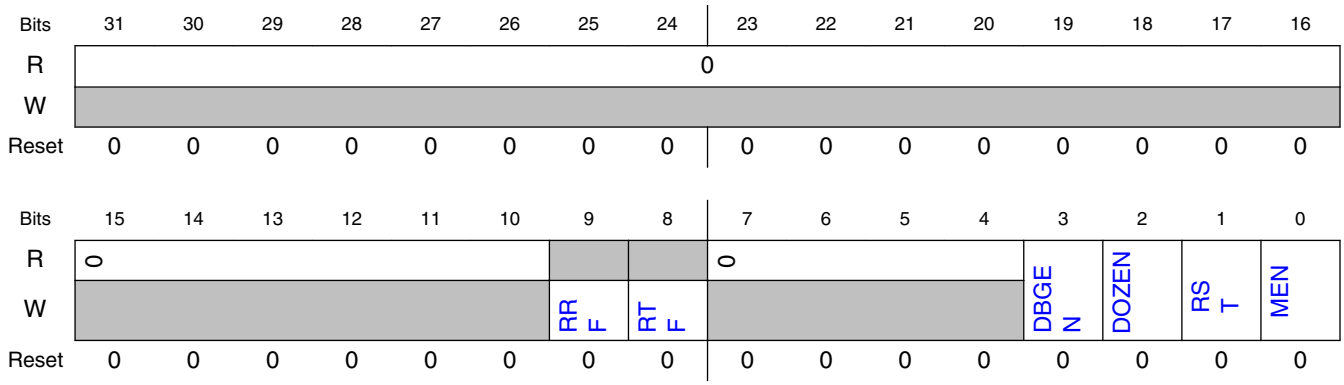
Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size Sets the maximum number of words in the receive FIFO, which is 2^{RXFIFO}
7-0 TXFIFO	Transmit FIFO Size Sets the maximum number of words in the transmit FIFO, which is 2^{TXFIFO}

52.4.1.4 Control Register (CR)

52.4.1.4.1 Offset

Register	Offset
CR	10h

52.4.1.4.2 Diagram



52.4.1.4.3 Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO 0b - No effect 1b - Receive FIFO is reset
8 RTF	Reset Transmit FIFO 0b - No effect 1b - Transmit FIFO is reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables or disables the LPSPI module in debug mode. Debug Enable bit should be updated only when the LPSPI module is disabled. 0b - LPSPI module is disabled in debug mode 1b - LPSPI module is enabled in debug mode
2 DOZEN	Doze Mode Enable Enables or disables the LPSPI module in Doze mode. Doze Mode Enable bit should be updated only when the LPSPI module is disabled. 0b - LPSPI module is enabled in Doze mode 1b - LPSPI module is disabled in Doze mode
1 RST	Software Reset Reset all internal logic and registers, except the Control Register. RST remains set until cleared by software. 0b - Module is not reset 1b - Module is reset
0 MEN	Module Enable 0b - Module is disabled 1b - Module is enabled

52.4.1.5 Status Register (SR)

52.4.1.5.1 Offset

Register	Offset
SR	14h

52.4.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							MBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DMF	REF	TEF	TCF	FCF	WCF	0						RDF	TDF
W			W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

52.4.1.5.3 Fields

Field	Function
31-25 —	Reserved
24 MBF	Module Busy Flag 0b - LPSPI is idle 1b - LPSPI is busy
23-14 —	Reserved
13 DMF	Data Match Flag Indicates that the received data has matched the MATCH0 and/or MATCH1 fields (as configured by CFGR1[MATCFG], Configuration Register 1). 0b - Have not received matching data 1b - Have received matching data
12 REF	Receive Error Flag The Receive Error Flag will set when the Receiver FIFO overflows. When the Receive Error Flag is set, it is recommended to first end the transfer, empty the Receive FIFO, clear the Receive Error Flag and then restart the transfer from the beginning. 0b - Receive FIFO has not overflowed

Table continues on the next page...

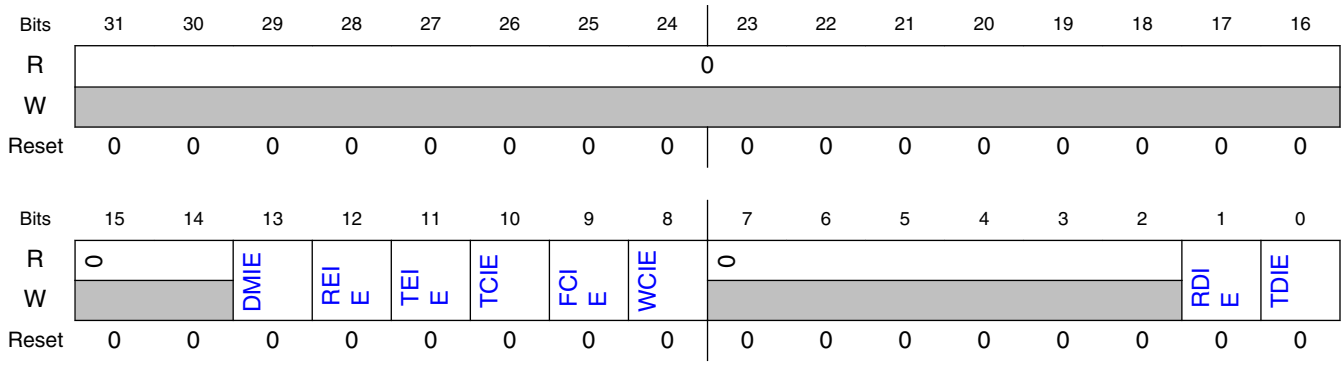
Field	Function
	1b - Receive FIFO has overflowed
11 TEF	Transmit Error Flag The Transmit Error Flag will set when the Transmit FIFO underruns. When the Transmit Error Flag is set, it is recommended to first end the transfer, clear the Transmit Error Flag and then restart the transfer from the beginning. 0b - Transmit FIFO underrun has not occurred 1b - Transmit FIFO underrun has occurred
10 TCF	Transfer Complete Flag In master mode when the LPSPI returns to idle state with the transmit FIFO empty, the Transfer Complete Flag will set. 0b - All transfers have not completed 1b - All transfers have completed
9 FCF	Frame Complete Flag The Frame Complete Flag will set at the end of each frame transfer, when the PCS negates. 0b - Frame transfer has not completed 1b - Frame transfer has completed
8 WCF	Word Complete Flag The Word Complete Flag will set when the last bit of a received word is sampled. 0b - Transfer of a received word has not yet completed 1b - Transfer of a received word has completed
7-2 —	Reserved
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than FCR[RXWATER] (FIFO Control Register) 0b - Receive Data is not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than FCR[TXWATER] (FIFO Control Register) 0b - Transmit data not requested 1b - Transmit data is requested

52.4.1.6 Interrupt Enable Register (IER)

52.4.1.6.1 Offset

Register	Offset
IER	18h

52.4.1.6.2 Diagram



52.4.1.6.3 Fields

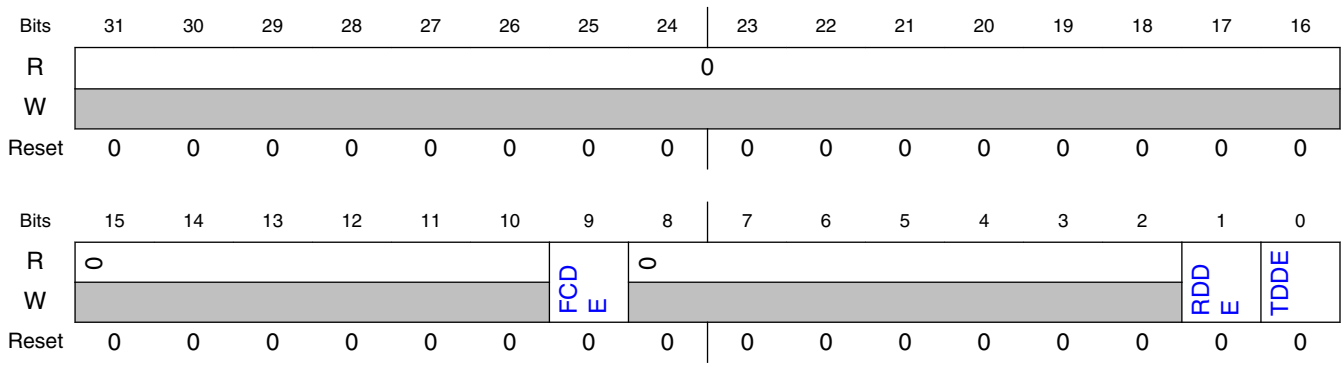
Field	Function
31-14 —	Reserved
13 DMIE	Data Match Interrupt Enable 0b - Disabled 1b - Enabled
12 REIE	Receive Error Interrupt Enable 0b - Disabled 1b - Enabled
11 TEIE	Transmit Error Interrupt Enable 0b - Disabled 1b - Enabled
10 TCIE	Transfer Complete Interrupt Enable 0b - Disabled 1b - Enabled
9 FCIE	Frame Complete Interrupt Enable 0b - Disabled 1b - Enabled
8 WCIE	Word Complete Interrupt Enable 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable 0b - Disabled 1b - Enabled
0 TDIE	Transmit Data Interrupt Enable 0b - Disabled 1b - Enabled

52.4.1.7 DMA Enable Register (DER)

52.4.1.7.1 Offset

Register	Offset
DER	1Ch

52.4.1.7.2 Diagram



52.4.1.7.3 Fields

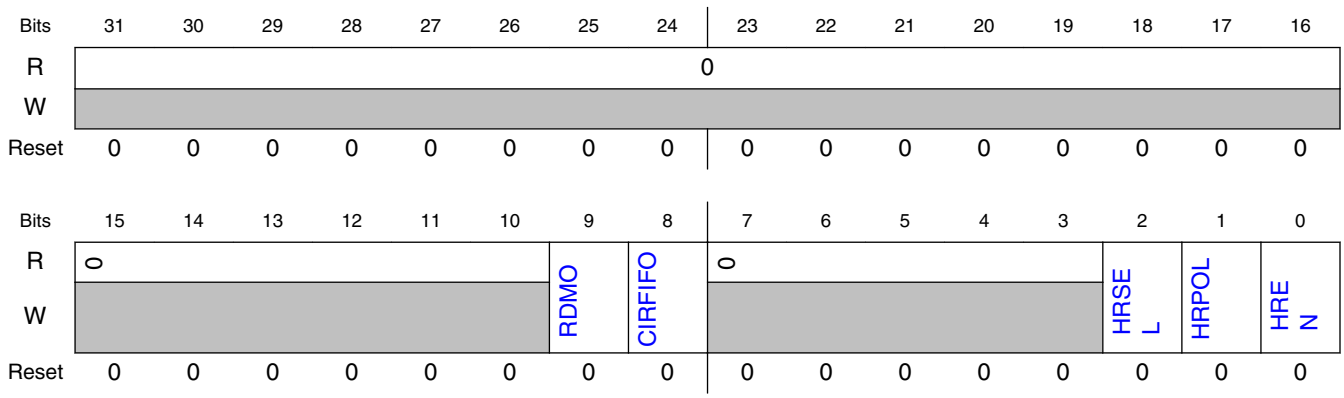
Field	Function
31-10 —	Reserved
9 FCDE	Frame Complete DMA Enable Enables DMA end-of-packet processing when set. After the last word of a frame is read from the receive data FIFO, reading the receive data FIFO will return an end-of-packet signal with the receive data forced to 0xFFFF_FFFF. This will continue until the DMA minor loop completes, and then the Frame Complete Flag will be cleared if the receive FIFO is empty or if the LPSPI is busy. 0b - DMA request is disabled 1b - DMA request is enabled
8-2 —	Reserved
1 RDDE	Receive Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled
0 TDDE	Transmit Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled

52.4.1.8 Configuration Register 0 (CFGR0)

52.4.1.8.1 Offset

Register	Offset
CFGR0	20h

52.4.1.8.2 Diagram



52.4.1.8.3 Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only When Receive Data Match Only is enabled, all received data that does not cause the Data Match Flag (SR[DMF]) to set is discarded. • Receive Data Match Only bit should be set when the LPSPI is idle and the Data Match Flag is clear • After the Data Match Flag (DMF) is set, the Receive Data Match Only bit configuration is ignored • When disabling RDMO and to ensure that no receive data is lost, before clearing the Data Match Flag, first clear Receive Data Match Only (RDMO) 0b - Received data is stored in the receive FIFO as in normal operations 1b - Received data is discarded unless the Data Match Flag (DMF) is set
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO will be emptied as it normally is, but after the LPSPI is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This restoring of the read pointer will cause the contents of the transmit FIFO to be cycled through repeatedly. 0b - Circular FIFO is disabled 1b - Circular FIFO is enabled
7-3	Reserved

Table continues on the next page...

Field	Function
—	
2 HRSEL	Host Request Select Selects the source of the host request input. When the host request function is enabled with the LPSPI_HREQ pin, the LPSPI_PCS[1] function is disabled. 0b - Host request input is the LPSPI_HREQ pin 1b - Host request input is the input trigger
1 HRPOL	Host Request Polarity Configures the polarity of the host request pin. 0b - Active low 1b - Active high
0 HREN	Host Request Enable When enabled in master mode, the LPSPI will only start a new SPI bus transfer if the host request input is asserted. When the LPSPI is busy, the host request input is ignored. 0b - Host request is disabled 1b - Host request is enabled

52.4.1.9 Configuration Register 1 (CFGR1)

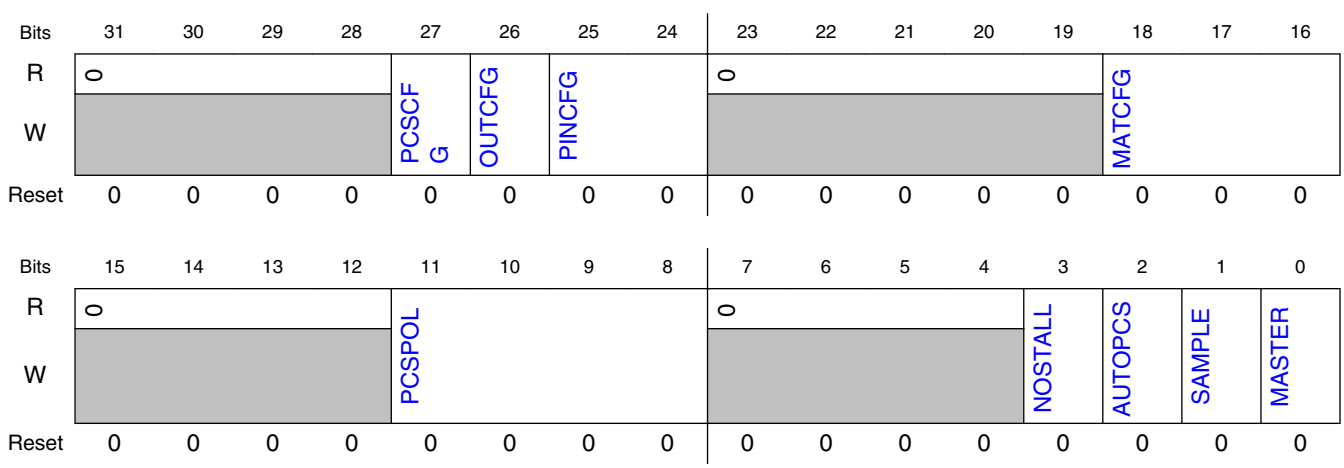
52.4.1.9.1 Offset

Register	Offset
CFGR1	24h

52.4.1.9.2 Function

The CFGR1 should only be written when the LPSPI is disabled.

52.4.1.9.3 Diagram



52.4.1.9.4 Fields

Field	Function
31-28 —	Reserved
27 PCSCFG	Peripheral Chip Select Configuration If performing 4-bit transfers, the Peripheral Chip Select Configuration bit must be set. 0b - PCS[3:2] are enabled 1b - PCS[3:2] are disabled
26 OUTCFG	Output Configuration Configures if the output data is tristated between accesses (LPSPI_PCS is negated). If performing 2-bit or 4-bit transfers, the Output Configuration bit must be set. 0b - Output data retains last value when chip select is negated 1b - Output data is tristated when chip select is negated
25-24 PINCFG	Pin Configuration Configures which pins are used for input and output data during 1-bit transfers. If performing 2-bit or 4-bit transfers, the Pin Configuration field must be 00. 00b - SIN is used for input data and SOUT is used for output data 01b - SIN is used for both input and output data 10b - SOUT is used for both input and output data 11b - SOUT is used for input data and SIN is used for output data
23-19 —	Reserved
18-16 MATCFG	Match Configuration Configures the condition that will cause the DMF to set. **NOTE:** *Syntax:* * is boolean AND, + is boolean OR 000b - Match is disabled 001b - Reserved 010b - 010b - Match is enabled, if 1st data word equals MATCH0 OR MATCH1, i.e., (1st data word = MATCH0 + MATCH1) 011b - 011b - Match is enabled, if any data word equals MATCH0 OR MATCH1, i.e., (any data word = MATCH0 + MATCH1) 100b - 100b - Match is enabled, if 1st data word equals MATCH0 AND 2nd data word equals MATCH1, i.e., [(1st data word = MATCH0) * (2nd data word = MATCH1)] 101b - 101b - Match is enabled, if any data word equals MATCH0 AND the next data word equals MATCH1, i.e., [(any data word = MATCH0) * (next data word = MATCH1)] 110b - 110b - Match is enabled, if (1st data word AND MATCH1) equals (MATCH0 AND MATCH1), i.e., [(1st data word * MATCH1) = (MATCH0 * MATCH1)] 111b - 111b - Match is enabled, if (any data word AND MATCH1) equals (MATCH0 AND MATCH1), i.e., [(any data word * MATCH1) = (MATCH0 * MATCH1)]
15-12 —	Reserved
11-8 PCSPOL	Peripheral Chip Select Polarity Configures the polarity of each Peripheral Chip Select pin. Each PCSPOL bit position (let's call it *n*) corresponds to a PCS[*n*] pin. For example, PCSPOL[0] is chip select 0 (at PCS[0] pin), and PCSPOL[1] is chip select 1 (at PCS[1] pin). If a PCSPOL bit:

Table continues on the next page...

Field	Function
	• =0, then the PCS[*n*] pin is active low. • =1, then the PCS[*n*] pin is active high. **NOTE:** The entire PCSPOL field is not fully supported in every LPSPI module instance. Refer to the chip-specific information for LPSPI.
7-4 —	Reserved
3 NOSTALL	No Stall In master mode, the LPSPI will stall transfers when the transmit FIFO is empty or when the receive FIFO is full, ensuring that no transmit FIFO underrun or receive FIFO overrun can occur. Setting the No Stall bit will disable this functionality. 0b - Transfers will stall when the transmit FIFO is empty or the receive FIFO is full 1b - Transfers will not stall, allowing transmit FIFO underruns or receive FIFO overruns to occur
2 AUTOPCS	Automatic PCS For correct operations, the LPSPI slave normally requires the PCS to negate between frames. Setting the Automatic PCS bit will cause the LPSPI to generate an internal PCS signal at the end of each transfer word when the Clock Phase bit TCR[CPHA] = 1. • When the Automatic PCS bit is set, the SCK must remain idle for at least 4 LPSPI functional clock cycles (divided by the Prescaler Value TCR[PRESCALE] configuration) between each word, to ensure correct operations • In master mode, the Automatic PCS bit is ignored 0b - Automatic PCS generation is disabled 1b - Automatic PCS generation is enabled
1 SAMPLE	Sample Point When set, the LPSPI master will sample the input data on a delayed LPSPI_SCK edge, which improves the setup time when sampling data. • The input data setup time in master mode with delayed LPSPI_SCK edge is equal to the input data setup time in slave mode • In slave mode, the SAMPLE bit is ignored 0b - Input data is sampled on SCK edge 1b - Input data is sampled on delayed SCK edge
0 MASTER	Master Mode Configures the LPSPI in master or slave mode. The Master Mode bit directly controls the direction of the LPSPI_SCK and LPCPI_PCS pins. 0b - Slave mode 1b - Master mode

52.4.1.10 Data Match Register 0 (DMR0)

52.4.1.10.1 Offset

Register	Offset
DMR0	30h

52.4.1.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.1.10.3 Fields

Field	Function
31-0 MATCH0	Match 0 Value When Receive Data Match Only (CFGR0[RDMO]) is enabled, the Match 0 Value is compared against the received data.

52.4.1.11 Data Match Register 1 (DMR1)

52.4.1.11.1 Offset

Register	Offset
DMR1	34h

52.4.1.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.1.11.3 Fields

Field	Function
31-0 MATCH1	Match 1 Value When Receive Data Match Only (CFGR0[RDMO]) is enabled, the Match 1 Value is compared against the received data.

52.4.1.12 Clock Configuration Register (CCR)

52.4.1.12.1 Offset

Register	Offset
CCR	40h

52.4.1.12.2 Function

The Clock Configuration Register is only used in master mode, and the Clock Configuration Register cannot be changed when the LPSPI is enabled.

52.4.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKPCS								PCSSCK							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.1.12.4 Fields

Field	Function
31-24 SCKPCS	SCK-to-PCS Delay In master mode: configures the delay from the last SCK edge to the PCS negation.

Table continues on the next page...

Field	Function
	• The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle.
23-16 PCSSCK	PCS-to-SCK Delay In master mode: configures the delay from the PCS assertion to the first SCK edge. • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle.

15-8
DBT
	Delay Between Transfers
	In master mode:
	• Configures the delay from the PCS negation to the next PCS assertion.
	  • The delay is equal to (DBT + 2) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration.
	  • The minimum delay is 2 cycles.
	  • Half of the delay occurs before PCS assertion and the other half of the delay occurs after PCS negation. If the command word is updated between 2 transfers, then the command word is updated half-way between the PCS negation of the last transfer and PCS assertion of the next transfer.
	  • The command word sets which PCS signal is used (of PCS[3:0]}, the polarity/phase of the SCK signal, and the Prescaler Value.
	• Configures the delay from the last SCK edge of a transfer word and the first SCK edge of the next transfer word, in a continuous transfer.
	  • The delay is equal to (DBT + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration.
	  • The minimum delay is 1 cycle.

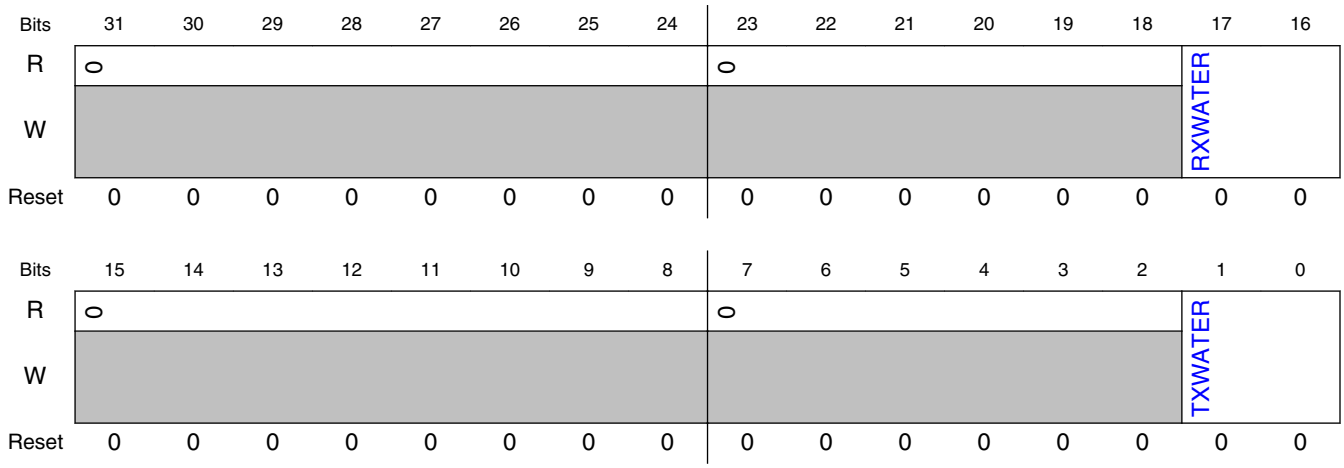
7-0 SCKDIV	SCK Divider In master mode, the SCK Divider configures the divide ratio of the SCK pin. • The SCK period is equal to (SCKDIV+2) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum SCK period is 2 cycles. • If the SCK period is an odd number of cycles, then the 1st half of the SCK period will be 1 cycle longer than the 2nd half of the SCK period.

52.4.1.13 FIFO Control Register (FCR)

52.4.1.13.1 Offset

Register	Offset
FCR	58h

52.4.1.13.2 Diagram



52.4.1.13.3 Fields

Field	Function
31-24 —	Reserved
23-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size will be truncated.
15-8 —	Reserved
7-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size will be truncated.

52.4.1.14 FIFO Status Register (FSR)

52.4.1.14.1 Offset

Register	Offset
FSR	5Ch

52.4.1.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0					RXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0					TXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.1.14.3 Fields

Field	Function
31-24 —	Reserved
23-19 —	Reserved
18-16 RXCOUNT	Receive FIFO Count Returns the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Returns the number of words currently stored in the transmit FIFO.

52.4.1.15 Transmit Command Register (TCR)

52.4.1.15.1 Offset

Register	Offset
TCR	60h

52.4.1.15.2 Function

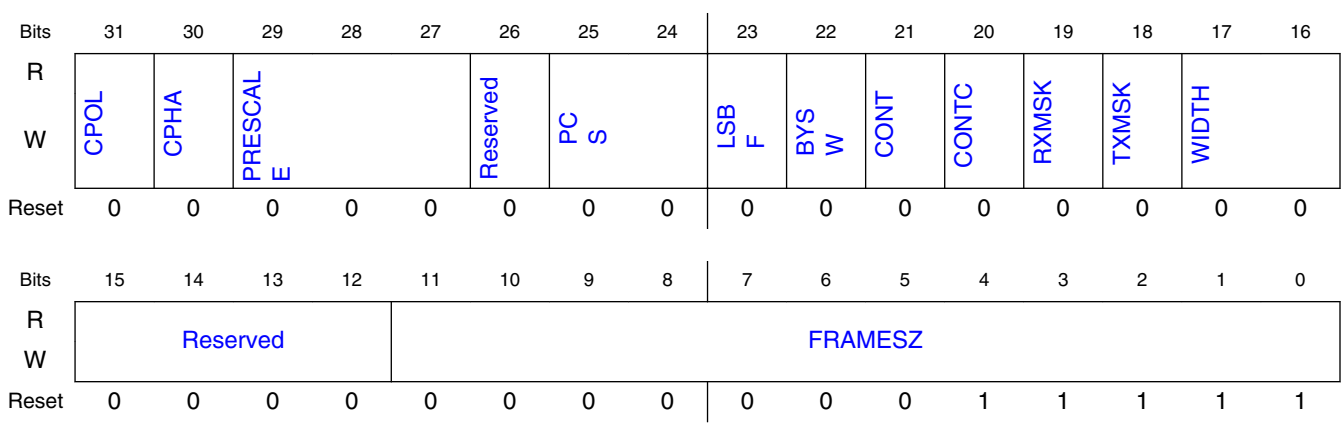
Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO, in the order that the data are written. The Command Register should only be written using 32-bit writes. Command Register writes will be tagged and cause the command register to update, after that entry reaches the top of the FIFO. This allows changes to the command word and the transmit data itself to be interleaved. Changing the command word will cause all subsequent SPI bus transfers to be performed using the new command word.

- **In master mode,** writing a new command word does not initiate a new transfer, unless TXMSK is set. Transfers are initiated by transmit data in the transmit FIFO, or by a new command word (with TXMSK set). Hardware will clear TXMSK when the LPSPI_PCS negates.
- **In master mode,** if the command word is changed before an existing frame has completed, then the existing frame will terminate and the command word will then update. The command word can be changed during a continuous transfer, if CONTC of the new command word is set and the command word is written on a frame size boundary.
- **In slave mode,** the command word should be changed only when the LPSPI is idle and there is no SPI bus transfer.

Avoid register reading problems: Reading the Transmit Command Register will return the current state of the command register. Reading the Transmit Command Register at the same time that the Transmit Command Register is loaded from the transmit FIFO, can return an incorrect Transmit Command Register value. It is recommended:

- to either read the Transmit Command Register when the transmit FIFO is empty,
- or to read the Transmit Command Register more than once and then compare the returned values.

52.4.1.15.3 Diagram



52.4.1.15.4 Fields

Field	Function
31 CPOL	Clock Polarity The Clock Polarity field is only updated between frames. 0b - The inactive state value of SCK is low 1b - The inactive state value of SCK is high
30 CPHA	Clock Phase The Clock Phase field is only updated between frames. 0b - Data is captured on the leading edge of SCK and changed on the following edge of SCK 1b - Data is changed on the leading edge of SCK and captured on the following edge of SCK
29-27 PRESCALE	Prescaler Value For all SPI bus transfers, the Prescaler value applied to the clock configuration register. The Prescaler Value field is only updated between frames. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128
26 —	Reserved
25-24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. The Peripheral Chip Select field is only updated between frames. **NOTE:** The entire PCS field is not fully supported in every LPSPI module instance. Refer to the chip-specific information for LPSPI. 00b - Transfer using LPSPI_PCS[0] 01b - Transfer using LPSPI_PCS[1] 10b - Transfer using LPSPI_PCS[2] 11b - Transfer using LPSPI_PCS[3]
23 LSBF	LSB First 0b - Data is transferred MSB first 1b - Data is transferred LSB first
22 BYSW	Byte Swap Byte swap will swap the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and for each received data word stored to the FIFO (or compared with match registers). 0b - Byte swap is disabled 1b - Byte swap is enabled
21 CONT	Continuous Transfer • In master mode, continuous transfer will keep the PCS asserted at the end of the frame size, until a command word is received that starts a new frame. • In slave mode, when continuous transfer is enabled, the LPSPI will only transmit the first FRAMESZ bits; after which the LPSPI will transmit received data (assuming a 32-bit shift register). 0b - Continuous transfer is disabled 1b - Continuous transfer is enabled

Table continues on the next page...

Field	Function
20 CONTC	Continuing Command In master mode, the Continuing Command bit allows the command word to be changed within a continuous transfer. • The initial command word must enable continuous transfer (CONT=1), • the continuing command must set this bit (CONTC=1), • and the continuing command word must be loaded on a frame size boundary. For example, if the continuous transfer has a frame size of 64-bits, then a continuing command word must be loaded on a 64-bit boundary. 0b - Command word for start of new transfer 1b - Command word for continuing transfer
19 RXMSK	Receive Data Mask When set, receive data is masked (receive data is not stored in receive FIFO). 0b - Normal transfer 1b - Receive data is masked
18 TXMSK	Transmit Data Mask When set, transmit data is masked (no data is loaded from transmit FIFO and output pin is tristated). In master mode, the Transmit Data Mask bit will initiate a new transfer which cannot be aborted by another command word; the Transmit Data Mask bit will be cleared by hardware at the end of the transfer. 0b - Normal transfer 1b - Mask transmit data
17-16 WIDTH	Transfer Width For 2-bit or 4-bit transfers, either Receive Data Mask (RXMSK) or Transmit Data Mask (TXMSK) must be set. 00b - 1 bit transfer 01b - 2 bit transfer 10b - 4 bit transfer 11b - Reserved
15-12 —	Reserved. Software should only write zero to this field.
11-0 FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits • The minimum word size is 2 bits; a frame size of 33 bits (or similar) is not supported. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32-bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits. For example, a 72-bit transfer will consist of 3 words: the 1st and 2nd words are 32 bits, and the 3rd word is 8 bits.

52.4.1.16 Transmit Data Register (TDR)

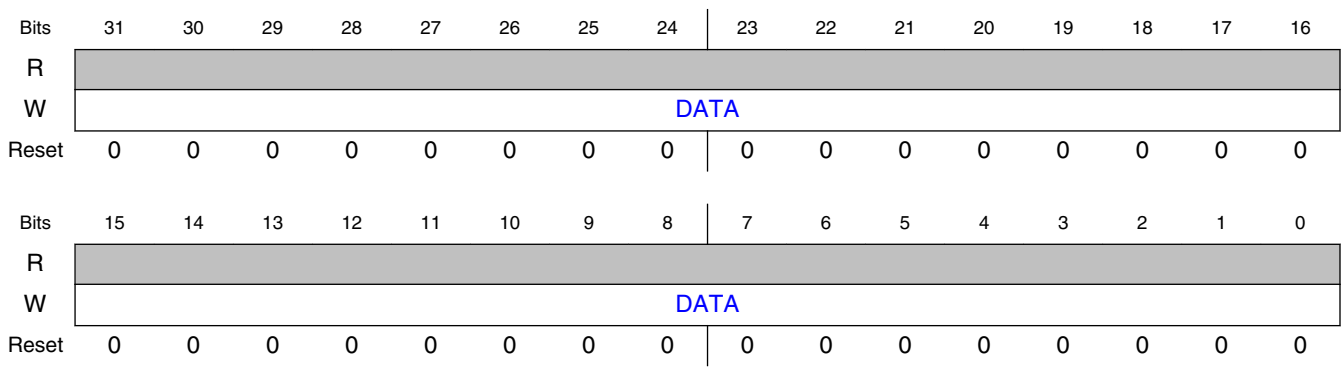
52.4.1.16.1 Offset

Register	Offset
TDR	64h

52.4.1.16.2 Function

Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO, in the order that it (the data) was written. The Data Register can be written using 32-bit, 16-bit or 8-bit writes, but each write will push data into the FIFO with zero pushed in unwritten bytes.

52.4.1.16.3 Diagram



52.4.1.16.4 Fields

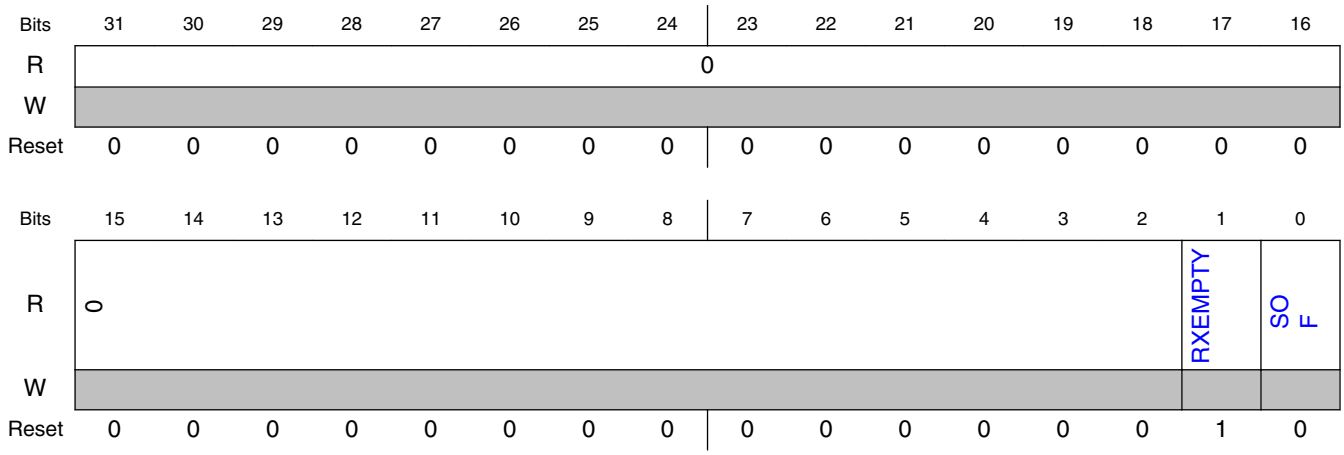
Field	Function
31-0 DATA	Transmit Data Both 8-bit and 16-bit writes of transmit data will zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8-bit and 16-bit writes (to 32 bits), means that the higher order (most significant) empty parts of the 8-bit and 16-bit writes are filled with zeroes.

52.4.1.17 Receive Status Register (RSR)

52.4.1.17.1 Offset

Register	Offset
RSR	70h

52.4.1.17.2 Diagram



52.4.1.17.3 Fields

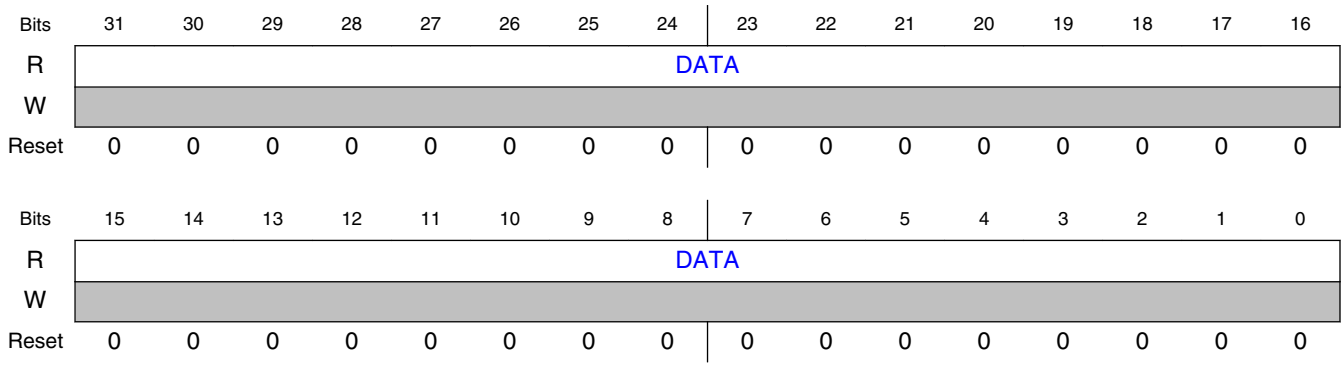
Field	Function
31-2 —	Reserved
1 RXEMPTY	RX FIFO Empty 0b - RX FIFO is not empty 1b - RX FIFO is empty
0 SOF	Start Of Frame Indicates that this is the first data word received after LPSPI_PCS assertion. 0b - Subsequent data word received after LPSPI_PCS assertion 1b - First data word received after LPSPI_PCS assertion

52.4.1.18 Receive Data Register (RDR)

52.4.1.18.1 Offset

Register	Offset
RDR	74h

52.4.1.18.2 Diagram



52.4.1.18.3 Fields

Field	Function
31-0 DATA	Receive Data

52.4.2 LPSPI register descriptions

52.4.2.1 LPSPI Memory map

LPSPI2 base address: 4029_0000h

LPSPI3 base address: 402A_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0101_0004h
4h	Parameter Register (PARAM)	32	RO	0000_0404h
10h	Control Register (CR)	32	RW	0000_0000h
14h	Status Register (SR)	32	W1C	0000_0001h
18h	Interrupt Enable Register (IER)	32	RW	0000_0000h
1Ch	DMA Enable Register (DER)	32	RW	0000_0000h
20h	Configuration Register 0 (CFGR0)	32	RW	0000_0000h
24h	Configuration Register 1 (CFGR1)	32	RW	0000_0000h
30h	Data Match Register 0 (DMR0)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
34h	Data Match Register 1 (DMR1)	32	RW	0000_0000h
40h	Clock Configuration Register (CCR)	32	RW	0000_0000h
58h	FIFO Control Register (FCR)	32	RW	0000_0000h
5Ch	FIFO Status Register (FSR)	32	RO	0000_0000h
60h	Transmit Command Register (TCR)	32	RW	0000_001Fh
64h	Transmit Data Register (TDR)	32	WO	0000_0000h
70h	Receive Status Register (RSR)	32	RO	0000_0002h
74h	Receive Data Register (RDR)	32	RO	0000_0000h

52.4.2.2 Version ID Register (VERID)

52.4.2.2.1 Offset

Register	Offset
VERID	0h

52.4.2.2.2 Function

Contains version numbers for the module design and feature set.

52.4.2.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

52.4.2.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module specification. Read-only field.
23-16 MINOR	Minor Version Number Returns the minor version number for the module specification. Read-only field.
15-0 FEATURE	Module Identification Number Returns the feature set number. Read-only field. 0000000000000100b - Standard feature set supporting a 32-bit shift register.

52.4.2.3 Parameter Register (PARAM)

52.4.2.3.1 Offset

Register	Offset
PARAM	4h

52.4.2.3.2 Function

Contains parameter values that were implemented in the module.

52.4.2.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXFIFO								TXFIFO							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

52.4.2.3.4 Fields

Field	Function
31-24	Reserved

Table continues on the next page...

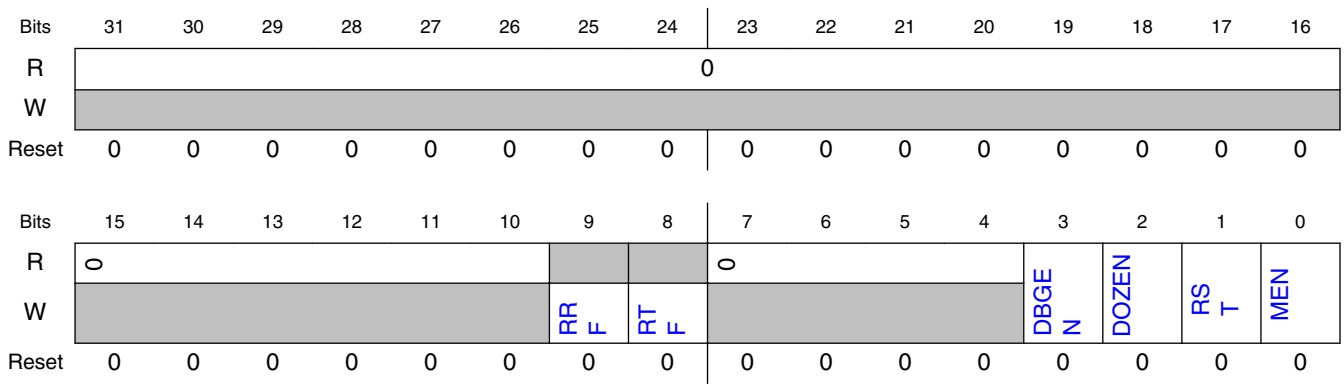
Field	Function
—	
23-16 —	Reserved
15-8 RXFIFO	Receive FIFO Size Sets the maximum number of words in the receive FIFO, which is 2^{RXFIFO}
7-0 TXFIFO	Transmit FIFO Size Sets the maximum number of words in the transmit FIFO, which is 2^{TXFIFO}

52.4.2.4 Control Register (CR)

52.4.2.4.1 Offset

Register	Offset
CR	10h

52.4.2.4.2 Diagram



52.4.2.4.3 Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO 0b - No effect 1b - Receive FIFO is reset

Table continues on the next page...

Field	Function
8 RTF	Reset Transmit FIFO 0b - No effect 1b - Transmit FIFO is reset
7-4 —	Reserved
3 DBGEN	Debug Enable Enables or disables the LPSPI module in debug mode. Debug Enable bit should be updated only when the LPSPI module is disabled. 0b - LPSPI module is disabled in debug mode 1b - LPSPI module is enabled in debug mode
2 DOZEN	Doze Mode Enable Enables or disables the LPSPI module in Doze mode. Doze Mode Enable bit should be updated only when the LPSPI module is disabled. 0b - LPSPI module is enabled in Doze mode 1b - LPSPI module is disabled in Doze mode
1 RST	Software Reset Reset all internal logic and registers, except the Control Register. RST remains set until cleared by software. 0b - Module is not reset 1b - Module is reset
0 MEN	Module Enable 0b - Module is disabled 1b - Module is enabled

52.4.2.5 Status Register (SR)

52.4.2.5.1 Offset

Register	Offset
SR	14h

52.4.2.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							MBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DMF	REF	TEF	TCF	FCF	WCF	0						RDF	TDF
W			W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

52.4.2.5.3 Fields

Field	Function
31-25 —	Reserved
24 MBF	Module Busy Flag 0b - LPSPI is idle 1b - LPSPI is busy
23-14 —	Reserved
13 DMF	Data Match Flag Indicates that the received data has matched the MATCH0 and/or MATCH1 fields (as configured by CFGR1[MATCFG], Configuration Register 1). 0b - Have not received matching data 1b - Have received matching data
12 REF	Receive Error Flag The Receive Error Flag will set when the Receiver FIFO overflows. When the Receive Error Flag is set, it is recommended to first end the transfer, empty the Receive FIFO, clear the Receive Error Flag and then restart the transfer from the beginning. 0b - Receive FIFO has not overflowed 1b - Receive FIFO has overflowed
11 TEF	Transmit Error Flag The Transmit Error Flag will set when the Transmit FIFO underruns. When the Transmit Error Flag is set, it is recommended to first end the transfer, clear the Transmit Error Flag and then restart the transfer from the beginning. 0b - Transmit FIFO underrun has not occurred 1b - Transmit FIFO underrun has occurred
10 TCF	Transfer Complete Flag In master mode when the LPSPI returns to idle state with the transmit FIFO empty, the Transfer Complete Flag will set. 0b - All transfers have not completed 1b - All transfers have completed
9	Frame Complete Flag

Table continues on the next page...

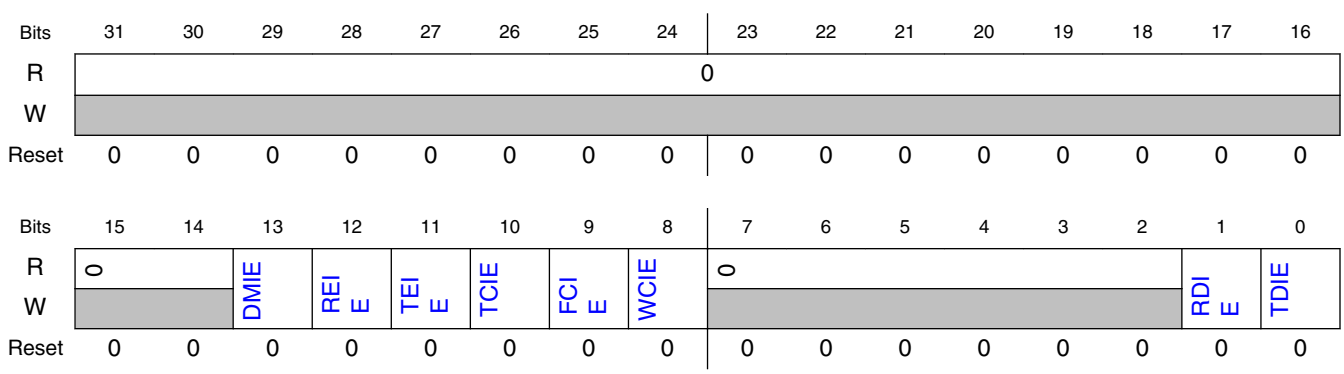
Field	Function
FCF	The Frame Complete Flag will set at the end of each frame transfer, when the PCS negates. 0b - Frame transfer has not completed 1b - Frame transfer has completed
8 WCF	Word Complete Flag The Word Complete Flag will set when the last bit of a received word is sampled. 0b - Transfer of a received word has not yet completed 1b - Transfer of a received word has completed
7-2 —	Reserved
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than FCR[RXWATER] (FIFO Control Register) 0b - Receive Data is not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than FCR[TXWATER] (FIFO Control Register) 0b - Transmit data not requested 1b - Transmit data is requested

52.4.2.6 Interrupt Enable Register (IER)

52.4.2.6.1 Offset

Register	Offset
IER	18h

52.4.2.6.2 Diagram



52.4.2.6.3 Fields

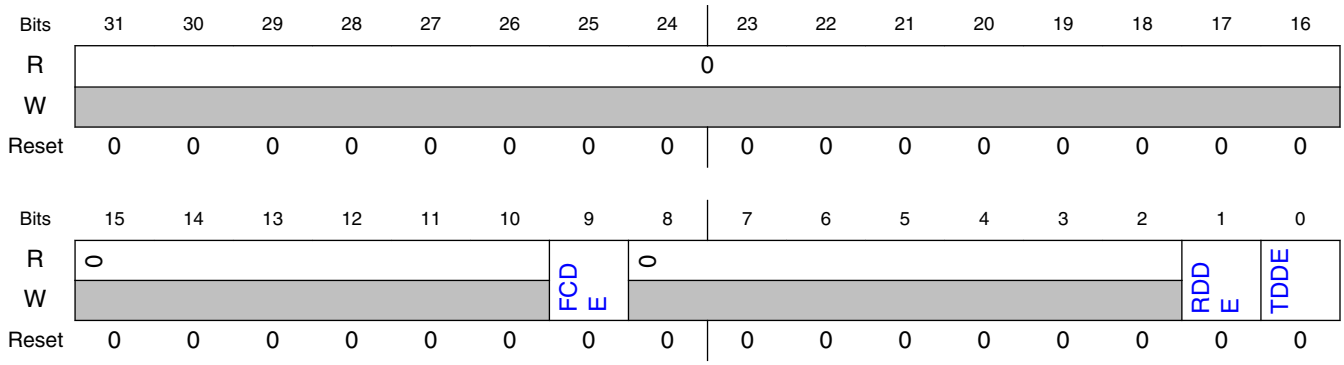
Field	Function
31-14 —	Reserved
13 DMIE	Data Match Interrupt Enable 0b - Disabled 1b - Enabled
12 REIE	Receive Error Interrupt Enable 0b - Disabled 1b - Enabled
11 TEIE	Transmit Error Interrupt Enable 0b - Disabled 1b - Enabled
10 TCIE	Transfer Complete Interrupt Enable 0b - Disabled 1b - Enabled
9 FCIE	Frame Complete Interrupt Enable 0b - Disabled 1b - Enabled
8 WCIE	Word Complete Interrupt Enable 0b - Disabled 1b - Enabled
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable 0b - Disabled 1b - Enabled
0 TDIE	Transmit Data Interrupt Enable 0b - Disabled 1b - Enabled

52.4.2.7 DMA Enable Register (DER)

52.4.2.7.1 Offset

Register	Offset
DER	1Ch

52.4.2.7.2 Diagram



52.4.2.7.3 Fields

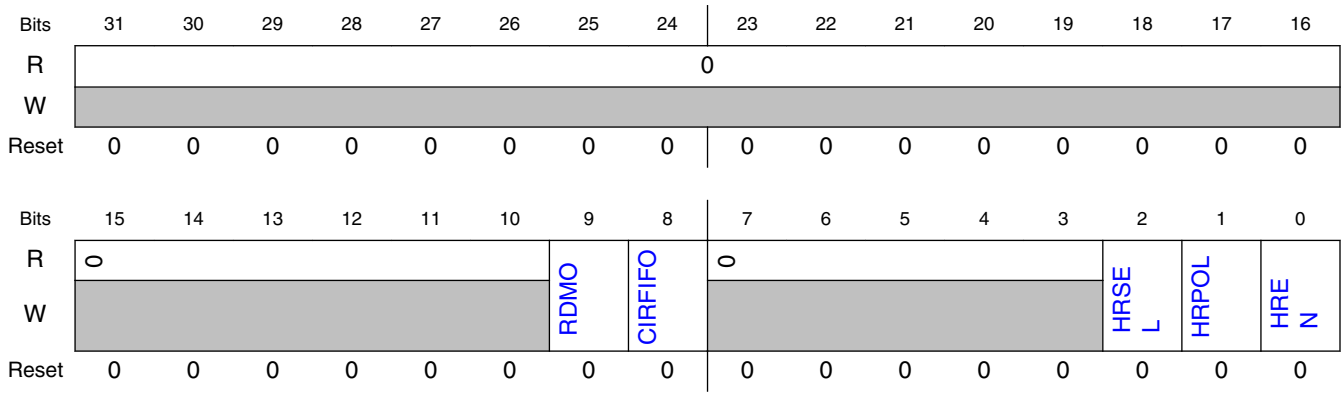
Field	Function
31-10 —	Reserved
9 FCDE	Frame Complete DMA Enable Enables DMA end-of-packet processing when set. After the last word of a frame is read from the receive data FIFO, reading the receive data FIFO will return an end-of-packet signal with the receive data forced to 0xFFFF_FFFF. This will continue until the DMA minor loop completes, and then the Frame Complete Flag will be cleared if the receive FIFO is empty or if the LPSPI is busy. 0b - DMA request is disabled 1b - DMA request is enabled
8-2 —	Reserved
1 RDDE	Receive Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled
0 TDDE	Transmit Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled

52.4.2.8 Configuration Register 0 (CFGR0)

52.4.2.8.1 Offset

Register	Offset
CFGR0	20h

52.4.2.8.2 Diagram



52.4.2.8.3 Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only When Receive Data Match Only is enabled, all received data that does not cause the Data Match Flag (SR[DMF]) to set is discarded. • Receive Data Match Only bit should be set when the LPSPI is idle and the Data Match Flag is clear • After the Data Match Flag (DMF) is set, the Receive Data Match Only bit configuration is ignored • When disabling RDMO and to ensure that no receive data is lost, before clearing the Data Match Flag, first clear Receive Data Match Only (RDMO) 0b - Received data is stored in the receive FIFO as in normal operations 1b - Received data is discarded unless the Data Match Flag (DMF) is set
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO will be emptied as it normally is, but after the LPSPI is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This restoring of the read pointer will cause the contents of the transmit FIFO to be cycled through repeatedly. 0b - Circular FIFO is disabled 1b - Circular FIFO is enabled
7-3 —	Reserved
2 HRSEL	Host Request Select Selects the source of the host request input. When the host request function is enabled with the LPSPI_HREQ pin, the LPSPI_PCS[1] function is disabled. 0b - Host request input is the LPSPI_HREQ pin 1b - Host request input is the input trigger
1 HRPOL	Host Request Polarity Configures the polarity of the host request pin. 0b - Active low 1b - Active high

Table continues on the next page...

Field	Function
0 HREN	Host Request Enable When enabled in master mode, the LPSPI will only start a new SPI bus transfer if the host request input is asserted. When the LPSPI is busy, the host request input is ignored. 0b - Host request is disabled 1b - Host request is enabled

52.4.2.9 Configuration Register 1 (CFGR1)

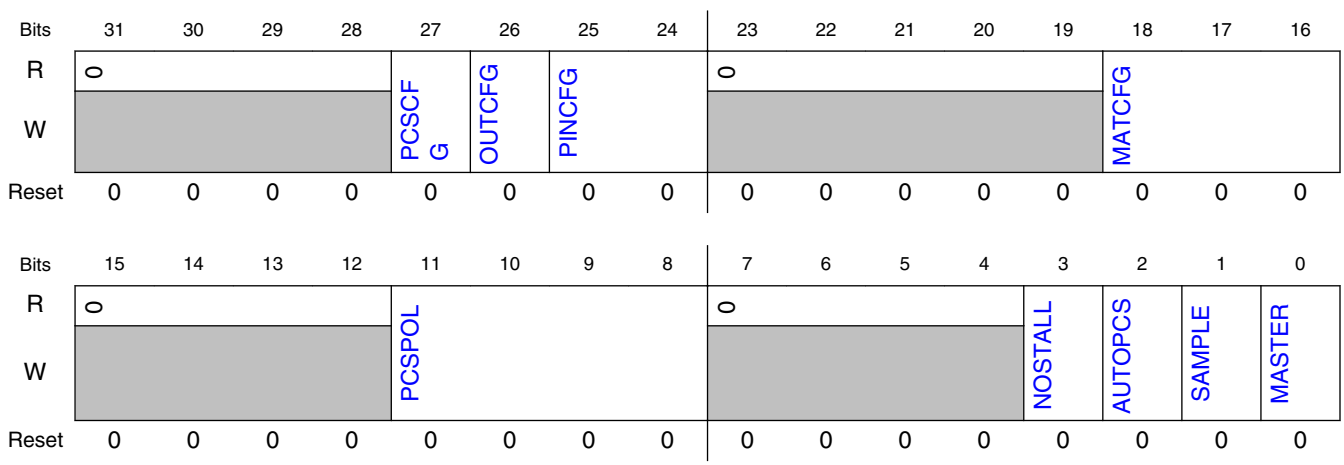
52.4.2.9.1 Offset

Register	Offset
CFGR1	24h

52.4.2.9.2 Function

The CFGR1 should only be written when the LPSPI is disabled.

52.4.2.9.3 Diagram



52.4.2.9.4 Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Field	Function
27 PCSCFG	Peripheral Chip Select Configuration If performing 4-bit transfers, the Peripheral Chip Select Configuration bit must be set. 0b - PCS[3:2] are enabled 1b - PCS[3:2] are disabled
26 OUTCFG	Output Configuration Configures if the output data is tristated between accesses (LPSPI_PCS is negated). If performing 2-bit or 4-bit transfers, the Output Configuration bit must be set. 0b - Output data retains last value when chip select is negated 1b - Output data is tristated when chip select is negated
25-24 PINCFG	Pin Configuration Configures which pins are used for input and output data during 1-bit transfers. If performing 2-bit or 4-bit transfers, the Pin Configuration field must be 00. 00b - SIN is used for input data and SOUT is used for output data 01b - SIN is used for both input and output data 10b - SOUT is used for both input and output data 11b - SOUT is used for input data and SIN is used for output data
23-19 —	Reserved
18-16 MATCFG	Match Configuration Configures the condition that will cause the DMF to set. **NOTE:** *Syntax:* * is boolean AND, + is boolean OR 000b - Match is disabled 001b - Reserved 010b - 010b - Match is enabled, if 1st data word equals MATCH0 OR MATCH1, i.e., (1st data word = MATCH0 + MATCH1) 011b - 011b - Match is enabled, if any data word equals MATCH0 OR MATCH1, i.e., (any data word = MATCH0 + MATCH1) 100b - 100b - Match is enabled, if 1st data word equals MATCH0 AND 2nd data word equals MATCH1, i.e., [(1st data word = MATCH0) * (2nd data word = MATCH1)] 101b - 101b - Match is enabled, if any data word equals MATCH0 AND the next data word equals MATCH1, i.e., [(any data word = MATCH0) * (next data word = MATCH1)] 110b - 110b - Match is enabled, if (1st data word AND MATCH1) equals (MATCH0 AND MATCH1), i.e., [(1st data word * MATCH1) = (MATCH0 * MATCH1)] 111b - 111b - Match is enabled, if (any data word AND MATCH1) equals (MATCH0 AND MATCH1), i.e., [(any data word * MATCH1) = (MATCH0 * MATCH1)]
15-12 —	Reserved
11-8 PCSPOL	Peripheral Chip Select Polarity Configures the polarity of each Peripheral Chip Select pin. Each PCSPOL bit position (let's call it *n*) corresponds to a PCS[*n*] pin. For example, PCSPOL[0] is chip select 0 (at PCS[0] pin), and PCSPOL[1] is chip select 1 (at PCS[1] pin). If a PCSPOL bit: • =0, then the PCS[*n*] pin is active low. • =1, then the PCS[*n*] pin is active high. **NOTE:** The entire PCSPOL field is not fully supported in every LPSPI module instance. Refer to the chip-specific information for LPSPI.
7-4 —	Reserved
3	No Stall

Table continues on the next page...

Field	Function
NOSTALL	In master mode, the LPSPI will stall transfers when the transmit FIFO is empty or when the receive FIFO is full, ensuring that no transmit FIFO underrun or receive FIFO overrun can occur. Setting the No Stall bit will disable this functionality. 0b - Transfers will stall when the transmit FIFO is empty or the receive FIFO is full 1b - Transfers will not stall, allowing transmit FIFO underruns or receive FIFO overruns to occur
2 AUTOPCS	Automatic PCS For correct operations, the LPSPI slave normally requires the PCS to negate between frames. Setting the Automatic PCS bit will cause the LPSPI to generate an internal PCS signal at the end of each transfer word when the Clock Phase bit TCR[CPHA] = 1. • When the Automatic PCS bit is set, the SCK must remain idle for at least 4 LPSPI functional clock cycles (divided by the Prescaler Value TCR[PRESCALE] configuration) between each word, to ensure correct operations • In master mode, the Automatic PCS bit is ignored 0b - Automatic PCS generation is disabled 1b - Automatic PCS generation is enabled
1 SAMPLE	Sample Point When set, the LPSPI master will sample the input data on a delayed LPSPI_SCK edge, which improves the setup time when sampling data. • The input data setup time in master mode with delayed LPSPI_SCK edge is equal to the input data setup time in slave mode • In slave mode, the SAMPLE bit is ignored 0b - Input data is sampled on SCK edge 1b - Input data is sampled on delayed SCK edge
0 MASTER	Master Mode Configures the LPSPI in master or slave mode. The Master Mode bit directly controls the direction of the LPSPI_SCK and LPCPI_PCS pins. 0b - Slave mode 1b - Master mode

52.4.2.10 Data Match Register 0 (DMR0)

52.4.2.10.1 Offset

Register	Offset
DMR0	30h

52.4.2.10.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.2.10.3 Fields

Field	Function
31-0 MATCH0	Match 0 Value When Receive Data Match Only (CFGR0[RDMO]) is enabled, the Match 0 Value is compared against the received data.

52.4.2.11 Data Match Register 1 (DMR1)

52.4.2.11.1 Offset

Register	Offset
DMR1	34h

52.4.2.11.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MATCH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.2.11.3 Fields

Field	Function
31-0 MATCH1	Match 1 Value When Receive Data Match Only (CFGR0[RDMO]) is enabled, the Match 1 Value is compared against the received data.

52.4.2.12 Clock Configuration Register (CCR)

52.4.2.12.1 Offset

Register	Offset
CCR	40h

52.4.2.12.2 Function

The Clock Configuration Register is only used in master mode, and the Clock Configuration Register cannot be changed when the LPSPI is enabled.

52.4.2.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SCKPCS								PCSSCK							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DBT								SCKDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.2.12.4 Fields

Field	Function
31-24 SCKPCS	SCK-to-PCS Delay In master mode: configures the delay from the last SCK edge to the PCS negation.

Table continues on the next page...

Field	Function
	• The delay is equal to (SCKPCS + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle.
23-16 PCSSCK	PCS-to-SCK Delay In master mode: configures the delay from the PCS assertion to the first SCK edge. • The delay is equal to (PCSSCK + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum delay is 1 cycle.

15-8
DBT
	Delay Between Transfers
	In master mode:
	• Configures the delay from the PCS negation to the next PCS assertion.
	  • The delay is equal to (DBT + 2) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration.
	  • The minimum delay is 2 cycles.
	  • Half of the delay occurs before PCS assertion and the other half of the delay occurs after PCS negation. If the command word is updated between 2 transfers, then the command word is updated half-way between the PCS negation of the last transfer and PCS assertion of the next transfer.
	  • The command word sets which PCS signal is used (of PCS[3:0]}, the polarity/phase of the SCK signal, and the Prescaler Value.
	• Configures the delay from the last SCK edge of a transfer word and the first SCK edge of the next transfer word, in a continuous transfer.
	  • The delay is equal to (DBT + 1) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration.
	  • The minimum delay is 1 cycle.

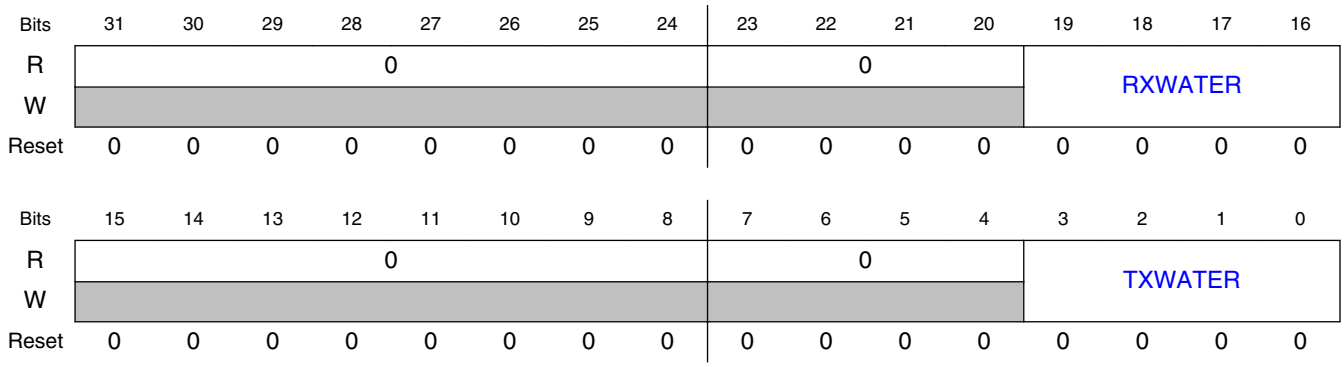
7-0 SCKDIV	SCK Divider In master mode, the SCK Divider configures the divide ratio of the SCK pin. • The SCK period is equal to (SCKDIV+2) cycles of the LPSPI functional clock divided by the Prescaler Value TCR[PRESCALE] configuration. • The minimum SCK period is 2 cycles. • If the SCK period is an odd number of cycles, then the 1st half of the SCK period will be 1 cycle longer than the 2nd half of the SCK period.

52.4.2.13 FIFO Control Register (FCR)

52.4.2.13.1 Offset

Register	Offset
FCR	58h

52.4.2.13.2 Diagram



52.4.2.13.3 Fields

Field	Function
31-24 —	Reserved
23-20 —	Reserved
19-16 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal or greater than the FIFO size will be truncated.
15-8 —	Reserved
7-4 —	Reserved
3-0 TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal or greater than the FIFO size will be truncated.

52.4.2.14 FIFO Status Register (FSR)

52.4.2.14.1 Offset

Register	Offset
FSR	5Ch

52.4.2.14.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0			RXCOUNT				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0			TXCOUNT				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

52.4.2.14.3 Fields

Field	Function
31-24 —	Reserved
23-21 —	Reserved
20-16 RXCOUNT	Receive FIFO Count Returns the number of words currently stored in the receive FIFO.
15-8 —	Reserved
7-5 —	Reserved
4-0 TXCOUNT	Transmit FIFO Count Returns the number of words currently stored in the transmit FIFO.

52.4.2.15 Transmit Command Register (TCR)

52.4.2.15.1 Offset

Register	Offset
TCR	60h

52.4.2.15.2 Function

Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO, in the order that the data are written. The Command Register should only be written using 32-bit writes. Command Register writes will be tagged and cause the command register to update, after that entry reaches the top of the FIFO. This allows changes to the command word and the transmit data itself to be interleaved. Changing the command word will cause all subsequent SPI bus transfers to be performed using the new command word.

- **In master mode,** writing a new command word does not initiate a new transfer, unless TXMSK is set. Transfers are initiated by transmit data in the transmit FIFO, or by a new command word (with TXMSK set). Hardware will clear TXMSK when the LPSPI_PCS negates.
- **In master mode,** if the command word is changed before an existing frame has completed, then the existing frame will terminate and the command word will then update. The command word can be changed during a continuous transfer, if CONTC of the new command word is set and the command word is written on a frame size boundary.
- **In slave mode,** the command word should be changed only when the LPSPI is idle and there is no SPI bus transfer.

Avoid register reading problems: Reading the Transmit Command Register will return the current state of the command register. Reading the Transmit Command Register at the same time that the Transmit Command Register is loaded from the transmit FIFO, can return an incorrect Transmit Command Register value. It is recommended:

- to either read the Transmit Command Register when the transmit FIFO is empty,
- or to read the Transmit Command Register more than once and then compare the returned values.

52.4.2.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CPOL	CPHA	PRESCALE			Reserved	PCS		LSBF	BYSW	CONT	CONTC	RXMSK	TXMSK	WIDTH	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				FRAMESZ											
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

52.4.2.15.4 Fields

Field	Function
31 CPOL	Clock Polarity The Clock Polarity field is only updated between frames. 0b - The inactive state value of SCK is low 1b - The inactive state value of SCK is high
30 CPHA	Clock Phase The Clock Phase field is only updated between frames. 0b - Data is captured on the leading edge of SCK and changed on the following edge of SCK 1b - Data is changed on the leading edge of SCK and captured on the following edge of SCK
29-27 PRESCALE	Prescaler Value For all SPI bus transfers, the Prescaler value applied to the clock configuration register. The Prescaler Value field is only updated between frames. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128
26 —	Reserved
25-24 PCS	Peripheral Chip Select Configures the peripheral chip select used for the transfer. The Peripheral Chip Select field is only updated between frames. **NOTE:** The entire PCS field is not fully supported in every LPSPI module instance. Refer to the chip-specific information for LPSPI. 00b - Transfer using LPSPI_PCS[0] 01b - Transfer using LPSPI_PCS[1] 10b - Transfer using LPSPI_PCS[2] 11b - Transfer using LPSPI_PCS[3]
23 LSBF	LSB First 0b - Data is transferred MSB first 1b - Data is transferred LSB first
22 BYSW	Byte Swap Byte swap will swap the contents of [31:24] with [7:0] and [23:16] with [15:8] for each transmit data word read from the FIFO and for each received data word stored to the FIFO (or compared with match registers). 0b - Byte swap is disabled 1b - Byte swap is enabled
21 CONT	Continuous Transfer • In master mode, continuous transfer will keep the PCS asserted at the end of the frame size, until a command word is received that starts a new frame. • In slave mode, when continuous transfer is enabled, the LPSPI will only transmit the first FRAMESZ bits; after which the LPSPI will transmit received data (assuming a 32-bit shift register). 0b - Continuous transfer is disabled 1b - Continuous transfer is enabled

Table continues on the next page...

Field	Function
20 CONTC	Continuing Command In master mode, the Continuing Command bit allows the command word to be changed within a continuous transfer. • The initial command word must enable continuous transfer (CONT=1), • the continuing command must set this bit (CONTC=1), • and the continuing command word must be loaded on a frame size boundary. For example, if the continuous transfer has a frame size of 64-bits, then a continuing command word must be loaded on a 64-bit boundary. 0b - Command word for start of new transfer 1b - Command word for continuing transfer
19 RXMSK	Receive Data Mask When set, receive data is masked (receive data is not stored in receive FIFO). 0b - Normal transfer 1b - Receive data is masked
18 TXMSK	Transmit Data Mask When set, transmit data is masked (no data is loaded from transmit FIFO and output pin is tristated). In master mode, the Transmit Data Mask bit will initiate a new transfer which cannot be aborted by another command word; the Transmit Data Mask bit will be cleared by hardware at the end of the transfer. 0b - Normal transfer 1b - Mask transmit data
17-16 WIDTH	Transfer Width For 2-bit or 4-bit transfers, either Receive Data Mask (RXMSK) or Transmit Data Mask (TXMSK) must be set. 00b - 1 bit transfer 01b - 2 bit transfer 10b - 4 bit transfer 11b - Reserved
15-12 —	Reserved. Software should only write zero to this field.
11-0 FRAMESZ	Frame Size Configures the frame size in number of bits equal to (FRAMESZ + 1). • The minimum frame size is 8 bits • The minimum word size is 2 bits; a frame size of 33 bits (or similar) is not supported. • If the frame size is larger than 32 bits, then the frame is divided into multiple words of 32-bits; each word is loaded from the transmit FIFO and stored in the receive FIFO separately. • If the size of the frame is not divisible by 32, then the last load of the transmit FIFO and store of the receive FIFO will contain the remainder bits. For example, a 72-bit transfer will consist of 3 words: the 1st and 2nd words are 32 bits, and the 3rd word is 8 bits.

52.4.2.16 Transmit Data Register (TDR)

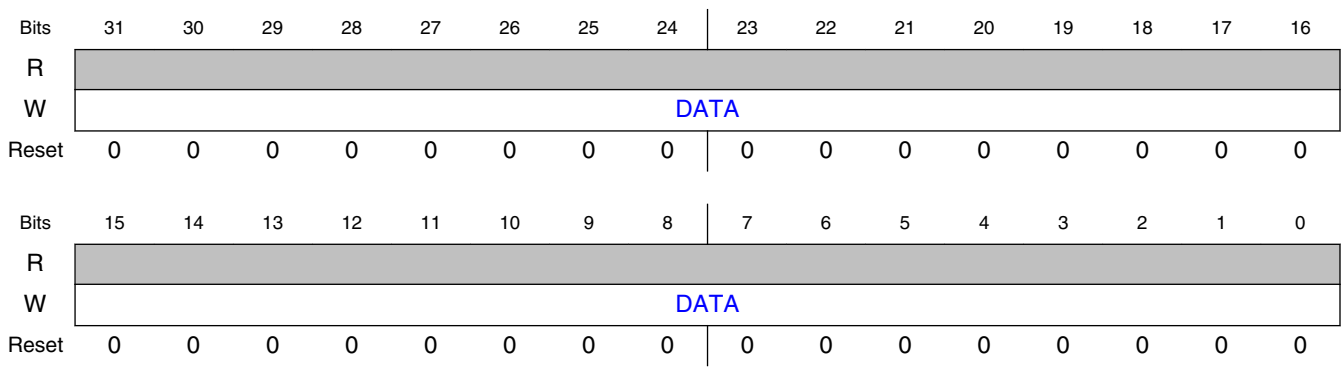
52.4.2.16.1 Offset

Register	Offset
TDR	64h

52.4.2.16.2 Function

Writes to either the Transmit Command Register or Transmit Data Register will push the data into the transmit FIFO, in the order that it (the data) was written. The Data Register can be written using 32-bit, 16-bit or 8-bit writes, but each write will push data into the FIFO with zero pushed in unwritten bytes.

52.4.2.16.3 Diagram



52.4.2.16.4 Fields

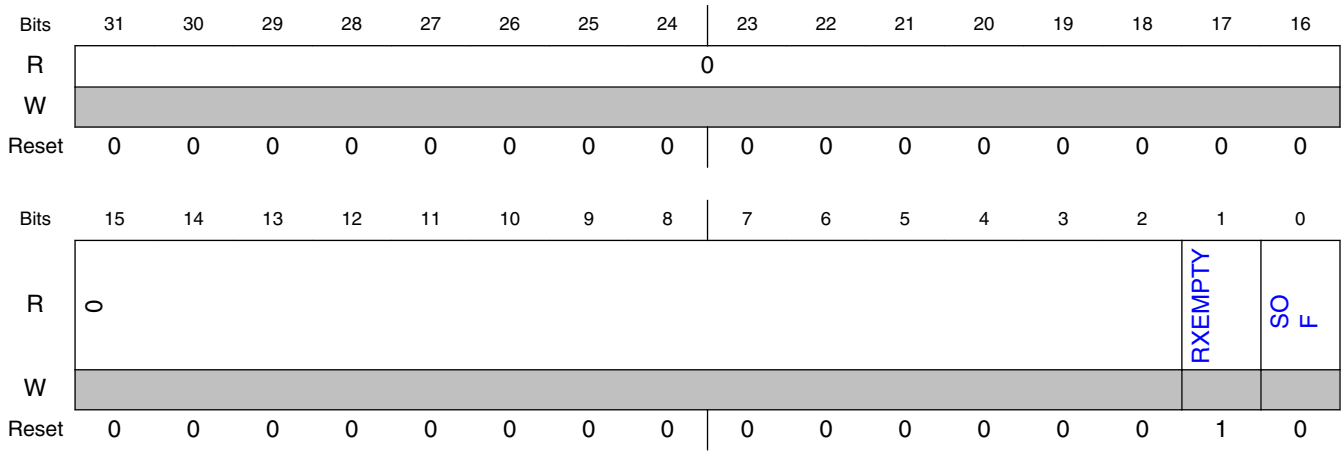
Field	Function
31-0 DATA	Transmit Data Both 8-bit and 16-bit writes of transmit data will zero-extend the data written and push the data into the transmit FIFO. To zero-extend 8-bit and 16-bit writes (to 32 bits), means that the higher order (most significant) empty parts of the 8-bit and 16-bit writes are filled with zeroes.

52.4.2.17 Receive Status Register (RSR)

52.4.2.17.1 Offset

Register	Offset
RSR	70h

52.4.2.17.2 Diagram



52.4.2.17.3 Fields

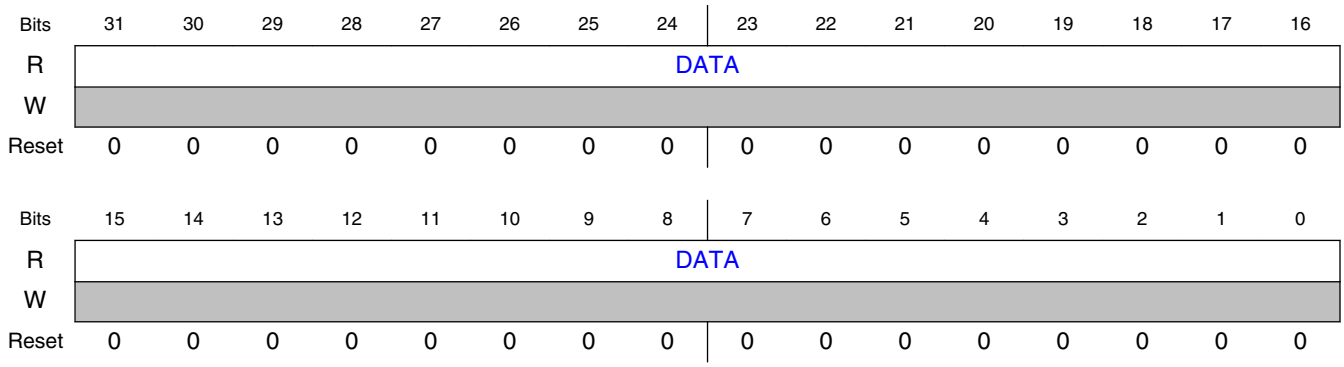
Field	Function
31-2 —	Reserved
1 RXEMPTY	RX FIFO Empty 0b - RX FIFO is not empty 1b - RX FIFO is empty
0 SOF	Start Of Frame Indicates that this is the first data word received after LPSPI_PCS assertion. 0b - Subsequent data word received after LPSPI_PCS assertion 1b - First data word received after LPSPI_PCS assertion

52.4.2.18 Receive Data Register (RDR)

52.4.2.18.1 Offset

Register	Offset
RDR	74h

52.4.2.18.2 Diagram



52.4.2.18.3 Fields

Field	Function
31-0 DATA	Receive Data

Chapter 53
Low Power Inter-Integrated Circuit (LPI2C)

53.1 Chip-specific LPI2C information

Table 53-1. Reference links to related information

Topic	Related module	Reference
Full description	LPI2C	LPI2C
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

53.1.1 LPI2C

The LPI2C is a low power Inter-Integrated Circuit (I2C) module that supports an efficient interface to an I2C bus as a master. The LPI2C can continue operating in stop modes provided an appropriate clock is available and is designed for low CPU overhead with DMA offloading of FIFO register accesses. The LPI2C implements logic support for standard and fast modes of operation.

The following table shows the LPI2C configuration:

Table 53-2. LPI2C configuration

Parameter	Description
Name	LPI2C
Instances	8
Configurable features	LPI2C0-7: SMBUS, 4TX/RX FIFO
Interface speed	LPI2C0-7: 1 Mbps normal mode max
External I/O pins	LPI2C0-7: SDA, SCL. See the attached IOMUXC spreadsheet for pinout details.

53.1.2 Slave mode support

All LPI2C instances in i.MX 7ULP don't support Slave mode. LPI2C module may have been generated including "Slave mode" in order to keep SMBUS feature and all features of Slave mode are not verified. Therefore, HREQ, SCLS and SDAS signals are not required.

53.2 Introduction

The LPI2C is a low power Inter-Integrated Circuit (I2C) module that supports an efficient interface to an I^2C bus as a master and/or as a slave.

- The LPI2C implements logic support for standard-mode, fast-mode, fast-mode plus and ultra-fast modes of operation.
- The LPI2C is designed to use little CPU overhead, with DMA offloading of FIFO register accesses.
- The LPI2C can continue operating in stop modes if an appropriate clock is available.

The LPI2C module also complies with the System Management Bus (SMBus) Specification, version 2. The SMBus is a single-ended simple two-wire bus, which is typically used for low bandwidth communications.

NOTE

The I^2C (Inter-Integrated Circuit) serial bus is multi-master, multi-slave, packet-switched, and single-ended, and is often used to attach microcontroller ICs to lower-speed peripheral ICs.

53.2.1 Features

The LPI2C supports:

- Standard, Fast, Fast+ and Ultra Fast modes are supported
- High speed mode (HS) in slave mode
- High speed mode (HS) in master mode, if SCL pin implements current source pull-up (device-specific)
- Multi-master support, including synchronization and arbitration. Multi-master means any number of master nodes can be present. Additionally, master and slave roles may be changed between messages (after a STOP is sent).
- Clock stretching: Sometimes multiple I2C nodes may be driving the lines at the same time. If any I2C node is driving a line low, then that line will be low. I2C nodes that

are starting to transmit a logical one (by letting the line float high) can detect that the line is low, and thereby know that another I2C node is active at the same time.
    - When node detection is used on the SCL line, it is called *clock stretching*, and clock stretching is used as a I2C flow control mechanism for multiple laves.
    - When node detection is used on the SDA line, it is called *arbitration*, and arbitration ensures that there is only one I2C node transmitter at a time.
- General call, 7-bit and 10-bit addressing
- Software reset, START byte and Device ID (also require software support)

The LPI2C master supports:

- Command/transmit FIFO of 4words.
- Receive FIFO of 4words.
- Command FIFO will wait for idle I2C bus before initiating transfer
- Command FIFO can initiate (repeated) START and STOP conditions and one or more master-receiver transfers
- STOP condition can be generated from command FIFO, or generated automatically when the transmit FIFO is empty
- Host request input to control the start time of an I2C bus transfer
- Flexible receive data match can generate interrupt on data match and/or discard unwanted data
- Flag and optional interrupt to signal Repeated START condition, STOP condition, loss of arbitration, unexpected NACK, and command word errors
- Supports configurable bus idle timeout and pin-stuck-low timeout

The LPI2C slave supports:

- Separate I2C slave registers to minimize software overhead because of master/slave switching
- Support for 7-bit or 10-bit addressing, address range, SMBus alert and general call address
- Transmit data register that supports interrupt or DMA requests
- Receive data register that supports interrupt or DMA requests
- Software-controllable ACK or NACK, with optional clock stretching on ACK/NACK bit
- Configurable clock stretching, to avoid transmit FIFO underrun and receive FIFO overrun errors
- Flag and optional interrupt at end of packet, STOP condition, or bit error detection

53.2.2 Block Diagram

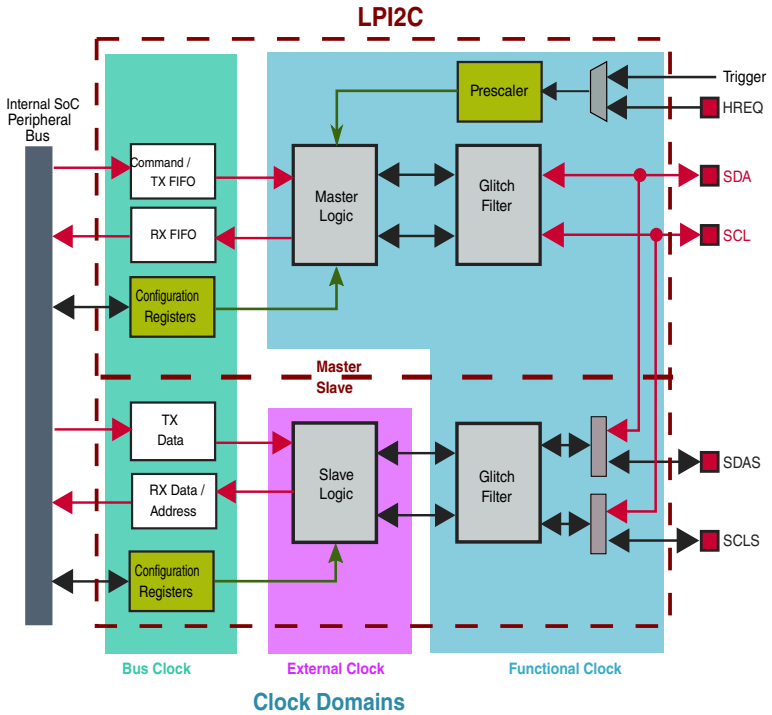


Figure 53-1. LPI2C block diagram

53.2.3 Modes of operation

Table 53-3. Chip modes supported by the LPI2C module

Chip mode	LPI2C Operation
Run	Normal operations
Stop/Wait	Can continue operating in stop/wait modes if the Doze Enable bit (MCR[DOZEN]) is clear and the LPI2C is using an external or internal clock source that remains operating during stop/wait modes.
Low Power Stop (also called Low Leakage Stop or LLS)	Before entering low power stop mode, the LPI2C will wait for the current transfer to finish any pending operation, while temporarily ignoring The Doze Enable (MCR[DOZEN]) bit.
Debug	Can continue operating in debug mode if the Debug Enable bit (MCR[DBGE]) is set.

53.2.4 Signal Descriptions

Table 53-4. Signals

Signal	Name	2-Wire Scheme	4-Wire Scheme	I/O
SCL	LPI2C clock line	SCL	In 4-wire mode, this is the SCL input pin.	I/O
SDA	LPI2C data line	SDA	In 4-wire mode, this is the SDA input pin.	I/O
HREQ	Host request	If host request is asserted and the I2C bus is idle, then it will initiate an LPI2C master transfer.		I
SCLS	Secondary I2C clock line	Not used	In 4-wire mode, this is the SCLS output pin. If LPI2C master/slave are configured to use separate pins, then this the LPI2C slave SCL pin.	I/O
SDAS	Secondary I2C data line	Not used	In 4-wire mode, this is the SDAS output pin. If LPI2C master/slave are configured to use separate pins, then this the LPI2C slave SDA pin.	I/O

53.2.5 Wiring options

LPI2C can be used to implement 2-wire or 4-wire I^{2}C serial busses.

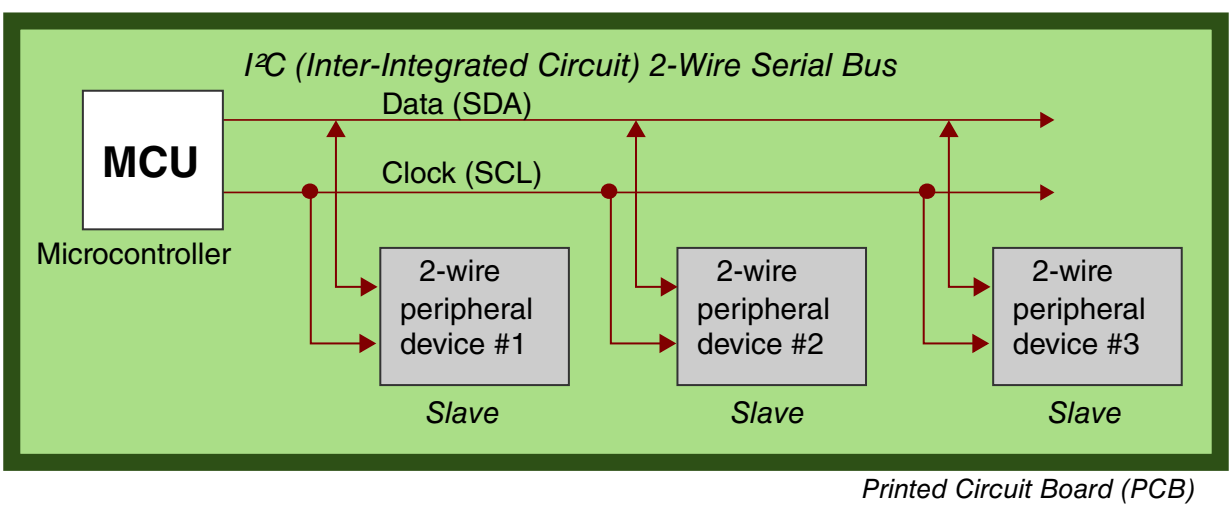


Figure 53-2. 2-Wire scheme

Some applications can provide a lot of load and noise on the I^{2}C bus; to ensure robust I^{2}C operations, a 4-wire interface with the MCU can be used, splitting the 2 lines into inputs and outputs. Using a few transistors, resistors and diodes, designers can make their own inexpensive line drivers.

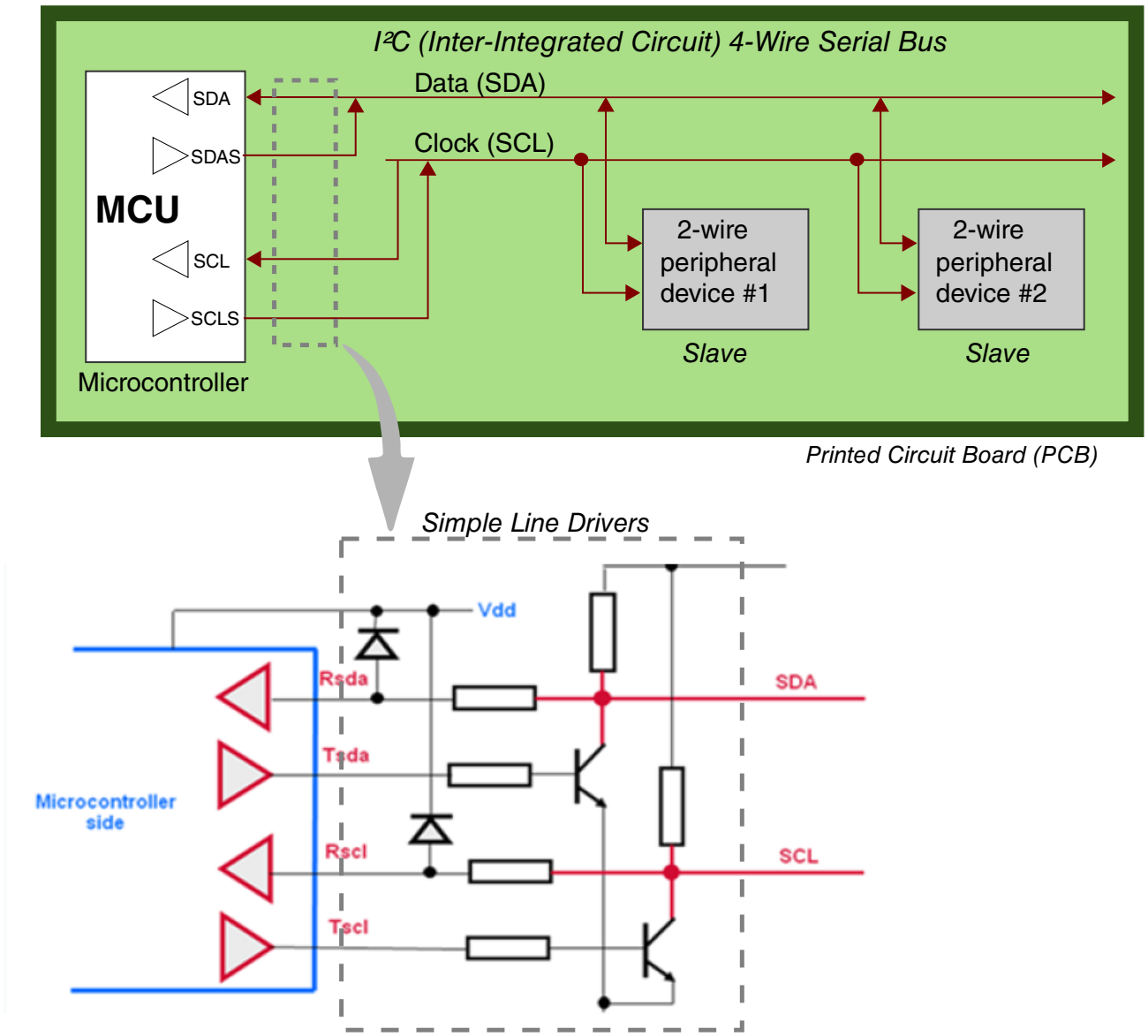


Figure 53-3. 4-Wire scheme

53.3 Memory Map and Registers

NOTE

Writing a Read-Only (RO) register or reading a Write-Only (WO) register can cause bus errors. This module will not check if programmed values in the registers are correct; the application software must ensure that valid programmed values are being written.

53.3.1 LPI2C register descriptions

53.3.1.1 LPI2C Memory map

LPI2C0 base address: 4103_3000h

LPI2C1 base address: 4103_4000h

LPI2C2 base address: 4103_5000h

LPI2C3 base address: 4103_6000h

LPI2C4 base address: 402B_0000h

LPI2C5 base address: 402C_0000h

LPI2C6 base address: 40A4_0000h

LPI2C7 base address: 40A5_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_0003h
4h	Parameter Register (PARAM)	32	RO	0000_0202h
10h	Master Control Register (MCR)	32	RW	0000_0000h
14h	Master Status Register (MSR)	32	W1C	0000_0001h
18h	Master Interrupt Enable Register (MIER)	32	RW	0000_0000h
1Ch	Master DMA Enable Register (MDER)	32	RW	0000_0000h
20h	Master Configuration Register 0 (MCFGR0)	32	RW	0000_0000h
24h	Master Configuration Register 1 (MCFGR1)	32	RW	0000_0000h
28h	Master Configuration Register 2 (MCFGR2)	32	RW	0000_0000h
2Ch	Master Configuration Register 3 (MCFGR3)	32	RW	0000_0000h
40h	Master Data Match Register (MDMR)	32	RW	0000_0000h
48h	Master Clock Configuration Register 0 (MCCR0)	32	RW	0000_0000h
50h	Master Clock Configuration Register 1 (MCCR1)	32	RW	0000_0000h
58h	Master FIFO Control Register (MFCR)	32	RW	0000_0000h
5Ch	Master FIFO Status Register (MFSR)	32	RO	0000_0000h
60h	Master Transmit Data Register (MTDR)	32	WO	0000_0000h
70h	Master Receive Data Register (MRDR)	32	RO	0000_4000h
110h	Slave Control Register (SCR)	32	RW	0000_0000h
114h	Slave Status Register (SSR)	32	W1C	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
118h	Slave Interrupt Enable Register (SIER)	32	RW	0000_0000h
11Ch	Slave DMA Enable Register (SDER)	32	RW	0000_0000h
124h	Slave Configuration Register 1 (SCFGR1)	32	RW	0000_0000h
128h	Slave Configuration Register 2 (SCFGR2)	32	RW	0000_0000h
140h	Slave Address Match Register (SAMR)	32	RW	0000_0000h
150h	Slave Address Status Register (SASR)	32	RO	0000_4000h
154h	Slave Transmit ACK Register (STAR)	32	RW	0000_0000h
160h	Slave Transmit Data Register (STDR)	32	WO	0000_0000h
170h	Slave Receive Data Register (SRDR)	32	RO	0000_4000h

53.3.1.2 Version ID Register (VERID)

53.3.1.2.1 Offset

Register	Offset
VERID	0h

53.3.1.2.2 Function

.

53.3.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

53.3.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design specification. Read-only field.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification. Read-only field.
15-0 FEATURE	Feature Specification Number Returns the feature set number. Read-only field. 0000000000000010b - Master only, with standard feature set 0000000000000011b - Master and slave, with standard feature set

53.3.1.3 Parameter Register (PARAM)

53.3.1.3.1 Offset

Register	Offset
PARAM	4h

53.3.1.3.2 Function

.

53.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MRXFIFO				0				MTXFIFO			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

53.3.1.3.4 Fields

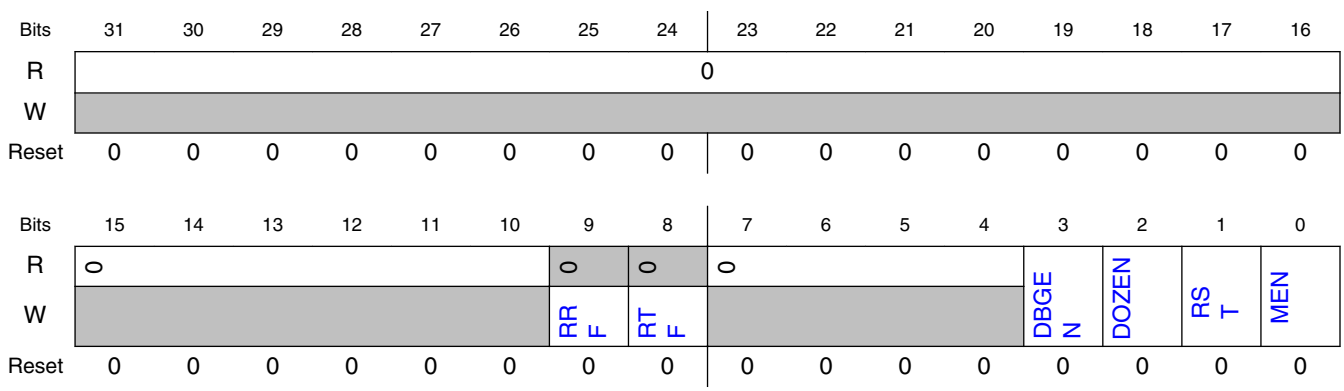
Field	Function
31-16 —	Reserved
15-12 —	Reserved
11-8 MRXFIFO	Master Receive FIFO Size Configures the number of words in the master receive FIFO to $2^{MRXFIFO}$
7-4 —	Reserved
3-0 MTXFIFO	Master Transmit FIFO Size Configures the number of words in the master transmit FIFO to $2^{MTXFIFO}$

53.3.1.4 Master Control Register (MCR)

53.3.1.4.1 Offset

Register	Offset
MCR	10h

53.3.1.4.2 Diagram



53.3.1.4.3 Fields

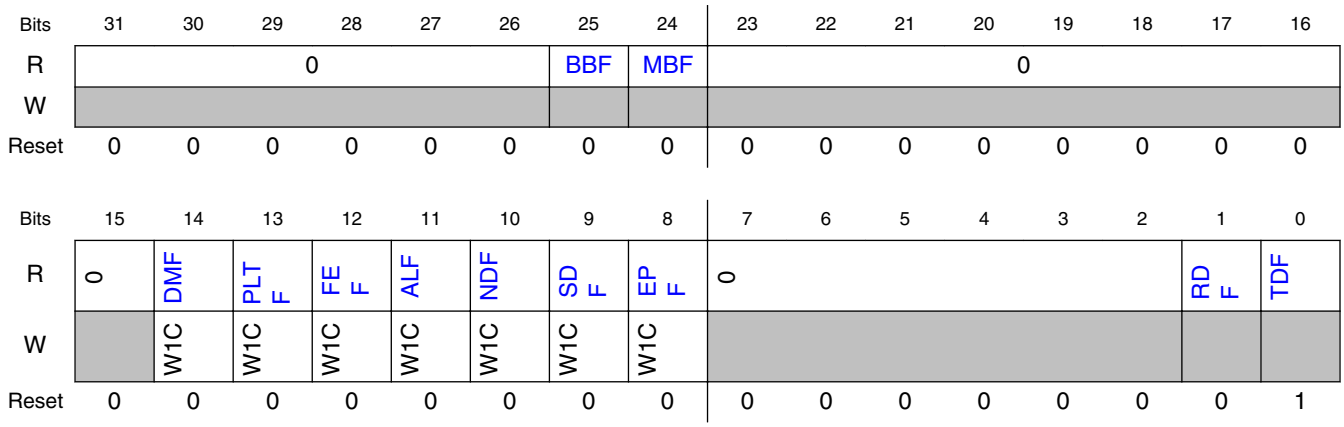
Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO 0b - No effect 1b - Receive FIFO is reset
8 RTF	Reset Transmit FIFO 0b - No effect 1b - Transmit FIFO is reset
7-4 —	Reserved
3 DBGEN	Debug Enable 0b - Master is disabled in debug mode 1b - Master is enabled in debug mode
2 DOZEN	Doze mode enable Enables or disables the master in Doze mode 0b - Master is enabled in Doze mode 1b - Master is disabled in Doze mode
1 RST	Software Reset Reset all internal master logic and registers, except the Master Control Register. RST remains set until cleared by software. 0b - Master logic is not reset 1b - Master logic is reset
0 MEN	Master Enable 0b - Master logic is disabled 1b - Master logic is enabled

53.3.1.5 Master Status Register (MSR)

53.3.1.5.1 Offset

Register	Offset
MSR	14h

53.3.1.5.2 Diagram



53.3.1.5.3 Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag 0b - I2C Bus is idle 1b - I2C Bus is busy
24 MBF	Master Busy Flag 0b - I2C Master is idle 1b - I2C Master is busy
23-15 —	Reserved
14 DMF	Data Match Flag Indicates that the received data has matched the MATCH0 and/or MATCH1 fields (as configured by MCFGR1[MATCFG], Match Configuration). Received data *that is discarded due to CMD field* does not cause Data Match Flag to set. 0b - Have not received matching data 1b - Have received matching data
13 PLTF	Pin Low Timeout Flag Will set when the SCL and/or SDA input is low for more than PINLOW cycles (Pin Low Timeout, MCFGR3[PINLOW]), even when the LPI2C master is idle. • Software is responsible for resolving the pin low condition. • Pin Low Timeout Flag cannot be cleared as long as the pin low timeout continues. • Before the LPI2C can initiate a START condition, the Pin Low Timeout Flag must be cleared. 0b - Pin low timeout has not occurred or is disabled 1b - Pin low timeout has occurred
12 FEF	FIFO Error Flag Detects an attempt to send or receive data without first generating a (repeated) START condition. This can occur if the transmit FIFO underflows when the AUTOSTOP bit is set. When FIFO Error Flag is set, the LPI2C master will send a STOP condition (if busy), and will not initiate a new START condition until FIFO Error Flag has been cleared.

Table continues on the next page...

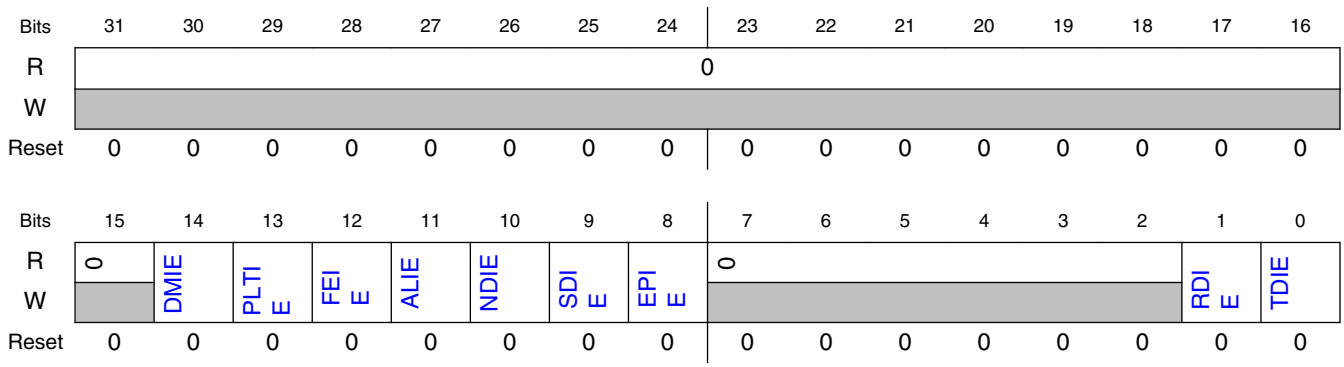
Field	Function
	0b - No error 1b - Master sending or receiving data without a START condition
11 ALF	Arbitration Lost Flag Set: • if the LPI2C master transmits a logic one and detects a logic zero on the I2C bus • or if the LPI2C master detects a START or STOP condition while the LPI2C master is transmitting data When the Arbitration Lost Flag sets, the LPI2C master will release the I2C bus (go idle), and the LPI2C master will not initiate a new START condition until the Arbitration Lost Flag has been cleared. 0b - Master has not lost arbitration 1b - Master has lost arbitration
10 NDF	NACK Detect Flag Set if the LPI2C master detects a NACK when transmitting an address or data. When set, the master will transmit a STOP condition and will not initiate a new START condition until NACK Detect Flag has been cleared. If a NACK is expected for a given address (as configured by the command word), then the NACK Detect Flag will set if a NACK is not generated. 0b - Unexpected NACK was not detected 1b - Unexpected NACK was detected
9 SDF	STOP Detect Flag Set when the LPI2C master generates a STOP condition. 0b - Master has not generated a STOP condition 1b - Master has generated a STOP condition
8 EPF	End Packet Flag Set when the LPI2C master generates either a repeated START condition or a STOP condition. Does not set when the master first generates a START condition. 0b - Master has not generated a STOP or Repeated START condition 1b - Master has generated a STOP or Repeated START condition
7-2 —	Reserved
1 RDF	Receive Data Flag The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. 0b - Receive Data is not ready 1b - Receive data is ready
0 TDF	Transmit Data Flag The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. 0b - Transmit data is not requested 1b - Transmit data is requested

53.3.1.6 Master Interrupt Enable Register (MIER)

53.3.1.6.1 Offset

Register	Offset
MIER	18h

53.3.1.6.2 Diagram



53.3.1.6.3 Fields

Field	Function
31-15 —	Reserved
14 DMIE	Data Match Interrupt Enable 0b - Disabled 1b - Enabled
13 PLTIE	Pin Low Timeout Interrupt Enable 0b - Disabled 1b - Enabled
12 FEIE	FIFO Error Interrupt Enable 0b - Enabled 1b - Disabled
11 ALIE	Arbitration Lost Interrupt Enable 0b - Disabled 1b - Enabled
10 NDIE	NACK Detect Interrupt Enable 0b - Disabled 1b - Enabled
9 SDIE	STOP Detect Interrupt Enable 0b - Disabled 1b - Enabled
8 EPIE	End Packet Interrupt Enable 0b - Disabled 1b - Enabled

Table continues on the next page...

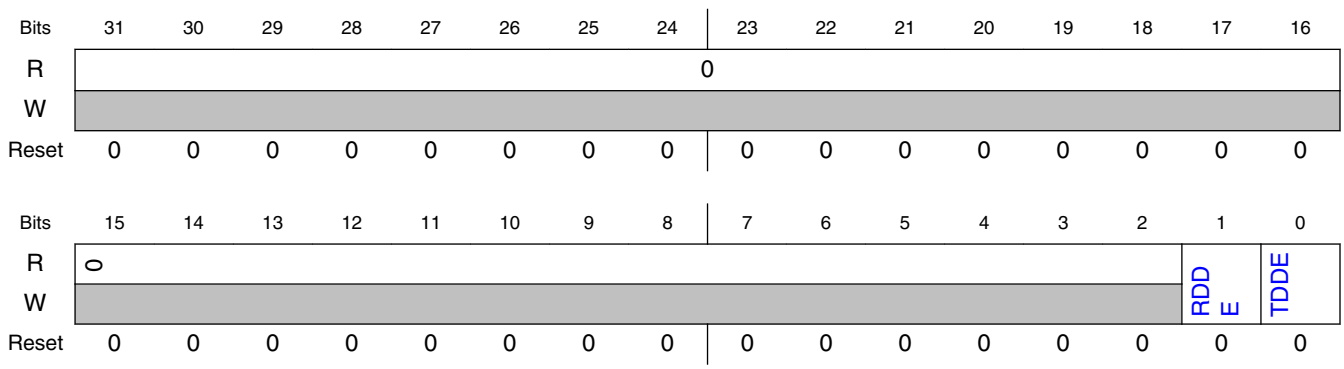
Field	Function
7-2 —	Reserved
1 RDIE	Receive Data Interrupt Enable 0b - Disabled 1b - Enabled
0 TDIE	Transmit Data Interrupt Enable 0b - Disabled 1b - Enabled

53.3.1.7 Master DMA Enable Register (MDER)

53.3.1.7.1 Offset

Register	Offset
MDER	1Ch

53.3.1.7.2 Diagram



53.3.1.7.3 Fields

Field	Function
31-2 —	Reserved
1 RDDE	Receive Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled
0	Transmit Data DMA Enable

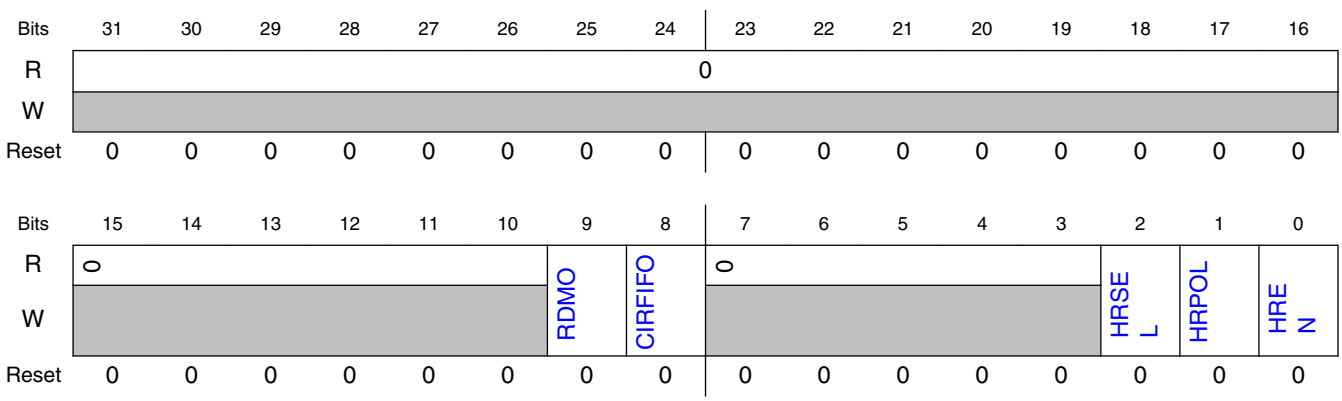
Field	Function
TDDE	0b - DMA request is disabled 1b - DMA request is enabled

53.3.1.8 Master Configuration Register 0 (MCFGR0)

53.3.1.8.1 Offset

Register	Offset
MCFGR0	20h

53.3.1.8.2 Diagram



53.3.1.8.3 Fields

Field	Function
31-10 —	Reserved
9 RDMO	Receive Data Match Only When enabled, all received data that does not cause the Data Match Flag (MSR[DMF]) to set is discarded. After the Data Match Flag is set, the RDMO configuration is ignored. When disabling RDMO, clear RDMO before clearing the Data Match Flag, to ensure that no receive data is lost. 0b - Received data is stored in the receive FIFO 1b - Received data is discarded unless the the Data Match Flag (MSR[DMF]) is set
8 CIRFIFO	Circular FIFO Enable When enabled, the transmit FIFO read pointer is saved to a temporary register. The transmit FIFO will be emptied as normal, but after the LPI2C master is idle and the transmit FIFO is empty, then the read pointer value will be restored from the temporary register. This will cause the contents of the transmit

Table continues on the next page...

Field	Function
	FIFO to be cycled through repeatedly. If AUTOSTOP is set, then a STOP condition will be sent whenever the transmit FIFO is empty and the read pointer is restored. 0b - Circular FIFO is disabled 1b - Circular FIFO is enabled
7-3 —	Reserved
2 HRSEL	Host Request Select Selects the source of the host request input. When host request input is enabled, the Host Request Select field should remain static (the Host Request Select should not change). 0b - Host request input is pin HREQ 1b - Host request input is input trigger
1 HRPOL	Host Request Polarity Configures the polarity of the host request input pin. When host request input is enabled, the Host Request Polarity field should remain static (Host Request Polarity should not change). 0b - Active low 1b - Active high
0 HREN	Host Request Enable When enabled, the LPI2C master will only initiate a START condition if the host request input is asserted and the bus is idle. A repeated START is not affected by the host request. 0b - Host request input is disabled 1b - Host request input is enabled

53.3.1.9 Master Configuration Register 1 (MCFGR1)

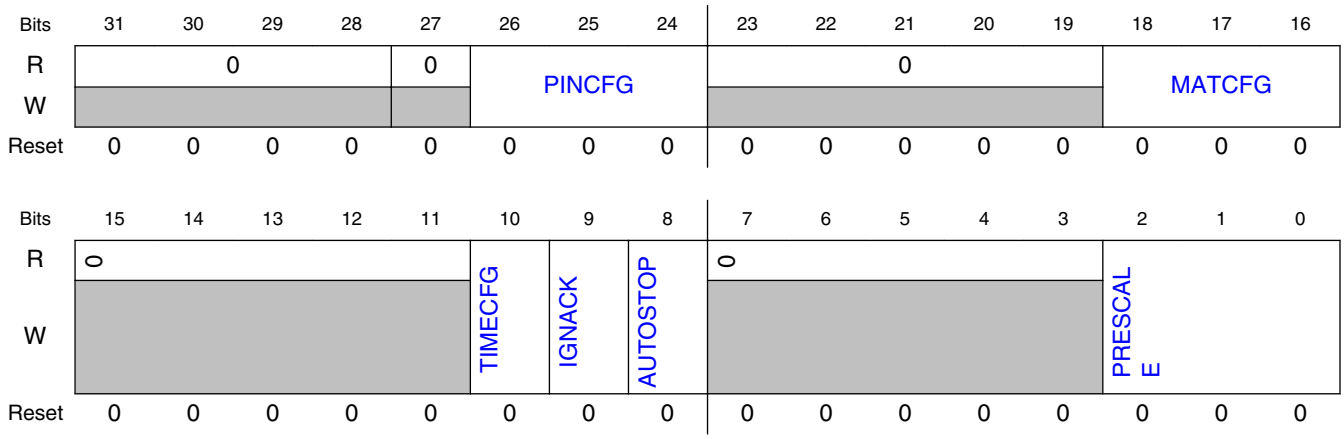
53.3.1.9.1 Offset

Register	Offset
MCFGR1	24h

53.3.1.9.2 Function

The MCFGR1 should only be written when the I2C Master is disabled.

53.3.1.9.3 Diagram



53.3.1.9.4 Fields

Field	Function
31-28 —	Reserved
27 —	Reserved
26-24 PINCFG	Pin Configuration Configures the pin mode for LPI2C.

Table 53-5. 2-pin / 4-pin pin configurations for masters and slaves

PINCFG	SCL / SDA pins	SCLS / SDAS pins
000	Bi-directional open drain for master and slave	Not used
001	Output-only (ultra-fast mode) open drain for master and slave	Not used
010	Bi-directional push-pull for master and slave	Not used
011	Input only for master and slave	Output-only push-pull for master and slave
100	Bi-directional open drain for master	Bi-directional open drain for slave
101	Output-only (ultra-fast mode) open drain for master	Output-only open drain for slave
110	Bi-directional push-pull for master	Bi-directional push-pull for slave
111	Input only for master and slave	Inverted output-only push-pull for master and slave

000b - 2-pin open drain mode
001b - 2-pin output only mode (ultra-fast mode)
010b - 2-pin push-pull mode
011b - 4-pin push-pull mode

Table continues on the next page...

Field	Function
	100b - 2-pin open drain mode with separate LPI2C slave 101b - 2-pin output only mode (ultra-fast mode) with separate LPI2C slave 110b - 2-pin push-pull mode with separate LPI2C slave 111b - 4-pin push-pull mode (inverted outputs)
23-19 —	Reserved
18-16 MATCFG	Match Configuration Configures the condition that will cause the DMF to set. 000b - Match is disabled 001b - Reserved 010b - Match is enabled (1st data word equals MATCH0 OR MATCH1) 011b - Match is enabled (any data word equals MATCH0 OR MATCH1) 100b - Match is enabled (1st data word equals MATCH0 AND 2nd data word equals MATCH1) 101b - Match is enabled (any data word equals MATCH0 AND next data word equals MATCH1) 110b - Match is enabled (1st data word AND MATCH1 equals MATCH0 AND MATCH1) 111b - Match is enabled (any data word AND MATCH1 equals MATCH0 AND MATCH1)
15-11 —	Reserved
10 TIMECFG	Timeout Configuration 0b - Pin Low Timeout Flag will set if SCL is low for longer than the configured timeout 1b - Pin Low Timeout Flag will set if either SCL or SDA is low for longer than the configured timeout
9 IGNACK	IGNACK When set, the received NACK field is ignored and assumed to be ACK. IGNACK bit is required to be set in Ultra-Fast Mode. 0b - LPI2C Master will receive ACK and NACK normally 1b - LPI2C Master will treat a received NACK as if it (NACK) was an ACK
8 AUTOSTOP	Automatic STOP Generation When enabled, a STOP condition is generated whenever the LPI2C master is busy and the transmit FIFO is empty. The STOP condition can also be generated using a transmit FIFO command. 0b - No effect 1b - STOP condition is automatically generated whenever the transmit FIFO is empty and the LPI2C master is busy
7-3 —	Reserved
2-0 PRESCALE	Prescaler Configures the clock prescaler used for all LPI2C master logic, except for the digital glitch filters. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

53.3.1.10 Master Configuration Register 2 (MCFGR2)

53.3.1.10.1 Offset

Register	Offset
MCFGR2	28h

53.3.1.10.2 Function

The Master Configuration Register 2 should only be written when the I2C Master is disabled.

53.3.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				BUSIDLE											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.10.4 Fields

Field	Function
31-28 —	Reserved
27-24 FILTSDA	Glitch Filter SDA Configures the I2C master digital glitch filters for SDA input. • A configuration of 0 will disable the glitch filter • Glitches equal to or less than FILTSDA cycles long will be filtered out and ignored • The latency through the glitch filter is equal to FILTSDA cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is automatically bypassed in High Speed mode
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C master digital glitch filters for SCL input.

Table continues on the next page...

Field	Function
	• A configuration of 0 will disable the glitch filter • Glitches equal to or less than FILTSCL cycles long will be filtered out and ignored • The latency through the glitch filter is equal to FILTSCL cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is automatically bypassed in High Speed mode
15-12 —	Reserved
11-0 BUSIDLE	Bus Idle Timeout Configures the bus idle timeout period in clock cycles. • If both SCL and SDA are high for longer than BUSIDLE cycles, then the I2C bus is assumed to be idle and the master can generate a START condition • When Bus Idle Timeout is set to zero, the Bus Idle Timeout is disabled

53.3.1.11 Master Configuration Register 3 (MCFGR3)

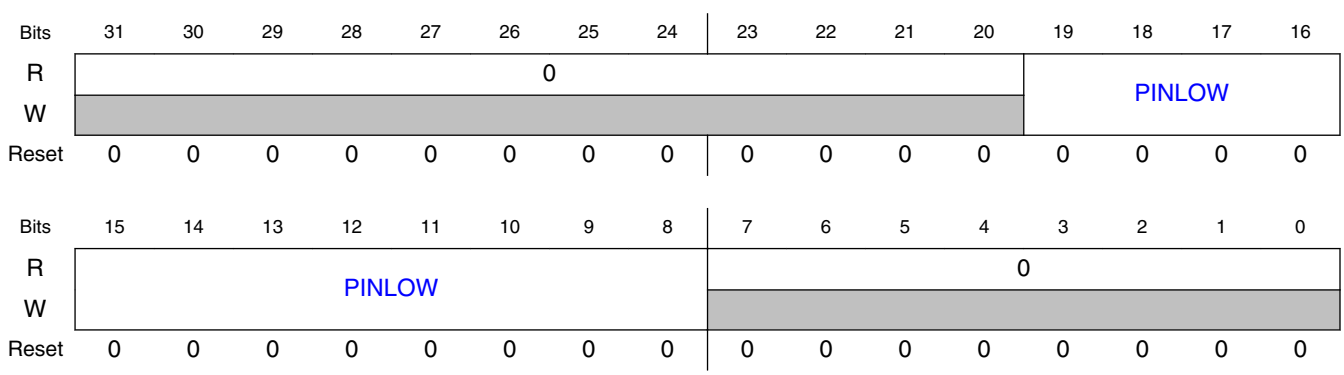
53.3.1.11.1 Offset

Register	Offset
MCFGR3	2Ch

53.3.1.11.2 Function

The MCFGR3 should only be written when the I2C Master is disabled.

53.3.1.11.3 Diagram



53.3.1.11.4 Fields

Field	Function
31-20 —	Reserved
19-8 PINLOW	Pin Low Timeout Configures the pin low timeout flag in clock cycles. • If SCL or, either SCL or SDA, is low for longer than (PINLOW * 256) cycles, then PLTF is set • When Pin Low Timeout is set to zero, the Pin Low Timeout feature is disabled
7-0 —	Reserved

53.3.1.12 Master Data Match Register (MDMR)

53.3.1.12.1 Offset

Register	Offset
MDMR	40h

53.3.1.12.2 Function

The MDMR should only be written when the I2C Master is disabled or idle.

53.3.1.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								MATCH1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								MATCH0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.12.4 Fields

Field	Function
31-24 —	Reserved
23-16 MATCH1	Match 1 Value Compared against the received data when receive data match is enabled.
15-8 —	Reserved
7-0 MATCH0	Match 0 Value Compared against the received data when receive data match is enabled.

53.3.1.13 Master Clock Configuration Register 0 (MCCR0)

53.3.1.13.1 Offset

Register	Offset
MCCR0	48h

53.3.1.13.2 Function

The MCCR0 cannot be changed when the I2C master is enabled and is used for standard, fast, fast-mode plus and ultra-fast transfers.

53.3.1.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DATAVD						0		SETHOLD					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		CLKHI						0		CLKLO					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.13.4 Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. Must be configured less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master • as the hold time for a START condition • as the setup and hold time for a repeated START condition • as the setup time for a STOP condition The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. The Clock Low Period value is also used for the minimum bus free time between a STOP and a START condition; this is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.

53.3.1.14 Master Clock Configuration Register 1 (MCCR1)

53.3.1.14.1 Offset

Register	Offset
MCCR1	50h

53.3.1.14.2 Function

The MCCR1 cannot be changed when the I2C master is enabled and is used for high speed mode transfers. The separate clock configuration for high speed mode allows arbitration to take place in Fast mode (with timing configured by MCCR0), before switching to high speed mode (with timing configured by MCCR1).

53.3.1.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		DATAVD						0		SETHOLD					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		CLKHI						0		CLKLO					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.14.4 Fields

Field	Function
31-30 —	Reserved
29-24 DATAVD	Data Valid Delay Minimum number of cycles (minus one) that is used as the data hold time for SDA. The Data Valid Delay must be configured to be less than the minimum SCL low period.
23-22 —	Reserved
21-16 SETHOLD	Setup Hold Delay Minimum number of cycles (minus one) that is used by the master • as the hold time for a START condition • as the setup and hold time for a repeated START condition • as the setup time for a STOP condition The setup time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.
15-14 —	Reserved
13-8 CLKHI	Clock High Period Minimum number of cycles (minus one) that the SCL clock is driven high by the master. The SCL high time is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.

Table continues on the next page...

Field	Function
7-6 —	Reserved
5-0 CLKLO	Clock Low Period Minimum number of cycles (minus one) that the SCL clock is driven low by the master. The Clock Low Period value is also used for the minimum bus free time between a STOP and a START condition; this is extended by the time it takes to detect a rising edge on the external SCL pin. Ignoring any additional board delay due to external loading, this time is equal to (2 + FILTSCL) / 2^PRESCALE cycles.

53.3.1.15 Master FIFO Control Register (MFCR)

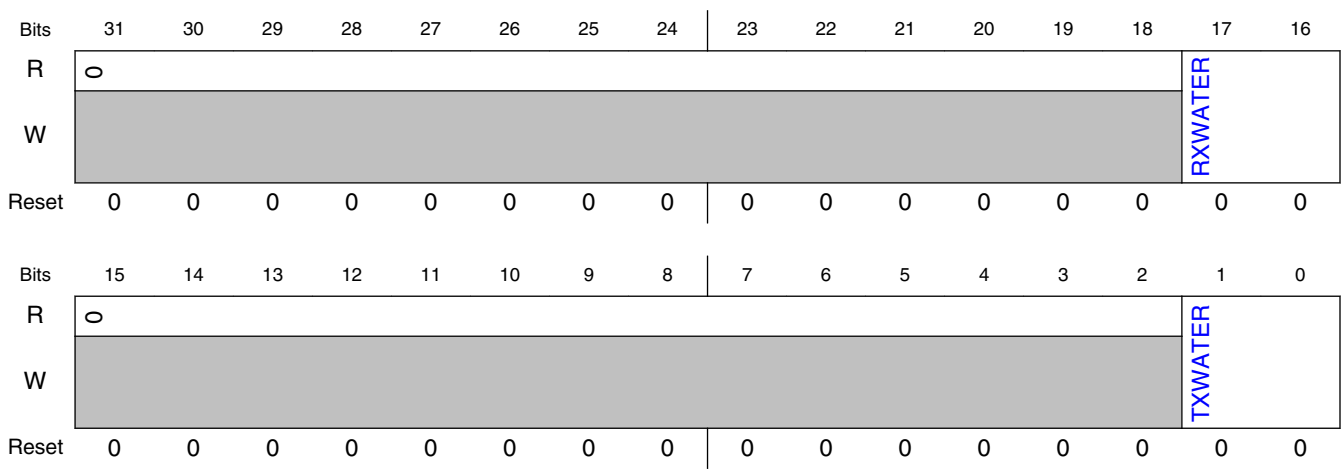
53.3.1.15.1 Offset

Register	Offset
MFCR	58h

53.3.1.15.2 Function

The Master FIFO control register is only used in Stop mode when the MFCR register is static (i.e., the MFCR register is not changing).

53.3.1.15.3 Diagram



53.3.1.15.4 Fields

Field	Function
31-18 —	Reserved
17-16 RXWATER	Receive FIFO Watermark The Receive Data Flag is set whenever the number of words in the receive FIFO is greater than RXWATER. Writing a value equal to or greater than the FIFO size will be truncated.
15-2 —	Reserved
1-0 TXWATER	Transmit FIFO Watermark The Transmit Data Flag is set whenever the number of words in the transmit FIFO is equal or less than TXWATER. Writing a value equal to or greater than the FIFO size will be truncated.

53.3.1.16 Master FIFO Status Register (MFSR)

53.3.1.16.1 Offset

Register	Offset
MFSR	5Ch

53.3.1.16.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													RXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													TXCOUNT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.16.3 Fields

Field	Function
31-19	Reserved

Table continues on the next page...

Field	Function
—	
18-16 RXCOUNT	Receive FIFO Count Returns the number of words in the receive FIFO.
15-3 —	Reserved
2-0 TXCOUNT	Transmit FIFO Count Returns the number of words in the transmit FIFO.

53.3.1.17 Master Transmit Data Register (MTDR)

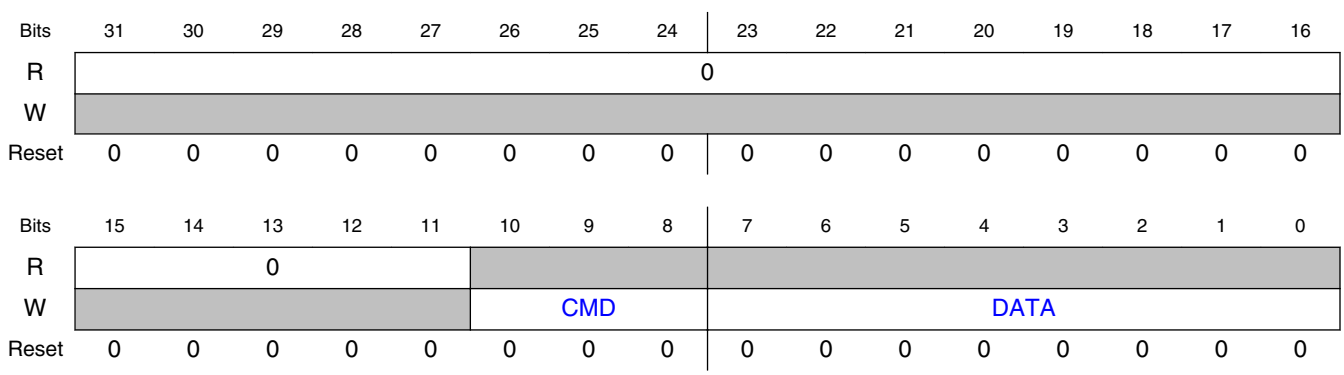
53.3.1.17.1 Offset

Register	Offset
MTDR	60h

53.3.1.17.2 Function

- An 8-bit write to the CMD field will store the data in the Command FIFO, but does not increment the FIFO write pointer.
- An 8-bit write to the DATA field will zero extend the CMD field, unless the CMD field has been written separately since the last FIFO write; it (the 8-bit write) also increments the FIFO write pointer.
- A 16-bit or 32-bit will write both the CMD and DATA fields and increment the FIFO.

53.3.1.17.3 Diagram



53.3.1.17.4 Fields

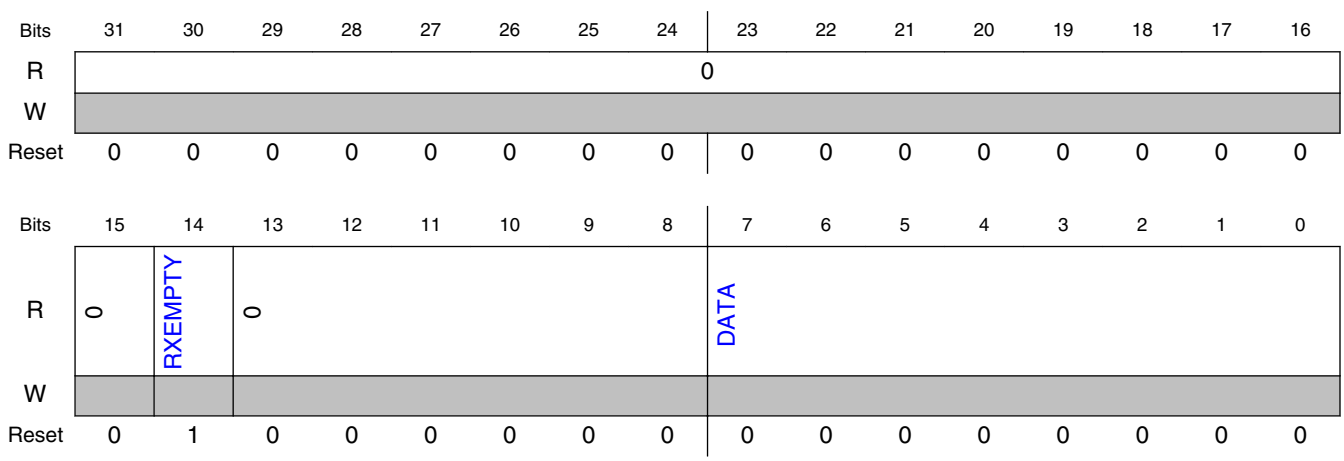
Field	Function
31-11 —	Reserved
10-8 CMD	Command Data 000b - Transmit DATA[7:0] 001b - Receive (DATA[7:0] + 1) bytes 010b - Generate STOP condition 011b - Receive and discard (DATA[7:0] + 1) bytes 100b - Generate (repeated) START and transmit address in DATA[7:0] 101b - Generate (repeated) START and transmit address in DATA[7:0]. This transfer expects a NACK to be returned. 110b - Generate (repeated) START and transmit address in DATA[7:0] using high speed mode 111b - Generate (repeated) START and transmit address in DATA[7:0] using high speed mode. This transfer expects a NACK to be returned.
7-0 DATA	Transmit Data Performing an 8-bit write to DATA will zero extend the CMD field.

53.3.1.18 Master Receive Data Register (MRDR)

53.3.1.18.1 Offset

Register	Offset
MRDR	70h

53.3.1.18.2 Diagram



53.3.1.18.3 Fields

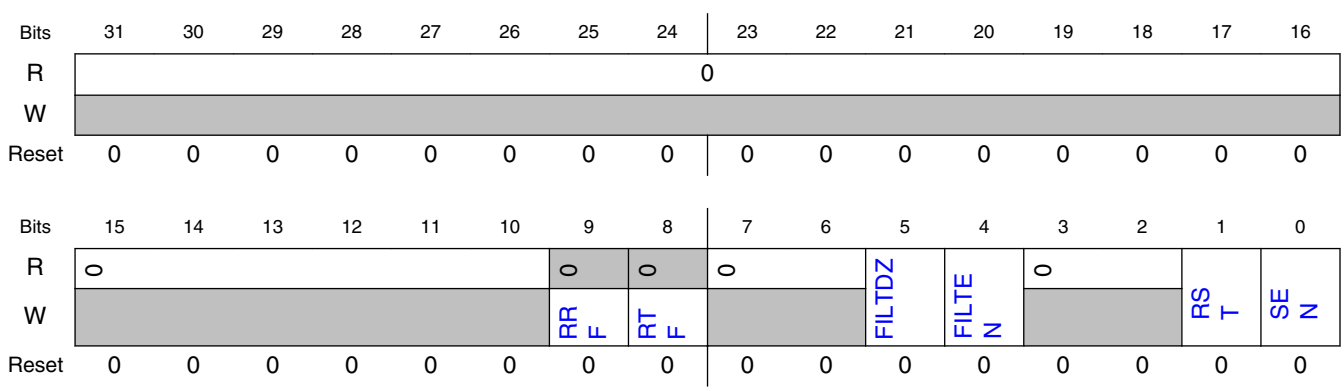
Field	Function
31-15 —	Reserved
14 RXEMPTY	RX Empty 0b - Receive FIFO is not empty 1b - Receive FIFO is empty
13-8 —	Reserved
7-0 DATA	Receive Data Reading the Receive Data register returns the data received by the I2C master that has not been discarded. Receive data can be discarded due to the CMD field, or the master can be configured to discard non-matching data.

53.3.1.19 Slave Control Register (SCR)

53.3.1.19.1 Offset

Register	Offset
SCR	110h

53.3.1.19.2 Diagram



53.3.1.19.3 Fields

Field	Function
31-10 —	Reserved
9 RRF	Reset Receive FIFO 0b - No effect 1b - Receive Data Register is now empty
8 RTF	Reset Transmit FIFO 0b - No effect 1b - Transmit Data Register is now empty
7-6 —	Reserved
5 FILTDZ	Filter Doze Enable Filter Doze Enable bit should only be updated when the I2C Slave is disabled. 0b - Filter remains enabled in Doze mode 1b - Filter is disabled in Doze mode
4 FILTEN	Filter Enable Filter Enable bit should only be updated when the I2C Slave is disabled. 0b - Disable digital filter and output delay counter for slave mode 1b - Enable digital filter and output delay counter for slave mode
3-2 —	Reserved
1 RST	Software Reset 0b - Slave mode logic is not reset 1b - Slave mode logic is reset
0 SEN	Slave Enable 0b - I2C Slave mode is disabled 1b - I2C Slave mode is enabled

53.3.1.20 Slave Status Register (SSR)

53.3.1.20.1 Offset

Register	Offset
SSR	114h

53.3.1.20.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						BBF	SBF	0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SARF	GCF	AM1F	AM0F	FEF	BEF	SDF	RSF	0				TAF	AVF	RDF	TDF
W					W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.20.3 Fields

Field	Function
31-26 —	Reserved
25 BBF	Bus Busy Flag Indicates if an I2C bus is idle or busy. 0b - I2C Bus is idle 1b - I2C Bus is busy
24 SBF	Slave Busy Flag Indicates if an I2C slave is idle or busy. 0b - I2C Slave is idle 1b - I2C Slave is busy
23-16 —	Reserved
15 SARF	SMBus Alert Response Flag • SMBus Alert Response Flag is cleared by reading the Address Status Register • SMBus Alert Response Flag cannot generate an asynchronous wakeup 0b - SMBus Alert Response is disabled or not detected 1b - SMBus Alert Response is enabled and detected
14 GCF	General Call Flag Indicates whether a slave has detected the General Call Address. • General Call Flag is cleared by reading the Address Status Register • General Call Flag cannot generate an asynchronous wakeup 0b - Slave has not detected the General Call Address or the General Call Address is disabled 1b - Slave has detected the General Call Address
13 AM1F	Address Match 1 Flag Indicates that the received address has matched the ADDR1 field or ADDR0 to ADDR1 range as configured by ADDRCFG.

Table continues on the next page...

Field	Function
	• Address Match 1 Flag is cleared by reading the Address Status Register • Address Match 1 Flag cannot generate an asynchronous wakeup 0b - Have not received an ADDR1 or ADDR0/ADDR1 range matching address 1b - Have received an ADDR1 or ADDR0/ADDR1 range matching address
12 AM0F	Address Match 0 Flag Indicates that the received address has matched the ADDR0 field as configured by ADDRCFG. • Address Match 0 Flag is cleared by reading the Address Status Register • Address Match 0 Flag cannot generate an asynchronous wakeup 0b - Have not received an ADDR0 matching address 1b - Have received an ADDR0 matching address
11 FEF	FIFO Error Flag FIFO error flag can only be set when clock stretching is disabled. 0b - FIFO underflow or overflow was not detected 1b - FIFO underflow or overflow was detected
10 BEF	Bit Error Flag Bit Error Flag will set if the LPI2C slave transmits a logic one and detects a logic zero on the I2C bus. The slave will ignore the rest of the transfer until the next (repeated) START condition. 0b - Slave has not detected a bit error 1b - Slave has detected a bit error
9 SDF	STOP Detect Flag STOP Detect Flag will set when the LPI2C slave detects a STOP condition and if the LPI2C slave matched the last address byte. 0b - Slave has not detected a STOP condition 1b - Slave has detected a STOP condition
8 RSF	Repeated Start Flag Repeated Start Flag will set when the LPI2C slave detects a repeated START condition and if the LPI2C slave matched the last address byte. The Repeated Start Flag does not set when the slave first detects a START condition. 0b - Slave has not detected a Repeated START condition 1b - Slave has detected a Repeated START condition
7-4 —	Reserved
3 TAF	Transmit ACK Flag Transmit ACK Flag is cleared by writing the Transmit ACK register. 0b - Transmit ACK/NACK is not required 1b - Transmit ACK/NACK is required
2 AVF	Address Valid Flag Address Valid Flag is cleared by reading the address status register. When Receive Data Configuration (SCFGR1[RXCFG]) is set, the Address Valid Flag is also cleared by reading the Receive Data register. 0b - Address Status Register is not valid 1b - Address Status Register is valid
1 RDF	Receive Data Flag Receive Data Flag is cleared by reading the receive data register. When Receive Data Configuration (SCFGR1[RXCFG]) is set, the Receive Data Flag is not cleared when reading the Receive Data register and if AVF is set. 0b - Receive data is not ready 1b - Receive data is ready

Table continues on the next page...

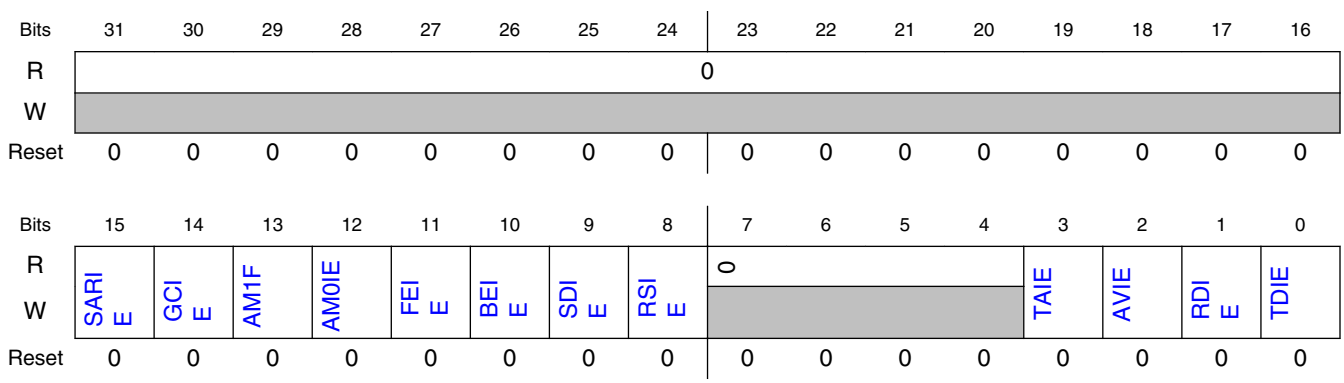
Field	Function
0 TDF	Transmit Data Flag Transmit Data Flag is cleared by writing the Transmit Data register. When Transmit Flag Configuration (TXCFG) is clear, and if a NACK or Repeated START or STOP condition is detected, then Transmit Data Flag is also cleared. 0b - Transmit data not requested 1b - Transmit data is requested

53.3.1.21 Slave Interrupt Enable Register (SIER)

53.3.1.21.1 Offset

Register	Offset
SIER	118h

53.3.1.21.2 Diagram



53.3.1.21.3 Fields

Field	Function
31-16 —	Reserved
15 SARIE	SMBus Alert Response Interrupt Enable 0b - Disabled 1b - Enabled
14 GCIE	General Call Interrupt Enable 0b - Disabled 1b - Enabled

Table continues on the next page...

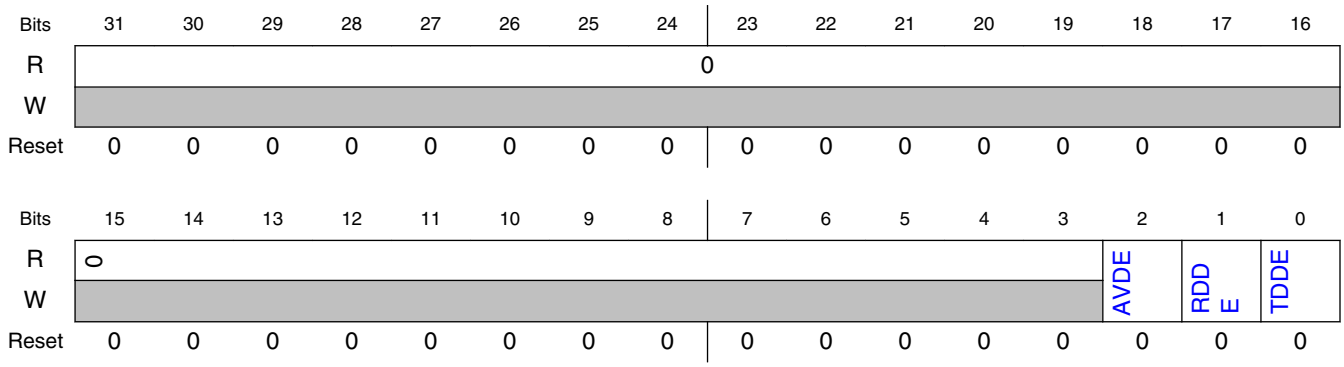
Field	Function
13 AM1F	Address Match 1 Interrupt Enable 0b - Disabled 1b - Enabled
12 AM0IE	Address Match 0 Interrupt Enable 0b - Enabled 1b - Disabled
11 FEIE	FIFO Error Interrupt Enable 0b - Disabled 1b - Enabled
10 BEIE	Bit Error Interrupt Enable 0b - Disabled 1b - Enabled
9 SDIE	STOP Detect Interrupt Enable 0b - Disabled 1b - Enabled
8 RSIE	Repeated Start Interrupt Enable 0b - Disabled 1b - Enabled
7-4 —	Reserved
3 TAIE	Transmit ACK Interrupt Enable 0b - Disabled 1b - Enabled
2 AVIE	Address Valid Interrupt Enable 0b - Disabled 1b - Enabled
1 RDIE	Receive Data Interrupt Enable 0b - Disabled 1b - Enabled
0 TDIE	Transmit Data Interrupt Enable 0b - Disabled 1b - Enabled

53.3.1.22 Slave DMA Enable Register (SDER)

53.3.1.22.1 Offset

Register	Offset
SDER	11Ch

53.3.1.22.2 Diagram



53.3.1.22.3 Fields

Field	Function
31-3 —	Reserved
2 AVDE	Address Valid DMA Enable The Address Valid DMA request is shared with the Receive Data DMA request. If both Address Valid DMA request and Receive Data DMA request are enabled, then set Receive Data Configuration (SCFGR1[RXCFG]), to allow the DMA to read the address from the Receive Data Register. 0b - DMA request is disabled 1b - DMA request is enabled
1 RDDE	Receive Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled
0 TDDE	Transmit Data DMA Enable 0b - DMA request is disabled 1b - DMA request is enabled

53.3.1.23 Slave Configuration Register 1 (SCFGR1)

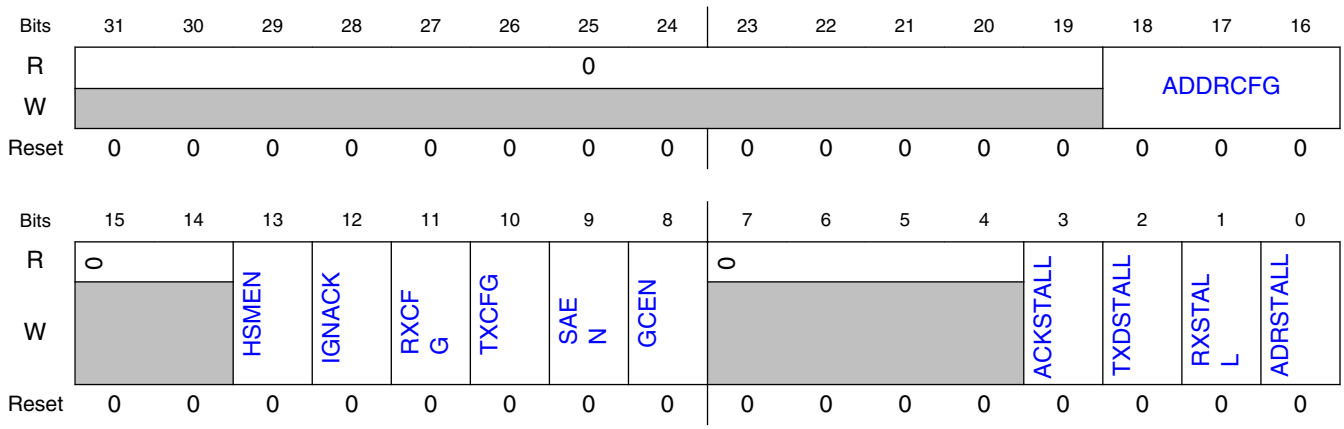
53.3.1.23.1 Offset

Register	Offset
SCFGR1	124h

53.3.1.23.2 Function

The Slave Configuration Register 1 should only be written when the I2C Slave is disabled.

53.3.1.23.3 Diagram



53.3.1.23.4 Fields

Field	Function
31-19 —	Reserved
18-16 ADDRCFG	Address Configuration Configures the condition that will cause an address to match. 000b - Address match 0 (7-bit) 001b - Address match 0 (10-bit) 010b - Address match 0 (7-bit) or Address match 1 (7-bit) 011b - Address match 0 (10-bit) or Address match 1 (10-bit) 100b - Address match 0 (7-bit) or Address match 1 (10-bit) 101b - Address match 0 (10-bit) or Address match 1 (7-bit) 110b - From Address match 0 (7-bit) to Address match 1 (7-bit) 111b - From Address match 0 (10-bit) to Address match 1 (10-bit)
15-14 —	Reserved
13 HSMEN	High Speed Mode Enable Enables detection of the High-Speed Mode master code of slave address 0000_1XX, but does not cause an address match on this code. When set and any HS-mode master code is detected, the FILTEN and ACKSTALL bits are ignored until the next STOP condition is detected. 0b - Disables detection of HS-mode master code 1b - Enables detection of HS-mode master code
12 IGNACK	Ignore NACK When Ignore NACK is set, the LPI2C slave will continue transfers after a NACK is detected. Ignore NACK bit is required to be set in Ultra-Fast Mode. 0b - Slave will end transfer when NACK is detected

Table continues on the next page...

Field	Function
	1b - Slave will not end transfer when NACK detected
11 RXCFG	Receive Data Configuration 0b - Reading the Receive Data register will return received data and clear the Receive Data flag (MSR[RDF]). 1b - Reading the Receive Data register when the Address Valid flag (SSR[AVF])is set, will return the Address Status register and clear the Address Valid flag. Reading the Receive Data register when the Address Valid flag is clear, will return received data and clear the Receive Data flag (MSR[RDF]).
10 TXCFG	Transmit Flag Configuration The transmit data flag will always assert before a NACK is detected at the end of a slave-transmit transfer. This can cause an extra word to be written to the transmit data FIFO. • When TXCFG=0, the Transmit Data register is automatically emptied when a slave-transmit transfer is detected. This causes the transmit data flag to assert whenever a slave-transmit transfer is detected, and causes the transmit data flag to negate at the end of the slave-transmit transfer. • When TXCFG=1, the Transmit Data flag will assert whenver the Transmit Data register is empty, and the Transmit Data flag will negate when the Transmit Data register is full. This allows the Transmit Data register to be filled before a slave-transmit transfer is detected, but can cause the Transmit Data register to be written before a NACK is detected on the last byte of a slave transmit transfer. 0b - Transmit Data Flag will only assert during a slave-transmit transfer when the Transmit Data register is empty 1b - Transmit Data Flag will assert whenever the Transmit Data register is empty
9 SAEN	SMBus Alert Enable 0b - Disables match on SMBus Alert 1b - Enables match on SMBus Alert
8 GCEN	General Call Enable 0b - General Call address is disabled 1b - General Call address is enabled
7-4 —	Reserved
3 ACKSTALL	ACK SCL Stall Enables SCL clock stretching during slave-transmit address byte(s) and slave-receiver address and data byte(s), to allow software to write the Transmit ACK Register before the ACK or NACK is transmitted. Clock stretching occurs when transmitting the 9th bit, and is therefore not compatible with high speed mode. When ACKSTALL is enabled, there is no need to set either RX SCL Stall (SCFGR1[RXSTALL]) or Address SCL Stall (SCFGR1[ADRSTALL]). 0b - Clock stretching is disabled 1b - Clock stretching is enabled
2 TXDSTALL	TX Data SCL Stall Enables SCL clock stretching when the Transmit Data flag (SSR[TDF]) is set during a slave-transmit transfer. Clock stretching occurs following the 9th bit, and is therefore compatible with high speed mode. 0b - Clock stretching is disabled 1b - Clock stretching is enabled
1 RXSTALL	RX SCL Stall Enables SCL clock stretching when the Receive Data flag (SSR[RDF]) is set during a slave-receive transfer. Clock stretching occurs following the 9th bit, and is therefore compatible with high speed mode. 0b - Clock stretching is disabled 1b - Clock stretching is enabled

Table continues on the next page...

Field	Function
0 ADRSTALL	Address SCL Stall Enables SCL clock stretching when the Address Valid Flag (SSR[AVF]) is asserted. Clock stretching only occurs following the 9th bit, and is therefore compatible with high speed mode. 0b - Clock stretching is disabled 1b - Clock stretching is enabled

53.3.1.24 Slave Configuration Register 2 (SCFGR2)

53.3.1.24.1 Offset

Register	Offset
SCFGR2	128h

53.3.1.24.2 Function

The Slave Configuration Register 2 should only be written when the I2C Slave is disabled.

53.3.1.24.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FILTSDA				0				FILTSCL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		DATAVD						0				CLKHOLD			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.24.4 Fields

Field	Function
31-28 —	Reserved
27-24	Glitch Filter SDA

Table continues on the next page...

Field	Function
FILTSDA	Configures the I2C slave digital glitch filters for SDA input. • A configuration of 0 will disable the glitch filter • Glitches equal to or less than FILTSDA cycles long will be filtered out and ignored • The latency through the glitch filter is equal to FILTSDA+3 cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is disabled in high speed mode
23-20 —	Reserved
19-16 FILTSCL	Glitch Filter SCL Configures the I2C slave digital glitch filters for SCL input. • A configuration of 0 will disable the glitch filter • Glitches equal to or less than FILTSCL cycles long will be filtered out and ignored • The latency through the glitch filter is equal to FILTSCL+3 cycles, and must be configured to be less than the minimum SCL low or high period • The glitch filter cycle count is not affected by the PRESCALE configuration, and the glitch filter cycle count is disabled in high speed mode
15-14 —	Reserved
13-8 DATAVD	Data Valid Delay Configures the SDA data valid delay time for the I2C slave, and is equal to FILTSCL+DATAVD+3 cycles. • The data valid delay must be configured to be less than the minimum SCL low period • The I2C slave data valid delay time is not affected by the PRESCALE configuration, and the I2C slave data valid delay time is disabled in high speed mode
7-4 —	Reserved
3-0 CLKHOLD	Clock Hold Time Configures the minimum clock hold time for the I2C slave, when clock stretching is enabled. • The minimum hold time is equal to CLKHOLD+3 cycles • The I2C slave clock hold time is not affected by the PRESCALE configuration, and the I2C slave clock hold time is disabled in high speed mode

53.3.1.25 Slave Address Match Register (SAMR)

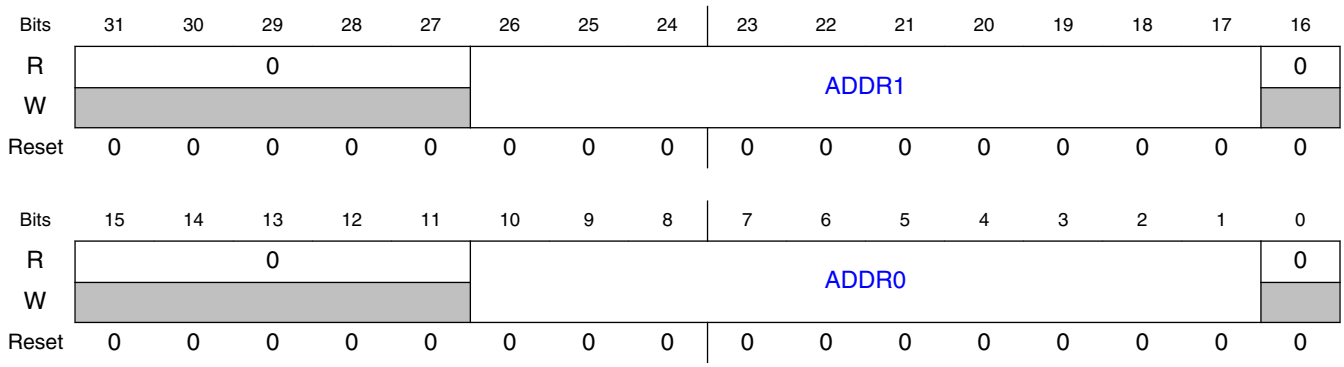
53.3.1.25.1 Offset

Register	Offset
SAMR	140h

53.3.1.25.2 Function

The SAMR should only be written when the I2C Slave is disabled.

53.3.1.25.3 Diagram



53.3.1.25.4 Fields

Field	Function
31-27 —	Reserved
26-17 ADDR1	Address 1 Value Compared against the received address to detect the Slave Address. • In 10-bit mode, the first address byte is compared to { 11110, ADDR1[26:25] } and the second address byte is compared to ADDR1[24:17] • In 7-bit mode, the address is compared to ADDR1[23:17]
16-11 —	Reserved
10-1 ADDR0	Address 0 Value Compared against the received address to detect the Slave Address. • In 10-bit mode, the first address byte is compared to { 11110, ADDR0[10:9] } and the second address byte is compared to ADDR0[8:1] • In 7-bit mode, the address is compared to ADDR0[7:1] •
0 —	Reserved

53.3.1.26 Slave Address Status Register (SASR)

53.3.1.26.1 Offset

Register	Offset
SASR	150h

53.3.1.26.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	ANV	0			RADDR										
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.26.3 Fields

Field	Function
31-15 —	Reserved
14 ANV	Address Not Valid 0b - Received Address (RADDR) is valid 1b - Received Address (RADDR) is not valid
13-11 —	Reserved
10-0 RADDR	Received Address The Received Address updates whenever the AMF is set; the AMF is cleared by reading the Slave Address Status register. • In 7-bit mode, the address byte is store in RADDR[7:0] • In 10-bit mode, the first address byte is { 11110, RADDR[10:9], RADDR[0] } and the second address byte is RADDR[8:1]. The R/W bit is therefore always stored in RADDR[0]

53.3.1.27 Slave Transmit ACK Register (STAR)

53.3.1.27.1 Offset

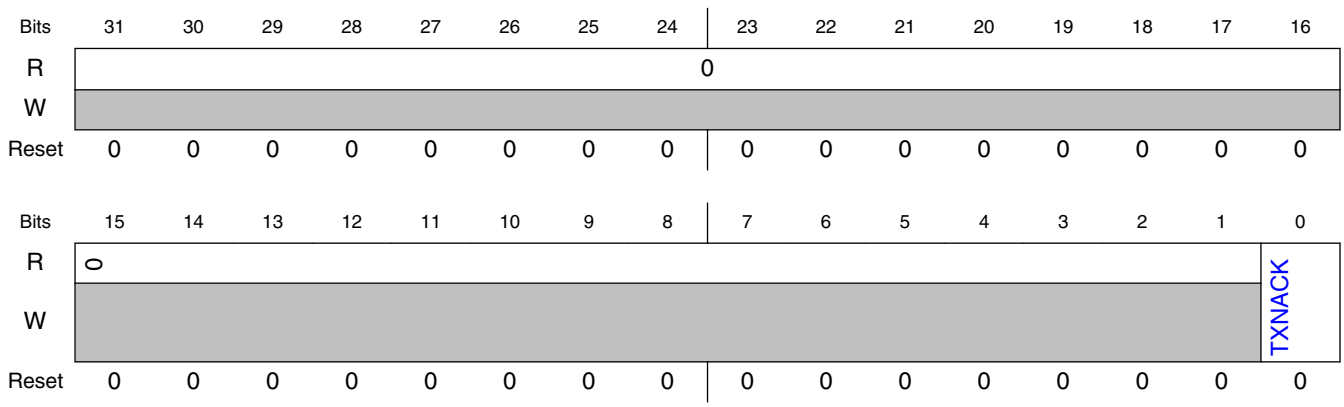
Register	Offset
STAR	154h

53.3.1.27.2 Function

The Slave Transmit ACK Register can only be written when the ACK SCL Stall bit is set in Slave Configuration Register 1 (SCFGR1[ACKSTALL].

- The ACKSTALL bit will enable clock stretching during the ACK/NACK bit slot, and during this time, the STAR register can be written by software.
- The logic ensures that the clock stretching continues for at least 1 bus clock cycle after the STAR register is updated.
- This clock stretching time can be extended more using the Clock Hold Time field (SCFGR2[CLKHOLD], Slave Configuration Register 2).

53.3.1.27.3 Diagram



53.3.1.27.4 Fields

Field	Function
31-1 —	Reserved
0 TXNACK	Transmit NACK After receiving each word, software can transmit either an ACK (logic 0) or a NACK (logic 1); Transmit NACK selects which to use: ACK or NACK. • When ACKSTALL is set, a Transmit NACK must be written once for each matching address byte and each received word. ACKSTALL must be set, because that will stall the data transfer until software reads the received word (and decides whether to respond with an ACK or NACK). • To configure the default ACK/NACK, Transmit NACK can also be written when LPI2C Slave is disabled or idle. 0b - Write a Transmit ACK for each received word 1b - Write a Transmit NACK for each received word

53.3.1.28 Slave Transmit Data Register (STDR)

53.3.1.28.1 Offset

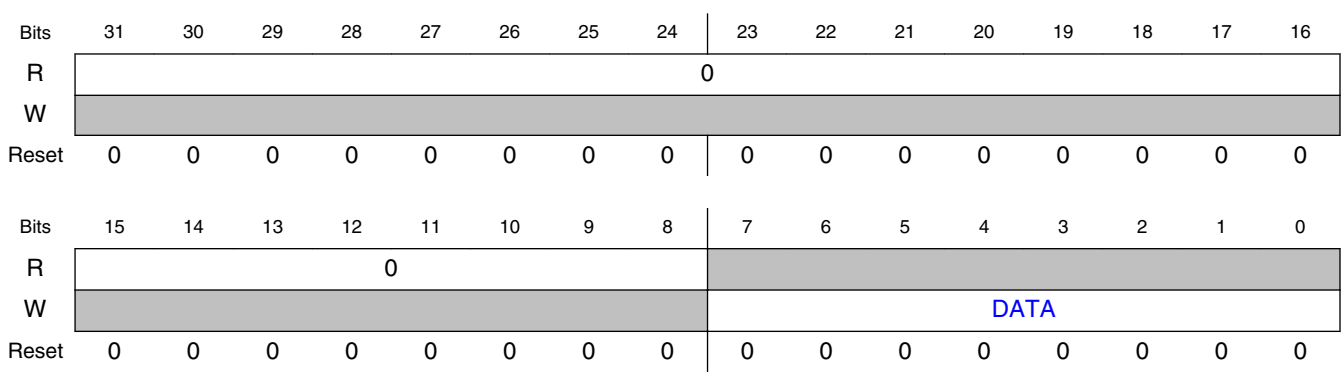
Register	Offset
STDR	160h

53.3.1.28.2 Function

Clock stretching (enabled or disabled) affects when the transmit data is transferred. The TXDSTALL bit will enable clock stretching during the 1st data bit of a slave-transmit transfer.

- **If clock stretching is enabled (TXDSTALL=1)**, then the transmit data transfer is stalled until the Slave Transmit register (STDR) is updated. Clock stretching is extended by at least 1 bus clock cycle after the Slave Transmit Data Register (STDR) is updated, and clock stretching can be delayed even more using the Clock Hold Time field (SCFGR2[CLKHOLD], Slave Configuration Register 2).
- **If clock stretching is disabled (TXDSTALL=0)**, then the transmit data should be written before the start of the slave-transmit transfer; otherwise (i.e., if the transmit data is not written before the start of the slave-transmit transfer), the FIFO Error Flag (SSR[FEF]) will be set.

53.3.1.28.3 Diagram



53.3.1.28.4 Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Field	Function
7-0 DATA	Transmit Data Writing to the Slave Transmit Data Register (STDR) will store I2C slave transmit data *in* the Slave Transmit Data Register

53.3.1.29 Slave Receive Data Register (SRDR)

53.3.1.29.1 Offset

Register	Offset
SRDR	170h

53.3.1.29.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SOF	RXEMPTY	0						DATA							
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

53.3.1.29.3 Fields

Field	Function
31-16 —	Reserved
15 SOF	Start Of Frame 0b - Indicates this is not the first data word since a (repeated) START or STOP condition 1b - Indicates this is the first data word since a (repeated) START or STOP condition
14 RXEMPTY	RX Empty 0b - The Receive Data Register is not empty 1b - The Receive Data Register is empty

Table continues on the next page...

Field	Function
13-8 —	Reserved
7-0 DATA	Receive Data Reading the Slave Receive Data Register returns the data received by the I2C slave

53.4 Functional description

53.4.1 Clocking and Resets

Table 53-6. Clocks

LPI2C Functional clock	The LPI2C functional clock is asynchronous to the bus clock and can remain enabled in low power modes to support I2C bus transfers by the LPI2C master. The functional clock is also used by the LPI2C slave to support digital filter and data hold time configurations. The LPI2C master divides the functional clock by a prescaler and the resulting frequency must be at least 8 times faster than the I2C bus bandwidth.
External clock	The LPI2C slave logic is clocked directly from the external pins SCL and SDA (or SCLS and SDAS if master and slave are implemented on separate pins). This allows the LPI2C slave to remain operational, even when the LPI2C functional clock is disabled. **NOTE:** The LPI2C slave digital filter must be disabled if the LPI2C functional clock is disabled, and this can affect compliance with some of the timing parameters of the I2C specification, such as the data hold time.
Bus clock	The bus clock is only used for bus accesses to the control and configuration registers. The bus clock frequency must be sufficient to support the data bandwidth requirements of the LPI2C master and slave registers.

Table 53-7. Resets

Chip reset	The logic and registers for the LPI2C master and slave are reset to their default state on a chip reset.
Software reset	• The LPI2C master implements a software reset bit in its Control Register. The MCR[RST] will reset all master logic and registers to their default state, except for the MCR itself. • The LPI2C slave implements a software reset bit in its Control Register. The SCR[RST] will reset all slave logic and registers to their default state, except for the SCR itself.
FIFO reset	• The LPI2C master implements write-only control bits that reset the transmit FIFO (MCR[RTF]) and receive FIFO (MCR[RRF]). After a FIFO is reset, that FIFO is empty. • The LPI2C slave implements write-only control bits that reset the transmit data register (SCR[RTF] and receive data register (SCR[RRF]). After a data register is reset, that data register is empty.

53.4.2 Master Mode

The LPI2C master logic operates independently from the slave logic to perform all master mode transfers on the I2C bus.

53.4.2.1 Transmit and Command FIFO commands

The transmit FIFO stores command data to initiate the various I2C operations. The following operations can be initiated through commands in the transmit FIFO:

- START or Repeated START condition with address byte and expecting ACK or NACK.
- Transmit data (this is the default for zero extended byte writes to the transmit FIFO).
- Receive 1-256 bytes of data (can also be configured to discard receive data and not store in receive FIFO).
- STOP condition (can also be configured to send STOP condition when transmit FIFO is empty).

Multiple transmit and receive commands can be inserted between the START condition and STOP conditon; transmit and receive commands must not be interleaved (to comply with the I2C specification). The receive data command and the receive data and discard commands can be interleaved, to ensure that only the desired received data is stored in the receive FIFO (or compared with the data match logic).

The LPI2C master will automatically transmit a NACK on the last byte of a receive data command unless the next command in the FIFO is also a receive data command. A NACK is also automatically transmitted if the transmit FIFO is empty when a receive data command completes.

The LPI2C master supports 10-bit addressing through a (repeated) START condition, followed by a transmit data byte containing the second address byte, followed by any number of data bytes with the master-transmit data.

A START or Repeated START condition that is expecting a NACK (for example,HS-mode master code) must be followed by a STOP or (repeated) START condition.

53.4.2.2 Master operations

Whenever the LPI2C is enabled, it monitors the I2C bus to detect when the I2C bus is idle (MSR[BBF]). The I2C bus is no longer considered idle if either SCL or SDA are low, and the I2C bus becomes idle if a STOP condition is detected or if a bus idle timeout

is detected (as configured by MCFGR2[BUSIDLE]). After the I2C bus is idle, the transmit FIFO is not empty, and the host request is either asserted or disabled, then the LPI2C master will initiate a transfer on the I2C bus. This involves the following steps:

- Wait the bus idle time equal to (MCCR0[CLKLO] + 1) multiplied by the prescaler (MCFGR1[PRESCALE]).
- Transmit a START condition and address byte using the timing configuration in the Master Clock Configuration Register 0 (MCCR0); if a high speed mode transfer is configured, then the timing configuration from Master Clock Configuration Register 1 (MCCR1) is used instead.
- Perform master-transmit or master-receive transfers, as configured by the transmit FIFO.
- Transmit NACK on the last byte of a master-receive transfer, unless the next command in the transmit FIFO is also a receive data command and the transmit FIFO is not empty.
- Transmit a Repeated START or STOP condition as configured by the transmit FIFO and/or MCFGR1[AUTOSTOP]. A repeated START can change which timing configuration register is used.

When the LPI2C master is disabled (either due to MCR[MEN] being clear or automatically due to mode entry), the LPI2C will continue to empty the transmit FIFO until a STOP condition is transmitted. However, the LPI2C will no longer stall the I2C bus waiting for the transmit or receive FIFO, and after the transmit FIFO is empty, the LPI2C will generate a STOP condition automatically.

The LPI2C master can stall the I2C bus under certain conditions; this will result in SCL pulled low continuously on the first bit of a byte, until the condition is removed:

- LPI2C master is enabled and busy, the transmit FIFO is empty, and MCFGR1[AUTOSTOP] is clear.
- LPI2C master is enabled and receiving data, receive data is not being discarded (due to command or receive data match), and the receive FIFO is full.

53.4.2.3 Receive FIFO and Data Matching

The receive FIFO is used to store receive data during master-receiver transfers. Receive data can also be configured to discard receive data instead of storing in the receive FIFO; this is configured by the command word in the transmit FIFO.

Receive data supports a receive data match function that can match received data against one of two bytes or against a masked data byte. The data match function can also be configured to compare only the first one or two received data words since the last (repeated) START condition. Receive data that is already discarded due to the command

word cannot cause the data match to set, and will delay the match on the first received data word until after the discarded data is received. The receiver match function can also be configured to discard all receive data until a data match is detected, using the MCFGR0[RDMO] control bit. When clearing the MCFGR0[RDMO] control bit following a data match, clear MCFGR0[RDMO] before clearing MSR[DMF], to allow all subsequent data to be received.

53.4.2.4 Timing Parameters

The following timing parameters can be configured by the LPI2C master. Parameters are configured separately for high speed mode (MCCR1) and other modes (MCCR0). This allows the high speed mode master code to be sent using the regular timing parameters, and then switch to the high speed mode timing (following a repeated START) until the next STOP condition.

The LPI2C master timing parameters in LPI2C functional clock cycles are configured as follows. They must be configured to meet the I2C timing specification for the required mode.

Table 53-8. Timing Parameters

I2C Specification Timing Parameter	I2C Specification Timing Symbol	LPI2C Timing Parameter (LPI2C functional clock cycles)
SCL clock period	tSCL	(CLKHI + CLKLO + 2 + SCL_LATENCY) x (2 ^ PRESCALE)
hold time (repeated) START condition	tHD:STA	(SETHOLD +1) x (2 ^ PRESCALE)
LOW period of the SCL clock	tLOW	(CLKLO + 1) x (2 ^ PRESCALE)
HIGH period of the SCL clock	tHIGH	(CLKHI + 1 + SCL_LATENCY) x (2 ^ PRESCALE)
setup time for a repeated START condition or STOP condition	tSU:STA, tSU:STO	(SETHOLD + 1 + SCL_LATENCY) x (2 ^ PRESCALE)
data hold time	tHD:DAT	(DATAVD + 1) x (2 ^ PRESCALE)
data setup time	tSU:DAT	(SDA_LATENCY + 1) x (2 ^ PRESCALE)
bus free time between a STOP and START condition	tBUF	(CLKLO + 1 + SDA_LATENCY) x (2 ^ PRESCALE)
data valid time, data valid acknowledge time	tVD:DAT, tVD:ACK	(DATAVD + 1) x (2 ^ PRESCALE)

The latency parameters are defined in the following table, these parameters assume the risetime is less than one LPI2C functional clock cycle. The risetime depends on a number of factors, including the I/O propagation delay, the I2C bus loading and the external pull-

up resistor sizing. A larger risetime will increase the number of cycles that the signal takes to propagate through the synchronizer (and glitch filter), which increases the latency.

Table 53-9. Synchronization Latency

Timing Parameter	Timing Definition
SCL_LATENCY	ROUNDDOWN ((2 + FILTSCL + SCL_RISETIME) / (2 ^ PRESCALE))
SDA_LATENCY	ROUNDDOWN ((2 + FILTSDA + SDA_RISETIME) / (2 ^ PRESCALE))

The following timing restrictions must be enforced to avoid unexpected START or STOP conditions on the I2C bus, or to avoid unexpected START or STOP conditions detected by the LPI2C master. The timing restrictions can be summarized as **SDA cannot change when SCL is high outside of a transmitted (repeated) START or STOP condition**.

Table 53-10. LPI2C Timing Parameter Restrictions

Timing Parameter	Minimum	Maximum	Comment
CLKLO	0x03	-	Also: CLKLO x (2 ^ PRESCALE) > SCL_LATENCY
CLKHI	0x01	-	Configure CLKHI to meet the duty cycle requirements in the I2C specification
SETHOLD	0x02	-	
DATAVD	0x01	CLKLO - SDA_LATENCY - 1	Configure DATAVD to meet the data hold requirement in the I2C specification
FILTSCL	0x00	[CLKLO x (2 ^ PRESCALE)] - 3	FILTSCL and FILTSDA are the only parameters not multiplied by (2 ^ PRESCALE)
FILTSDA	FILTSCL	[CLKLO x (2 ^ PRESCALE)] - 3	Configuring FILTSDA greater than FILTSCL can delay the SDA input to compensate for board level skew
BUSIDLE	(CLKLO+SETHOLD+2) x 2	-	Must also be greater than (CLKHI+1)

The timing parameters must be configured to meet the requirements of the I2C specification; this will depend on the mode being supported and the LPI2C functional clock frequency. When switching between two modes using the different clock configuration registers (for example, Fast and HS-mode), the PRESCALE factor must remain constant between the modes. Some example timing configurations are provided below.

Table 53-11. LPI2C Example Timing Configurations

I2C Mode	Clock Frequency	Baud Rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Fast	8 MHz	400 kbps	0x0	0x0/0x0	0x04	0x0B	0x05	0x02

Table continues on the next page...

Table 53-11. LPI2C Example Timing Configurations (continued)

I2C Mode	Clock Frequency	Baud Rate	PRESCALE	FILTSCL / FILTSDA	SETHOLD	CLKLO	CLKHI	DATAVD
Fast+	8 MHz	1 Mbps	0x0	0x0/0x0	0x02	0x03	0x01	0x01
Fast	48 MHz	400 kbps	0x0	0x1/0x1	0x1D	0x3E	0x35	0x0F
Fast	48 MHz	400 kbps	0x2	0x1/0x1	0x07	0x11	0x0B	0x03
Fast+	48 MHz	1 Mbps	0x2	0x1/0x1	0x03	0x06	0x04	0x04
HS-mode	48 MHz	3.2 Mbps	0x0	0x0/0x0	0x07	0x08	0x03	0x01
Fast	60 MHz	400 kbps	0x1	0x2/0x2	0x11	0x28	0x1F	0x08
Fast+	60 MHz	1 Mbps	0x1	0x2/0x2	0x07	0x0F	0x0B	0x01
HS-mode	60 MHz	3.33 Mbps	0x1	0x0/0x0	0x04	0x04	0x02	0x01

53.4.2.5 Error Conditions

The LPI2C master will monitor for errors while it is active, the following conditions will generate an error flag and block a new START condition from being sent, until the flag is cleared by software:

- A START or STOP condition is detected and is not generated by the LPI2C master (sets MSR[ALF]).
- Transmitting data on SDA and different values are being received (sets MSR[ALF]).
- NACK is detected when transmitting data, and MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- NACK is detected and is expecting ACK for the address byte, and MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- ACK is detected and is expecting NACK for the address byte, and MCFGR1[IGNACK] is clear (sets MSR[NDF]).
- Transmit FIFO is requesting to transmit or receive data without a START condition (sets MSR[FEF]).
- SCL (or SDA if MCFGR1[TIMECFG] is set) is low for (MCFGR2[TIMELOW] * 256) prescaler cycles without a pin transition (sets MSR[PLTF]).

Software must respond to the MSR[PTLF] flag to terminate the existing command either cleanly (by clearing MCR[MEN]), or abruptly (by setting MCR[SWRST]).

The MCFGR2[BUSIDLE] field can be used to force the I2C bus to be considered idle when SCL and SDA remain high for (BUSIDLE+1) prescaler cycles. The I2C bus is normally considered idle when the LPI2C master is first enabled, but when BUSIDLE is configured greater than zero then SCL and/or SDA must be high for (BUSIDLE+1) prescaler cycles before the I2C bus is first considered idle.

53.4.2.6 Pin Configuration

- **Open-drain support:** The LPI2C master defaults to open-drain configuration of the SDA and SCL pins. Support for true open drain depends on the specific device, and requires the pins where LPI2C pins are muxed to support true open drain.
- **High Speed mode support:** Support for high speed mode also depends on the specific device, and requires the SCL pin to support the current source pull-up required in the I2C specification.
- **Ultra-Fast mode support:** The LPI2C master also supports the output-only push-pull function required for I2C ultra-fast mode using the SDA and SCL pins. Support for ultra-fast mode also requires the IGNACK bit to be set.
- **Push-pull 2-wire support:** A push-pull 2-wire configuration is also available to the LPI2C master that may support a partial high speed mode, if the LPI2C is the only master and all I2C pins on the bus are at the same voltage. This will configure the SCL pin as push-pull for every clock except the 9th clock pulse, to allow high speed mode compatible slaves to perform clock stretching. In this mode, the SDA pin is tristated for master-receive data bits and master-transmit ACK/NACK bits, and is configured as push-pull at other times. To avoid the risk of contention when SDA is push-pull, the pin can be configured for open-drain operation, as part of the device-specific configuration.
- **Push-pull 4-wire support:** The push-pull 4-wire configuration separates the SCL input data and output data into separate pins, and separates the SDA input data and output data into separate pins. The SCL/SDA pins are used for input data; the SCLS/SDAS pins are used for output data, with configurable polarity. This simplifies external connections when connecting the LPI2C to the I2C bus through external level shifters or discrete components. When using this 4-wire configuration, the LPI2C master logic and LPI2C slave logic are not able to connect to separate I2C buses.

53.4.3 Slave Mode

To perform all slave mode transfers on the I2C bus, the LPI2C slave logic operates independently from the LPI2C master logic.

53.4.3.1 Address Matching

The LPI2C slave can be configured:

- to match one of two addresses, using either 7-bit or 10-bit addressing modes for each address
- to match a range of addresses in either 7-bit or 10-bit addressing modes

- to match the General Call Address, and generate appropriate flags
- to match the SMBus Alert Address, and generate appropriate flags
- to detect the high speed mode master code, and to disable the digital filters and output valid delay time until the next STOP condition is detected

After a valid address is matched, the LPI2C slave will automatically perform slave-transmit or slave-receive transfers until:

- a NACK is detected (unless IGNACK is set)
- a bit error is detected (the LPI2C slave is driving SDA, but a different value is sampled)
- a (repeated) START or STOP condition is detected

53.4.3.2 Transmit and Receive Data

- The Transmit and Receive Data registers are double-buffered and only update during a slave-transmit and slave-receive transfer, respectively.
- The slave address *that was received* can be configured to be read from either the Receive Data register (for example, when using DMA to transfer data), or from the Address Status register.
- The Transmit Data register can be configured to only request data after a slave-transmit transfer is detected, or to request new data whenever the Transmit Data register is empty.
- The Transmit Data register should only be written when the Transmit Data flag is set.
- The Receive Data register should only be read when the Received Data flag is set (or the Address Valid flag is set and RXCFG=1).
- The Address Status register should only be read when the Address Valid flag is set.

53.4.3.3 Clock Stretching

The LPI2C slave supports many configurable options for when clock stretching is performed. The following conditions can be configured to perform clock stretching:

- During the 9th clock pulse of the address byte and the Address Valid flag is set.
- During the 9th clock pulse of a slave-transmit transfer and the Transmit Data flag is set.
- During the 9th clock pulse of a slave-receive transfer and the Receive Data flag is set.
- During the 8th clock pulse of an address byte or a slave-receive transfer and the Transmit ACK flag is set. In high speed mode, this is disabled.
- Clock stretching can also be extended for CLKHOLD cycles, to allow additional setup time to sample the SDA pin externally. In high speed mode, this is disabled.

Unless extended by the CLKHOLD configuration, clock stretching will extend for one peripheral bus clock cycle after SDA updates when clock stretching is enabled.

53.4.3.4 Timing Parameters

The LPI2C slave can configure the following timing parameters. These parameters are disabled when SCR[FILTEN] is clear, when SCR[FILTDZ] is set in Doze mode, and when LPI2C slave detects high speed mode. When disabled, the LPI2C slave is clocked directly from the I2C bus, and may not satisfy all timing requirements of the I2C specification (such as SDA minimum hold time in Standard/Fast mode).

- SDA data valid time from SCL negation to SDA update
- SCL hold time when clock stretching is enabled to increase setup time when sampling SDA externally
- SCL glitch filter time
- SDA glitch filter time

The LPI2C slave imposes the following restrictions on the timing parameters.

- FILTSDA must be configured to greater than or equal to FILTSCL (unless compensating for board level skew between SDA and SCL).
- DATAVD must be configured less than the minimum SCL low period.

53.4.3.5 Error Conditions

The LPI2C slave can detect the following error conditions:

- Bit error flag will set when the LPI2C slave is driving SDA, but samples a different value than what is expected.
- FIFO error flag will set due to a transmit data underrun or a receive data overrun. To eliminate the possibility of underrun and overrun occurring, enable clock stretching.
- FIFO error flag will also set due to an address overrun when RXCFG is set, otherwise an address overrun is not flagged. To eliminate the possibility of overrun occurring, enable clock stretching.

The LPI2C slave does not implement a timeout due to SCL and/or SDA being stuck low. If this detection is required, then the LPI2C master logic should be used and so software can reset the LPI2C slave when this condition is detected.

53.4.4 Interrupts and DMA Requests

Depending on the specific device, interrupts and DMA requests can be combined in some ways:

- The LPI2C master and slave interrupts may be combined
- The LPI2C master and slave transmit DMA requests may be combined
- The LPI2C master and slave receive DMA requests may be combined

53.4.4.1 Master mode

The next table lists the master mode sources that can generate LPI2C master interrupts and LPI2C master transmit/receive DMA requests.

Table 53-12. Master Interrupts and DMA Requests

Master Status Register (MSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
TDF	Transmit Data Flag	Data can be written to Transmit FIFO, as configured by the Transmit FIFO Watermark MFCR[TXWATER]	Y	TX	Y
RDF	Receive Data Flag	Data can be read from the Receive FIFO, as configured by the Receive FIFO Watermark MFCR[RXWATER]	Y	RX	Y
EPF	End Packet Flag	Master has transmitted a Repeated START or STOP condition	Y	N	Y
SDF	STOP Detect Flag	Master has transmitted a STOP condition	Y	N	Y
NDF	NACK Detect Flag	• During an address byte, the master expected an ACK but detected a NACK • During an address byte, the master expected a NACK but detected an ACK • During a master-transmitter data byte, the master detected a NACK	Y	N	Y
ALF	Arbitration Lost Flag	• The master lost arbitration due to a START/STOP condition detected at the wrong time • Or the master was transmitting data but received different data than the data that was transmitted	Y	N	Y
FEF	FIFO Error Flag	The master is expecting a START condition in the Command FIFO, but the next entry in the Command FIFO is not a START condition	Y	N	Y
PLTF	Pin Low Timeout Flag	Pin low timeout is enabled and SCL (or SDA if configured) is low for longer than the configured timeout	Y	N	Y

Table continues on the next page...

Table 53-12. Master Interrupts and DMA Requests (continued)

Master Status Register (MSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
DMF	Data Match Flag	The received data matches the configured data match, but the received data is not discarded due to a command FIFO entry	Y	N	Y
MBF	Master Busy Flag	LPI2C master is busy transmitting/receiving data	N	N	N
BBF	Bus Busy Flag	LPI2C master is enabled and activity is detected on I2C bus, but a STOP condition has not been detected and a bus idle timeout (if enabled) has not occurred.	N	N	N

53.4.4.2 Slave mode

The next table lists the slave mode sources that can generate LPI2C slave interrupts and the LPI2C slave transmit/receive DMA requests.

Table 53-13. Slave Interrupts and DMA Requests

Slave Status Register (SSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
TDF	Transmit Data Flag	Data can be written to the Slave Transmit Data Register (STDR)	Y	TX	Y
RDF	Receive Data Flag	Data can be read from the Slave Receive Data Register (SRDR)	Y	RX	Y
AVF	Address Valid Flag	Address can be read from the Slave Address Status Register (SASR)	Y	RX	Y
TAF	Transmit ACK Flag	ACK/NACK can be written to the Slave Transmit ACK Register (STAR)	Y	N	Y
RSF	Repeated Start Flag	Slave has detected an address match followed by a Repeated START condition	Y	N	Y
SDF	STOP Detect Flag	Slave has detected an address match followed by a STOP condition	Y	N	Y
BEF	Bit Error Flag	Slave was transmitting data, but received different data than what was transmitted	Y	N	Y
FEF	FIFO Error Flag	• Transmit data underrun • Receive data overrun • Address status overrun (when Receive Data Configuration SCFGR1[RXCFG] = 1,)	Y	N	Y

Table continues on the next page...

Table 53-13. Slave Interrupts and DMA Requests (continued)

Slave Status Register (SSR)		Description	Can generate		
Flag	Name		Interrupt?	DMA Request?	Low Power Wakeup?
		FEF flag can only set when clock stretching is disabled.			
AM0F	Address Match 0 Flag	Slave detected an address match with SAMR[ADDR0] field	Y	N	N
AM1F	Address Match 1 Flag	Slave detected an address match with SAMR[ADDR1] field or using an address range	Y	N	N
GCF	General Call Flag	Slave detected an address match with the General Call address	Y	N	N
SARF	SMBus Alert Response Flag	Slave detected an address match with the SMBus Alert address	Y	N	N
SBF	Slave Busy Flag	LPI2C slave is busy receiving an address byte or is transmitting/receiving data	N	N	N
BBF	Bus Busy Flag	LPI2C slave is enabled and a START condition is detected on I2C bus, but a STOP condition has not been detected	N	N	N

53.4.5 Peripheral Triggers

The connection of the LPI2C peripheral triggers to other peripherals depend upon the specific device being used.

Table 53-14. LPI2C Triggers

Trigger	Description
Master Output Trigger	The LPI2C master generates an output trigger that can be connected to other peripherals on the device. The master output trigger asserts on both a Repeated START or STOP condition, and the master output trigger remains asserted for one cycle of the LPI2C functional clock divided by the prescaler.
Slave Output Trigger	The LPI2C slave generates an output trigger that can be connected to other peripherals on the device. The slave output trigger asserts on both a Repeated START or STOP condition that occurs after a slave address match, and the slave output trigger remains asserted until the next slave SCL pin negation.
Input Trigger	To control the start of a LPI2C bus transfer, the LPI2C input trigger can be selected instead of the HREQ input. The input trigger is synchronized and to be detected, the input trigger must assert for at least 2 cycles of the LPI2C functional clock divided by the PRESCALE configuration. When the LPI2C is busy, the HREQ input (and therefore the input trigger) is ignored.

Chapter 54
Universal System Bus On-The-Go (USB-OTG)

54.1 Chip-specific USBHSOTG information

Table 54-1. Reference links to related information

Topic	Related module	Reference
Full description	USBHSOTG	USBHSOTG
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

54.1.1 USB High Speed OTG Controller (USBHS)

The USBHS is a USB 2.0-compliant serial interface engine for implementing a USB interface. The registers and data structures are based on the Enhanced Host Controller Interface Specification for Universal Serial Bus (EHCI). The USBHS module can act as a host, a device or an On-The-Go negotiable host/device on the USB bus. It uses 32-bit AXI bus master and 32-bit IP bus

The USBHS controller interfaces to the processor's core. The controller is programmable to support host or device operation under firmware control.

The following table shows the USBHS configuration:

Table 54-2. USBHS configuration

Parameter	Description
Name	USBHS
Supported standard version	USB2.0 Specification
Instances	2
Configurable features	USBOTG1:

Table continues on the next page...

Table 54-2. USBHS configuration (continued)

Parameter	Description
	• Low-speed/Full-speed/High-speed • Host/Device support • UTMI interface (to on-chip PHY) • 8 bi-directional endpoints • 8x256x36 TX FIFO, 1x256x36 RX FIFO USBOTG2: • Low-speed/Full-speed/High-speed • Host/Device support • Interface to on-chip HSIC PHY (host only) and ULPI interface to off-chip PHY • 8 bi-directional endpoints • 8x256x36 TX FIFO, 1x256x36 RX FIFO
Interface speed	N/A
External I/O pins	• USBOTG1: OC, PWR (other USB0 interface pins are connected to the PHY). See the attached IOMUXC spreadsheet for pin details. • USBOTG2 ULPI interface: DAT[7:0], CLK, DIR, STP, NXT, OC, PWR. See the attached IOMUXC spreadsheet for pin details. • USBOTG2 HSIC Interface: STROBE, DATA. See the attached IOMUXC spreadsheet for pin details.

54.1.2 Overcurrent detection and Power Port features

- Overcurrent condition: If an overcurrent takes place, subsequent operation of USB is not guaranteed. The external device must be able to detect the over-current condition and report it to the part. This signaling is flagged for both OTG1 and OTG2 controllers. The controllers then store this indication so that SW can take corrective action. The overcurrent polarity is chosen via the OVER_CUR_POL bit of the OTG1/2 Control register (USBNC_n_CTRL1[OVER_CUR_POL]). By writing into the OVER_CUR_DIS bit of the same register (USBNC_n_CTRL1[OVER_CUR_DIS]), the overcurrent detection can be disabled.
- Power Port feature: The Power Port feature is used to control power of an external device. This feature is used to drive on or off the VBUS indication of an external device. The Port Power (PP) is configured by writing into the PP bit of the Port Status and Control 1 register (USB_nPORTSC1[PP]). The power polarity is chosen via the PWR_POL bit of the OTG1/2 Control register (USBNC_n_CTRL1[PWR_POL]).

NOTE

For the OTG1, which shares both ULPI and HSIC interfaces, these features are only supported when OTG1 is in ULPI mode. HSIC interface specification does not define such features.

54.1.3 Wake-up from VLLS mode by USB PHY wake-up

USB PHY wake-up sources may be used to wake-up from VLLS mode.

- If VDD_DIG1 is powered off once A7 enters the VLLS mode, then only USBNC_n_CTRL1[WKUP_DPDM_EN] is functional.
- If VDD_DIG1 is not powered off, then all 3 wake-up sources are available.

In A7 VLLS mode, the USB0_ID pin/pad will not be functional because the path from I/O pads to USB PHY is powered off. In that case, USBNC_n_CTRL1[WKUP_ID_EN] bit will be used to enable wake-up on a plugged status change instead of wake-up on ID change. Plugged status is clear if both DP, DM pins are logic high and set otherwise.

54.1.3.1 Both CA7 and CM4 are in VLLS mode

To wake-up from MCU from VLLS mode using USB PHY, it is necessary to configure the feature on SIM_GPR1 and SIM_DGO_GPR11 registers as follows.

1. Set SIM_GPR1[USB_PHY_WAKEUP_ISO_DISABLE]= 1 to disable USB Pins to wake-up logic isolation.
2. Wait at least 100 us before enabling wake-up interrupt to enable proper initialization and avoid an early wake-up. This is required due to the slow clock used by wake-up logic.

NOTE

It is expected that USB interface is unplugged or in suspended state. Any USB activity may result in an early wake-up.

3. Configure one source for USB wake-up at USBNC_n_CTRL1 register as follows:
   - USBNC_n_CTRL1[WKUP_DPDM_EN] (Enabled by default)
   - USBNC_n_CTRL1[WKUP_VBUS_EN]
   - USBNC_n_CTRL1[WKUP_ID_EN])
4. Set SIM_DGO_GPR11[USB_PHY_VLLS_WAKEUP_EN] = 1 (SIM_DGO_GPR11 bit 12) to enable wake-up interrupt. (Follow steps described in SIM DGO Register Protocol)
5. Enable LLWU M6IF interrupt to enable M4 wake-up from VLLS by LLWU_ME[WUME6] = 1.

6. Enter A7 VLLS mode.
7. Enter M4 VLLS mode.

When a wake-up activity is detected, a wake-up will be sent to LLWU. LLWU will wake up M4 and generate IRQ20 (LLWU) to M4. M4 should service the interrupt, may read PF[WUF6] and must clear SIM_DGO_GP11[USB_PHY_VLLS_WAKEUP_EN] (Follow steps described in SIM DGO Register Protocol). M4 should wake up A7 using MU.

54.1.3.2 Only A7 in VLLS mode

Follow steps #1 to 4 described in Both CA7 and CM4 are in VLLS mode, then enter A7 VLLS mode. In this case, the wake-up logic will generate an interrupt IRQ69 to M4. M4 should service interrupt, clear SIM_DGO_GP11[USB_PHY_VLLS_WAKEUP_EN] and use MU to wake up A7 as described in Both CA7 and CM4 are in VLLS mode.

54.1.4 Non VLLS Wake-up by USB PHY wakeup

USBPHY wake-up sources may also be used to generate interrupts when CA7 is not in VLLS mode. In this case, SIM_GPR1[USB_PHY_WAKEUP_ISO_DISABLE] and SIM_DGO_GP11[USB_PHY_VLLS_WAKEUP_EN] should not be set but SIM_GPR1[USB_PHY_NON_VLLS_WAKEUP_EN] should be used in place to enable interrupt.

Wake-up logic will generate interrupt IRQ69 to CM4.

54.2 Overview

The USB controller block provides high performance USB functionality that conforms to the *Universal Serial Bus Specification*, Rev. 2.0 (Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips; 2000), and the *On-The-Go and Embedded Host Supplement to the USB Revision* 2.0 Specification (Hewlett-Packard Company, Intel Corporation, LSI Corporation, Microsoft Corporation, Renesas Electronics Corporation, ST-Ericsson; 2012).

The USB controller consists of independent USB controller cores: two On-The-Go (OTG) controller cores. Each controller core supports UTMI, or ULPI and HSIC interfaces according to its feature. See Features for more details. controller cores are single-port cores. For the OTG cores, there is only one port. For the HSIC interface, there is also only one port which is used as a downstream port.

The following figure is a block diagram of USB.

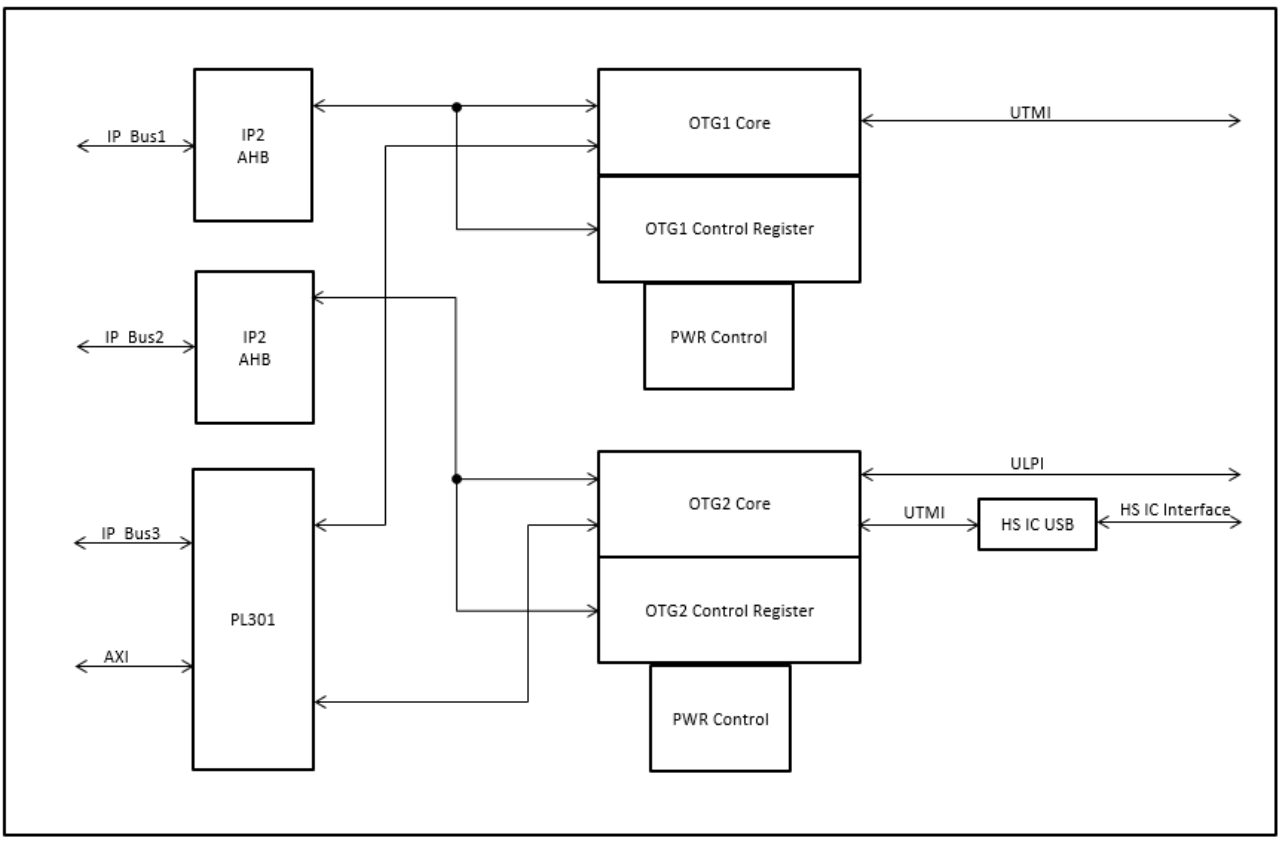


Figure 54-1. USB block diagram

54.2.1 Features

There are USB 2.0 controller cores in this chip:

- Controller Core 0 is also named 'OTG1 Core'; its connected port is named 'OTG1 port'.
- Controller Core 1 is also named 'OTG2 Core'; its connected port is named 'OTG2 port'.

The following list provides features of each of the controller cores.

- USB 2.0 Controller Core 0
  - High-Speed/Full-Speed/Low-Speed OTG core
  - HS/FS/LS UTMI compliant interface
  - High Speed, Full Speed and Low Speed operation in Host mode (with UTMI transceiver)
  - High Speed, and Full Speed operation in Peripheral mode (with UTMI transceiver)

    - Hardware support for OTG signaling, session request protocol, and host negotiation protocol
    - Up to 8 bidirectional endpoints
    - Support charger detection
- USB 2.0 Controller Core 1
    - High-Speed/Full-Speed/Low-Speed OTG core with ULPI interface
    - HS/FS/LS ULPI compliant interface
    - High Speed, Full Speed and Low Speed operation in Host mode (with ULPI transceiver)
    - High Speed, and Full Speed operation in Peripheral mode (with ULPI transceiver)
    - Hardware support for OTG signaling, session request protocol, and host negotiation protocol
    - Up to 8 bidirectional endpoints
    - High-Speed Host-Only core with HSIC interface
    - High Speed Inter-Chip USB compliant interface (HSIC)
- Low-power mode with local and remote wake-up capability
- Embedded DMA controller in each core

54.2.2 Modes of Operation

The USB has two main modes of operation: normal mode and low power mode.

Each USB OTG controller core can operate in High Speed operation (480 Mbps), Full Speed operation (12 Mbps) and Low Speed operation (1.5 Mbps).

This chapter explains the operation modes.

54.2.2.1 Normal Mode

The OTG controller core can operate in Host mode and Device (Peripheral) mode.

Each USB controller core has its corresponding port, which can work in one or more interface modes.

NOTE

Each controller supports only the interface type listed below. Selecting a different interface type in the PORTSC.PTS field results in unpredictable behavior and may cause the system to hang.

- OTG1 port

- This port supports on-chip UTMI transceiver only.

- OTG2 port
  - This port supports on-chip ULPI transceiver.
  - This port supports HSIC interface.

    Interface for on-board HSIC compatible USB peripherals

NOTE

HSIC is an inter-chip interface that is optimized for circuit board layouts.

54.2.2.2 Low-Power Mode

Each USB controller core has a low-power mode (Suspend mode) to save power consumption.

As described in the USB 2.0 specification, the device can go into the Suspend state after it sees a constant Idle state on the upstream facing port. The OTG controller core enters Suspend mode after 3 ms of inactivity on the port when it is in Device Operation mode. Host controllers, including the OTG controller in Host mode, do not suspend automatically but can be placed in Suspend mode by software.

Either the local Arm platform or the remote USB Host/Peripheral can initiate a wake-up sequence to resume USB communication. For details about Suspend/Resume, see USB Power Control.

54.3 External Signals

The table found here describes the external signals of USB.

Table 54-3. USB external signal table

Pin Name	Function		Description
	ALT11	ALT10	
OTG1 Interface			
PTE6	USB0_OC	-	USB0 Over-Current Indication
PTE14		-	
PTE7	USB0_PWR	-	USB0 Port Power Control
PTE15		-	
OTG2 ULPI Interface			

Table continues on the next page...

Table 54-3. USB external signal table (continued)

Pin Name	Function		Description
	ALT11	ALT10	
PTC16	USB1_ULPI_OC		USB1 Over-Current Indication
PTE12			
PTC19	USB1_ULPI_PWR		USB1 Port Power Control (PTC19) shared with OTG0 ID pin
PTE13			
PTF8	USB1_ULPI_CLK	-	
PTF9	USB1_ULPI_NXT	-	
PTF10	USB1_ULPI_STP	-	
PTF11	USB1_ULPI_DIR	-	
PTF12	USB1_ULPI_DATA0	-	ULPI 8-Bit Data Bus
PTF13	USB1_ULPI_DATA1	-	
PTF14	USB1_ULPI_DATA2	-	
PTF15	USB1_ULPI_DATA3	-	
PTF16	USB1_ULPI_DATA4	-	
PTF17	USB1_ULPI_DATA5	-	
PTF18	USB1_ULPI_DATA6	-	
PTF19	USB1_ULPI_DATA7	-	
PTB19	-	USB0_ID	OTG0 ID pin
PTC19	-	USB0_ID	OTG0 ID pin
PTC13	USB0_ID		OTG0 ID pin
OTG2 HSIC Interface			
HSIC_STROBE	-	-	DDR STROBE pad for HSIC interface
HSIC_DATA	-	-	DDR DATA pad for HSIC interface
OTG1 PHY Interface			
USB1_ULPI_DATA7	USB1_ULPI_DATA7	USB0_ID	OTG0 ID pin shared with ULPI Data Bus
USB0_DP	-	-	OTG0 D+ pin
USB0_DM	-	-	OTG0 D- pin
USB0_VBUS_DETECT	-	-	OTG0 VBUS pin

54.4 Functional Description

These sections describe the functionality of the various building blocks of the USB.

54.4.1 USB 2.0 Controller Core 0

The USB 2.0 Controller 0 is an instantiation of an EHCI-compatible core which supports high-, full-, and low-speed operation.

In Host mode, this controller core supports high-, full-, and low-speed operation. In Device mode, it supports high- and full-speed operation.

54.4.1.1 Host Mode

The controller supports direct connection of . a HS/FS/LS device with on-chip UTMI transceiver for USB0 and interface to on-chip HSIC PHY (host only) and ULPI for USB1

Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a USB 2.0 high speed hub has been implemented within the DMA and protocol engine blocks to support connection to full and low speed devices.

54.4.1.2 Peripheral (Device) Mode

- Up to eight bidirectional endpoints
- High/full-speed operation
- Support of HNP, and SRP
- Remote wake-up capability

54.4.2 USB 2.0 Controller Core 1

USB 2.0 Controller Core 1 is an instantiation of EHCI-compatible core which supports High Speed / Full Speed / Low Speed operation with ULPI interface. It also supports High Speed operation with HSIC interface.

This USB core's signals connect directly to I/O pins (HSIC interface).

54.4.3 USB Power Control

The USB controller supports suspend and wake-up functionality.

The power control block allows for placing the transceiver in USB low power mode when USB bus is IDLE, and supports local and remote wake-up to bring the transceiver out of USB low power mode when needed. Additionally, the power control block can wake-up the Arm platform from core sleep mode by generating an interrupt.

54.4.3.1 Entering Low Power Suspend Mode

In Host operation mode, low power suspend mode is entered as follows:

1. Clear the ASE and PSE bits in USB_USBCMD, and wait until the AS and PS bits in USB_USBSTS become "0".
2. Set the "SUSPEND" bit in USB_PORTSC1
3. Set the "PHCD" bit in USB_PORTSC1

For device operation mode, low power suspend mode is entered as follows:

1. After Host drive is IDLE for 3ms, an SLI interrupt is issued (the "DCSUSPEND" or "SLI" bit in USB_USBSTS)
2. Set the "PHCD" bit on USB_PORTSC1

54.4.3.2 Wake-Up Events

The power control block monitors the USB bus when the USB core is in the USB suspend state.

Depending on whether the core is on Host or Device mode, a number of wake-up conditions are monitored. Upon detection of a wake-up condition, an interrupt (asynchronous) will be generated to Arm platform if the related wake-up interrupt enable bit is set.

USB wake-up interrupt also re-activates the Arm platform clocks if they were stopped during the suspend.

54.4.3.2.1 Host Mode Events

The host controller wakes up on the following events:

- Remote Wake-up Request

A peripheral can request the host to reactivate the bus by driving wake-up signaling on the DM/DP lines. The power control block sends a wake-up request to the USB core when a J-K transition on DM/DP line is detected.

- Wake-Up On Overcurrent

If Wake-Up On Overcurrent is enabled (WKOC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core when an overcurrent event is detected.

- Wake-Up On Disconnect

If Wake-Up On Disconnect is enabled (WKDC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core when a disconnection event is detected (J-SE0/K-SE0 transition on DM/DP line).

- Wake-Up On Connect

If a Wake-Up On Connect is enabled (WKCN bit in the USB core register PORTSC1 is set '1'), the power control sub-block sends a wake-up request to the USB core when the connection event is detected (SE0-J/SE0-K transition on DM/DP line).

For a detailed description of register bits WKOC, WKDC, WKCN, please see Port Status & Control (USB_*n*PORTSC1).

54.4.4 Interrupts

54.4.4.1 USB Core Interrupts

Each USB core uses one dedicated vector in the Interrupt Table. The vector numbers associated with each of the cores can be found in the Interrupt section.

With the exception of the wake-up interrupts, all of the interrupt sources are controlled in the USB Cores. Refer to the Interrupt Enable Register (USB_*n*USBINTR) for details.

54.4.4.2 USB Wake-Up Interrupts

Each USB Core has an associated wake-up interrupt. The wake-up interrupts are generated outside of the USB controller cores, but using the same vector as the corresponding USB controller cores interrupt.

These interrupts are generated by the Power Control blocks which run on the 32 KHz standby clock. The wake-up interrupt is designed to work even when the USB and Arm platform clocks are disabled, such that a wake-up condition on the USB bus can re-activate the Arm platform clocks.

Because the wake-up interrupt is generated and cleared on a 32 KHz clock, this interrupt request responds very slowly to clear actions. For this reason, the software must disable the wake-up interrupt to clear the request flag. Disabling the interrupt masks the request instantaneously as this is clocked by the Arm platform clock. The software should wait for at least three 32 KHz clock cycles before re-enabling this interrupt to allow sufficient

time for the request flag to clear. Because this interrupt is only used during low power modes of the USB, it is sufficient to enable the wake-up interrupt just prior to entering the USB suspend mode.

54.5 USB Operation Model

This section describes the detailed application knowledge for OTG1 and OTG2 ports.

54.5.1 Register Interface

Configuration, control and status registers are divided into three categories, identification, capability and operational registers.

NOTE

USB controller registers support only DWORD (32-bit) access.

- Identification registers are used to declare the slave interface presence along with the complete set of the hardware configuration parameters.
- Static, read only capability registers define the software limits, restrictions, and capabilities of the host/device controller.
- Operational registers are dynamic control or status registers that may be read only, read/write, or read/write-to-clear. The following sections define the use of these registers.

EHCI registers are listed alongside device registers to show the complementary nature of host and device control.

The following table describes the Interface register sets.

Table 54-4. Interface Register Sets

Offset	Register Set	Explanation
000h-07Ch	Identification Registers	Identification registers are used to declare the slave interface presence and include a table of the hardware configuration parameters.
100h-124h	Capability Registers	Capability registers specify the limits, restrictions, and capabilities of a host/device controller implementation. These values are used as parameters to the host/device controller driver.
080h-0FCh 140h-1FCh	Operational Registers	Operational registers are used by the system software to control and monitor the operational state of the host/device controller.

54.5.1.1 Configuration, Control and Status Register Set

The following table describes the Device/Host capability registers.

NOTE

Depending on implementation, "*x*" can have the following values: UOG1 , UOG2.

Table 54-5. Device/Host Capability Registers

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
000h	4	USB_*x*_ID	Identification Register	O	O
004h	4	USB_*x*_HWGENERAL	General Hardware Parameters	O	O
008h	4	USB_*x*_HWHOST	Host Hardware Parameters	X	O
00Ch	4	USB_*x*_HWDEVICE	Device Hardware Parameters	O	X
010h	4	USB_*x*_HWTXBUF	TX Buffer Hardware Parameters	O	O
014h	4	USB_*x*_HWRXBUF	RX Buffer Hardware Parameters	O	O
018-07Fh		-	Reserved		
080h	4	USB_*x*_GPTIMER0LD	General Purpose Timer #0 Load Register	O	O
084h	4	USB_*x*_GPTIMER0CTRL	General Purpose Timer #0 Control Register	O	O
088h	4	USB_*x*_GPTIMER1LD	General Purpose Timer #1 Load Register	O	O
08Ch	4	USB_*x*_GPTIMER1CTRL	General Purpose Timer #1 Control Register	O	O
090h	4	USB_*x*_SBUSCFG	System Bus Interface Configuration Register	O	O
094-09Fh		-	Reserved		
100h	1	USB_*x*_CAPLENGTH	Capability Register Length	O	O
101h		-	Reserved		
102h	2	USB_*x*_HCIVERSION	Host Controller Interface Version Number	X	O
104h	4	USB_*x*_HCSPARAMS	Host Controller Structural Parameters	X	O
108h	4	USB_*x*_HCCPARAMS	Host Controller Capability Parameters	X	O
10C-11Fh		-	Reserved		
120h	2	USB_*x*_DCIVERSION	Device Controller Interface Version Number	O	X
122h	2	-	Reserved		
124h	4	USB_*x*_DCCPARAMS	Device Controller Capability Parameters	O	X
128-13Fh		-	Reserved		
140h	4	USB_*x*_USBCMD	USB Command Register	O	O
144h	4	USB_*x*_USBSTS	USB Status Register	O	O
148h	4	USB_*x*_USBINTR	USB Interrupt Enable Register	O	O
14Ch	4	USB_*x*_FRINDEX	USB Frame Index	O	O
150h	4	-	Reserved		
154h	4	USB_*x*_PERIODICLISTBASE	Frame List Base Address	X	O

Table continues on the next page...

Table 54-5. Device/Host Capability Registers (continued)

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
		USB_*x*_DEVICEADDR	USB Device Address	O	X
158h	4	USB_*x*_ASYNCLISTADDR	Next Asynchronous List Address	X	O
	4	USB_*x*_ENDPOINTLISTADDR	Address at Endpoint list in memory	O	X
15Ch	4	-	Reserved		
160h	4	USB_*x*_BURSTSIZE	Programmable Burst Size	O	O
164h	4	USB_*x*_TXFILLTUNING	Host Transmit Pre-Buffer Packet Tuning	X	O
168h	4	-	Reserved		
170h	4	-	Reserved		
178h	4	USB_*x*_ENDPTNAK	Endpoint NAK register	O	X
17Ch	4	USB_*x*_ENDPTNAKEN	Endpoint NAK Enable register	O	X
180h	4	USB_*x*_CONFIGFLAG	Configured Flag Register	X	O
184h	4	USB_*x*_PORTSC1	Port Status/Control Register 1	O	O
188-1A3h		-	Reserved		
1A4h	4	USB_*x*_OTGSC	On-The-Go Status/Control Register (OTG only)	O	O
1A8h	4	USB_*x*_USBMODE	USB Controller Operating Mode	O	O
1ACh	4	USB_*x*_ENDPTSETUPSTAT	Endpoint Setup Status	O	X
1B0h	4	USB_*x*_ENDPTPRIME	Endpoint Initialization	O	X
1B4h	4	USB_*x*_ENDPTFLUSH	Endpoint De-Initialization	O	X
1B8h	4	USB_*x*_ENDPTSTATUS	Endpoint Status	O	X
1BCh	4	USB_*x*_ENDPTCOMPLETE	Endpoint Complete	O	X
1C0 1C4 ... 1DCh	64	USB_*x*_ENDPTCTRL0 USB_*x*_ENDPTCTRL1 .... USB_*x*_ENDPTCTRL7	Endpoint Control Register 0-7	O	X

NOTE

"O" means the register is available in host/device operation mode;

"X" means the register is reserved in host/device operation mode

54.5.1.2 Identification Registers

Identification registers are used to declare the slave interface presence and include a table of the hardware configuration parameters.

54.5.1.3 OTG Operations

54.5.2 Host Data Structures

This section defines the interface data structures used to communicate control, status, and data between HCD (software) and the Enhanced Host Controller (hardware).

The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of a Periodic Schedule, Periodic Frame List, Asynchronous Schedule, Isochronous Transaction Descriptors, Split-transaction Isochronous Transfer Descriptors, Queue Heads, and Queue Element Transfer Descriptors.

The periodic frame list is the root of all periodic (isochronous and interrupt transfer type) transfers for the host controller. The asynchronous list is the root for all the bulk and control transfers. Isochronous data streams are managed using Isochronous Transaction Descriptors. Isochronous split-transaction data streams are managed with Split-transaction Isochronous Transfer Descriptors. All Interrupt, Control, and Bulk data streams are managed via queue heads and Queue Element Transfer Descriptors. These data structures are optimized to reduce the total memory footprint of the schedule and to reduce (on average) the number of memory accesses needed to execute a USB transaction.

Note that software must ensure that no interface data structure reachable by the EHCI host controller spans a 4 K-page boundary.

The data structures defined in this section are (from the host controller's perspective) a mix of read-only and read/writeable fields. The host controller must preserve the read-only fields on all data structure writes.

54.5.2.1 Periodic Frame List

This schedule is for all periodic transfers (isochronous and interrupt). The periodic schedule is referenced from the operational registers space using the USB_PERIODICLISTBASE address register and the USB_FRINDEX register.

The periodic schedule is based on an array of pointers called the Periodic Frame List.

The USB_PERIODICLISTBASE address register is combined with the USB_FRINDEX register to produce a memory pointer into the frame list. The Periodic Frame List implements a sliding window of work over time.

The following figure shows the organization of periodic schedule.

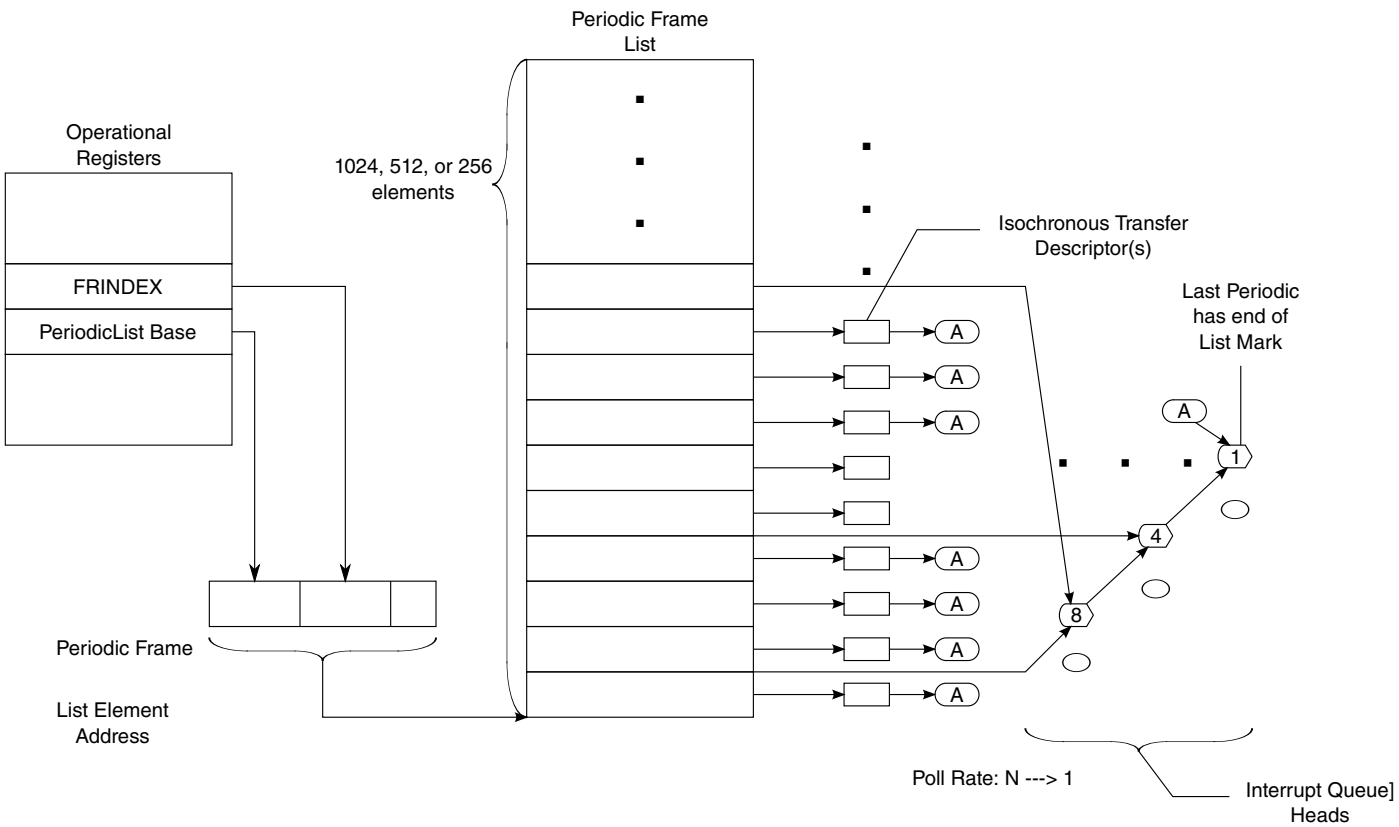


Figure 54-2. Periodic Schedule Organization

Split transaction Interrupt, Bulk and Control are also managed using queue heads and queue element transfer descriptors.

The periodic frame list is a 4 K-page aligned array of Frame List Link pointers. The length of the frame list may be programmable. The programmability of the periodic frame list is exported to system software via the USB_HCCPARAMS register. If non-programmable, the length is 1024 elements. If programmable, the length can be selected by system software as one of 256, 512, or 1024 elements. An implementation must support all three sizes. Programming the size (that is, the number of elements) is accomplished by system software writing the appropriate value into Frame List Size field in the USB_USBCMD register.

Frame List Link pointers direct the host controller to the first work item in the frame's periodic schedule for the current micro-frame. The link pointers are aligned on DWord boundaries within the Frame List.

The table below illustrates the format of the Frame list element pointer.

Table 54-6. Format of Frame List Element Pointer

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Frame List Link Pointer																											0		Typ			03-00H

Frame List Link pointers always reference memory objects that are 32-byte aligned. The referenced object may be an isochronous transfer descriptor for high-speed devices, a split-transaction isochronous transfer descriptor (for full-speed isochronous endpoints), or a queue head (used to support high-, full- and low-speed interrupt). System software should not place non-periodic schedule items into the periodic schedule. The least significant bits in a frame list pointer are used to key the host controller as to the type of object the pointer is referencing.

The least significant bit is the T-Bit (bit 0). When this bit is set to a one, the host controller never uses the value of the frame list pointer as a physical memory pointer. The Typ field is used to indicate the exact type of data structure being referenced by this pointer. The value encodings are.

Table 54-7. Typ Field Value Definitions

Value	Meaning
00b	Isochronous Transfer Descriptor
01b	Queue Head
10b	Split Transaction Isochronous Transfer Descriptor.
11b	Frame Span Traversal Node.

54.5.2.2 Asynchronous List Queue Head Pointer

The Asynchronous Transfer List (based at the USB_ASYNCLISTADDR register) is where all of the control and bulk transfers are managed.

Host controllers use this list only when it reaches the end of the periodic list, the periodic list is disabled, or the periodic list is empty.

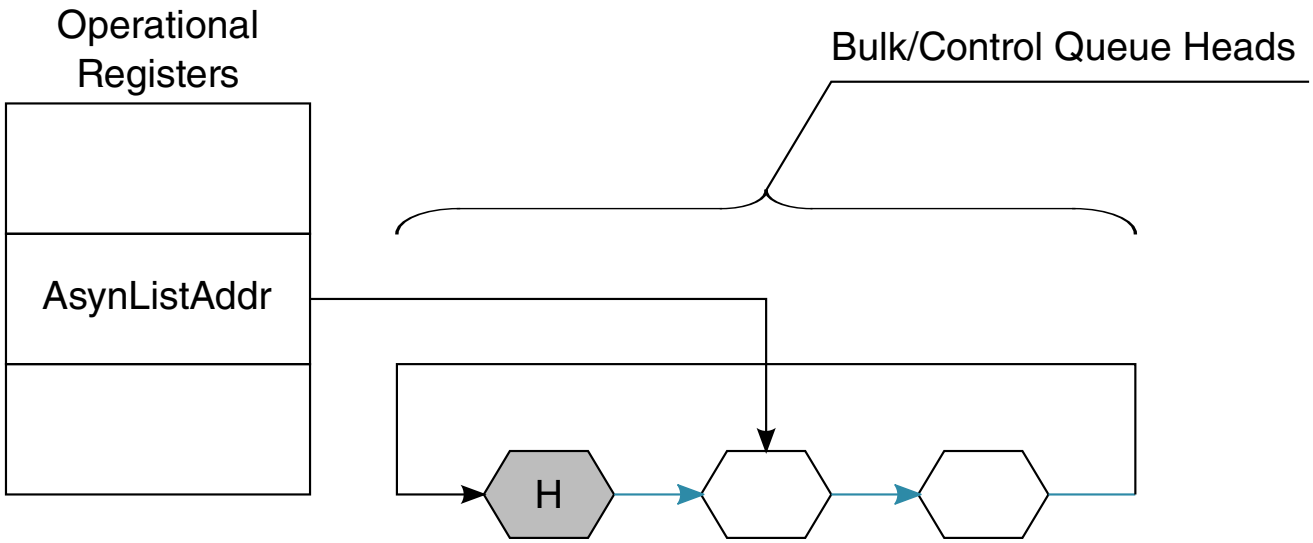


Figure 54-3. Asynchronous Schedule Organization

The Asynchronous list is a simple circular list of queue heads. The USB_ASYNCLISTADDR register is simply a pointer to the next queue head. This implements a pure round-robin service for all queue heads linked into the asynchronous list.

54.5.2.3 Isochronous (High-Speed) Transfer Descriptor (iTD)

The format of an isochronous transfer descriptor is shown in the table below.

This structure is used only for high-speed isochronous endpoints. All other transfer types should use queue structures. Isochronous TDs must be aligned on a 32-byte boundary.

Table 54-8. Isochronous Transaction Descriptor (iTD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Next Link Pointer																											0		Typ		T	03-00 H
Status				Transaction 0 Length												IOC	PG*			Transaction 0 Offset*												07-04 H
Status				Transaction 1 Length												IOC	PG*			Transaction 1 Offset*												0B-08H
Status				Transaction 2 Length												IOC	PG*			Transaction 2 Offset*												0F-0CH
Status				Transaction 3 Length												IOC	PG*			Transaction 3 Offset*												13-10 H

Table continues on the next page...

Table 54-8. Isochronous Transaction Descriptor (iTD) (continued)

Status	Transaction 4 Length	IOC	PG*	Transaction 4 Offset*	17-14H
Status	Transaction 5 Length	IOC	PG*	Transaction 5 Offset*	1B-18H
Status	Transaction 6 Length	IOC	PG*	Transaction 6 Offset*	1F-1CH
Status	Transaction 7 Length	IOC	PG*	Transaction 7 Offset*	23-20H
Buffer Pointer (Page 0)				EndPt / R / Device Address	27-24H
Buffer Pointer (Page 1)				I/O / Maximum Packet Size	2B-28H
Buffer Pointer (Page 2)				- / Mult	2F-2CH
Buffer Pointer (Page 3)				-	33-30H
Buffer Pointer (Page 4)				-	37-34H
Buffer Pointer (Page 5)				-	3B-38H
Buffer Pointer (Page 6)				-	3F-3CH

Host Controller Read/Write

Host Controller Read Only

These fields may be modified by the host controller if the I/O field indicates an OUT.

54.5.2.3.1 Next Link Pointer

The first DWord of an iTD is a pointer to the next schedule data structure.

The following table describes the Next Schedule Element pointer field.

Table 54-9. Next Schedule Element Pointer

Bit	Description
31-5 Link Pointer (LP)	These bits correspond to memory address signals [31:5], respectively. This field points to another Isochronous Transaction Descriptor (iTD/siTD) or Queue Head (QH).
4-3	These bits are reserved and their value has no effect on operation. Software should initialize this field to zero.

Table continues on the next page...

Table 54-9. Next Schedule Element Pointer (continued)

Reserved	
2-1 QH/(s)iTD Select (Typ)	This field indicates to the Host Controller whether the item referenced is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0 Terminate (T)	1= Link Pointer field is not valid. 0= Link Pointer field is valid.

54.5.2.3.2 iTD Transaction Status and Control List

DWords 1 through 8 are eight slots of transaction control and status.

Each transaction description includes:

- Status results field
- Transaction length (bytes to send for OUT transactions and bytes received for IN transactions).
- Buffer offset. The PG and Transaction X Offset fields are used with the buffer pointer list to construct the starting buffer address for the transaction.

The host controller uses the information in each transaction description plus the endpoint information contained in the first three DWords of the Buffer Page Pointer list, to execute a transaction on the USB.

The following table describes iTD Transaction Status and Control fields.

Table 54-10. iTD Transaction Status and Control

Bit	Description	
31-28 Status	This field records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding:	
	Bit	Definition
	31	Active. Set to one by software to enable the execution of an isochronous transaction by the Host Controller. When the transaction associated with this descriptor is completed, the Host Controller sets this bit to zero indicating that a transaction for this element should not be executed when it is next encountered in the schedule.
	30	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, no action is necessary.
	29	Babble Detected. Set to one by the Host Controller during status update when" babble" is detected during the transaction generated by this descriptor.

Table continues on the next page...

Table 54-10. iTD Transaction Status and Control (continued)

Bit	Description	
	28	Transaction Error (XactErr). Set to one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit may only be set for isochronous IN transactions.
27-16 Transaction X Length	For an OUT, this field is the number of data bytes the host controller sends during the transaction. The host controller is not required to update this field to reflect the actual number of bytes transferred during the transfer. For an IN, the initial value of the endpoint to deliver. During the status update, the host controller writes back the field is the number of bytes the host expects the number of bytes successfully received. The value in this register is the actual byte count (0‡zero length data, 1‡one byte, 2‡two bytes, etc.). The maximum value this field may contain is 0xC00 (3072).	
15 Interrupt On Complete (IOC)	If this bit is set to one, it specifies that when this transaction completes, the Host Controller should issue an interrupt at the next interrupt threshold.	
14-12 Page Select (PG)	These bits are set by software to indicate which of the buffer page pointers the offset field in this slot should be concatenated to produce the starting memory address for this transaction. The valid range of values for this field is 0 to 6.	
11-0 Transaction X Offset	This field is a value that is an offset, expressed in bytes, from the beginning of a buffer. This field is concatenated onto the buffer page pointer indicated in the adjacent *PG* field to produce the starting buffer address for this transaction.	

54.5.2.3.3 iTD Buffer Page Pointer List (Plus)

DWords 9-15 of an isochronous transaction descriptor are nominally page pointers (4 K aligned) to the data buffer for this transfer descriptor. This data structure requires the associated data buffer to be contiguous (relative to virtual memory), but allows the physical memory pages to be non-contiguous.

Seven page pointers are provided to support the expression of eight isochronous transfers. The seven pointers allow for 3 (transactions) * 1024 (maximum packet size) * 8 (transaction records) (24576 bytes) to be moved with this data structure, regardless of the alignment offset of the first page.

Because each pointer is a 4 K aligned page pointer, the least significant 12 bits in several of the page pointers are used for other purposes.

The tables below illustrate the field descriptions.

Table 54-11. iTD Buffer Pointer Page 0 (Plus)

Bit	Description
31-12 Buffer Pointer (Page 0)	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-8 Endpoint Number (Endpt)	This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.

Table continues on the next page...

Table 54-11. iTD Buffer Pointer Page 0 (Plus) (continued)

Bit	Description
7 Reserved	Bit reserved for future use and should be initialized by software to zero.
6-0 Device Address	This field selects the specific device serving as the data source or sink.

Table 54-12. iTD Buffer Pointer Page 1 (Plus)

Bit	Description
31-12 Buffer Pointer (Page 1)	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11 Direction (I/O)	0 = OUT; 1 = IN. This field encodes whether the high-speed transaction should use an IN or OUT PID.
10-0 Maximum Packet Size	This directly corresponds to the maximum packet size of the associated endpoint *(wMaxPacketSize)*. This field is used for high-bandwidth endpoints where more than one transaction is issued per transaction description (per micro-frame). This field is used with the *Multi* field to support high-bandwidth pipes. This field is also used for all IN transfers to detect packet babble. Software should not set a value larger than 1024 (400h). Any value larger yields undefined results.

Table 54-13. iTD Buffer Pointer Page 2 (Plus)

Bit	Description
31-12 Buffer Pointer	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-2 Reserved	This bit reserved for future use and should be set to zero.
1-0 Multi	This field is used to indicate to the host controller the number of transactions that should be executed per transaction description (per micro-frame). The valid values are: Value Meaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro- frame. 10b Two transactions to be issued for this endpoint per micro- frame. 11b Three transactions to be issued for this endpoint per micro- frame.

Table 54-14. iTD Buffer Pointer Page 3-6

Bit	Description
31-12 Buffer Pointer	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-0 Reserved	These bits reserved for future use and should be set to zero.

54.5.2.4 Split Transaction Isochronous Transfer Descriptor (siTD)

All Full-speed isochronous transfers through the internal transaction translator are managed using the siTD data structure. This data structure satisfies the operational requirements for managing the split transaction protocol.

The following table shows the Split Transaction Isochronous Transfer Descriptor (siTD).

Table 54-15. Split Transaction Isochronous Transfer Descriptor

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next Link Pointer																											0		Typ		T	03-00
I/O	Port Number							-	Hub Addr							Reserved				EndPt				-	Device Address							07-04[1]
Reserved																µFrame C-mask								µFrame S-mask								0B-08[1]
ioc	P	Reserved				Total Bytes to Transfer										µFrame C-prog-mask								Status								0F-0C[2]
Buffer Pointer (Page 0)																				Current Offset												13-10[2]
Buffer Pointer (Page 1)																				Reserved							TP		T-count			17-14[2]
Back Pointer																											0				T	1B-18

1. 04-0B: Static Endpoint State
2. 0C-13: Transfer results

Host Controller Read/Write

Host Controller Read Only

54.5.2.4.1 Next Link Pointer

DWord0 of a siTD is a pointer to the next schedule data structure.

The following table describes the Next Link Pointer fields.

Table 54-16. Next Link Pointer

Bit	Description
31-5	Next Link Pointer (LP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.

Table continues on the next page...

Table 54-16. Next Link Pointer (continued)

Bit	Description
4-3	Reserved. These bits must be written as zeros.
2-1	QH/(s)iTD Select (Typ). This field indicates to the Host Controller whether the item referenced is an iTD/siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0	Terminate (T). 1 = Link Pointer field is not valid. 0 = Link Pointer is valid.

54.5.2.4.2 siTD Endpoint Capabilities/Characteristics

DWords 1 and 2 specify static information about the full-speed endpoint, the addressing of the parent Companion Controller, and micro-frame scheduling control.

The tables below describe the Endpoint and transaction translator characteristics and micro-frame schedule control fields.

Table 54-17. Endpoint and Transaction Translator Characteristics

Bit	Description
31	Direction (I/O).0 = OUT; 1 = IN. This field encodes whether the full-speed transaction should be an IN or OUT.
30-24	Port Number. This field is the port number of the recipient Transaction Translator.
23	Reserved. Bit reserved and should be set to zero.
22-16	Hub Address. This field holds the device address of the Companion Controllers' hub.
15-12	Reserved. Field reserved and should be set to zero.
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7	Reserved. Bit is reserved for future use. It should be set to zero.
6-0	Device Address. This field selects the specific device serving as the data source or sink.

Table 54-18. Micro-frame Schedule Control

Bit	Description
31-16	Reserved. This field reserved for future use. It should be set to zero.
15-8	Split Completion Mask (mFrame C-Mask). This field (along with the *Active* and *SplitX- state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute complete-split transactions. When the criteria for using this field is met, an all zeros value has undefined behavior. The host

Table continues on the next page...

Table 54-18. Micro-frame Schedule Control (continued)

Bit	Description
	controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the m*Frame C-Mask* field is a one, then this siTD is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Split Start Mask (mFrame S-mask). This field (along with the *Active* and *SplitX-state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute start-split transactions. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the m*Frame S-mask* field is a one, then this siTD is a candidate for transaction execution. An all zeros value in this field, in combination with existing periodic frame list has undefined results.

54.5.2.4.3 siTD Transfer State

DWords 3-6 are used to manage the state of the transfer.

The following table describes siTD transfer state fields.

Table 54-19. siTD Transfer Status and Control

Bit	Description
31	Interrupt On Complete (ioc). 0 = Do not interrupt when transaction is complete. 1 = Do interrupt when transaction is complete. When the host controller determines that the split transaction has completed it asserts a hardware interrupt at the next interrupt threshold.
30	Page Select (P). Used to indicate which data page pointer should be concatenated with the *CurrentOffset* field to construct a data buffer pointer (0 selects *Page 0* pointer and 1 selects *Page 1*). The host controller is not required to write this field back when the siTD is retired (*Active* bit transitioned from a one to a zero).
29-26	Reserved. This field reserved for future use and should be set to zero.
25-16	Total Bytes To Transfer. This field is initialized by software to the total number of bytes expected in this transfer. Maximum value is 1023 (3FFh)
15-8	µFrame Complete-split Progress Mask (C-prog-Mask). This field is used by the host controller to record which split-completes has been executed.
7-0: Status—This field records the status of the transaction executed by the host controller for this slot. It is a bit vector with the encoding shown in the following rows.	
7	Active. Set to one by software to enable the execution of an isochronous split transaction by the Host Controller.
6	ERR. Set to a one by the Host Controller when an ERR response is received from the Companion Controller.
5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). In the case of an under run, the Host Controller transmits an incorrect CRC (thus invalidating the data at the endpoint). If an overrun condition occurs, no action is necessary.
4	Babble Detected. Set to a one by the Host Controller during status update when" babble" is detected during the transaction generated by this descriptor.
3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit is set only for IN transactions.
2	Missed Micro-Frame. The host controller detected that a host-induced hold- off caused the host controller to miss a required complete-split transaction.

Table continues on the next page...

Table 54-19. siTD Transfer Status and Control (continued)

Bit	Description
1	Split Transaction State (SplitXstate). The bit encodings are: Value Meaning 00b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint when a match is encountered in the S-mask. 01b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint when a match is encountered in the C-mask.
0	Reserved. Bit reserved for future use and should be set to zero.

54.5.2.4.4 siTD Buffer Pointer List (plus)

DWords 4 and 5 are the data buffer page pointers for the transfer. This structure supports one physical page cross. The most significant 20 bits of each DWord in this section are the 4 K (page) aligned buffer pointers.

The least significant 12 bits of each DWord are used as additional transfer state. The following table describes the siTD buffer pointer fields.

Table 54-20. Buffer Page Pointer List (plus)

Bit	Description
31-12	Buffer Pointer List. Bits [31:12] of DWords 4 and 5 are 4 K page aligned physical memory addresses. These bits correspond to physical address bits [31:12] respectively. The lower 12 bits in each pointer are defined and used as specified below. The field *P* (see siTD Transfer State) specifies the *current* active pointer.
Bits 11-0 (Page 0)	Current Offset—The 12 least significant bits of the Page 0 pointer are the current byte offset for the current page pointer (as selected with the page indicator bit (*P* field). The host controller is not required to write this field back when the siTD is retired (*Active* bit transitioned from a one to a zero).
Bits 11-0 (Page 1)—The least significant bits of the Page 1 pointer are split into three subfields as shown in the following rows.	
11-5 (Page 1)	Reserved
4-3 (Page 1)	Transaction position (TP). This field is used with T-count to determine whether to send *all*, *first*, *middle*, or *last* with each outbound transaction payload. System software must initialize this field with the appropriate starting value. The host controller must correctly manage this state during the lifetime of the transfer. The bit encodings are: Value Meaning 00b All. The entire full-speed transaction data payload is in this transaction (that is, less than or equal to 188 bytes). 01b Begin. This is the first data payload for a full- speed that is greater than 188 bytes. 10B Mid. This is the *middle* payload for a full-speed OUT transaction that is larger than 188 bytes.

Table continues on the next page...

Table 54-20. Buffer Page Pointer List (plus) (continued)

Bit	Description
	11b End. This is the *last* payload for a full-speed OUT transaction that was larger than 188 bytes.
2-0 (Page 1)	Transaction count (T-Count). Software initializes this field with the number of OUT start-splits this transfer requires. Any value larger than 6 is undefined.

54.5.2.4.5 siTD Back Link Pointer

DWord 6 of a siTD is simply another schedule link pointer. This pointer is always zero, or references a siTD, and it cannot reference any other schedule data structure.

The following table describes the siTD back link pointer fields.

Table 54-21. siTD Back Link Pointer

Bit	Description
31-5	siTD Back Pointer. This field is a physical memory pointer to a siTD.
4-1	Reserved. This field is reserved for future use. It should be set to zero.
0	Terminate (T). 1 = siTD Back Pointer field is not valid. 0 = siTD Back Pointer field is valid.

54.5.2.5 Queue element transfer descriptor (qTD)

This data structure is only used with a queue head. It describes one or more USB transactions to transfer up to 20480 (5*4096) bytes.

The structure contains two structure pointers used for queue advancement, a DWord of transfer state, and a five-element array of data buffer pointers.

It is 32 bytes and must be physically contiguous.

The buffer associated with this transfer must be virtually contiguous. The buffer may start on any byte boundary; however, for optimal utilization of on-chip busses it is recommended to align the buffers on a 32-byte boundary. A separate buffer pointer list element must be used for each physical page in the buffer, regardless of whether the buffer is physically contiguous.

Host controller updates (host controller writes) to stand-alone qTDs only occur during transfer retirement. References in the following bit field definitions of updates to the qTD are to the qTD portion of a queue head.

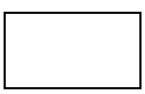
The following table shows the queue element transfer descriptor data structure.

Table 54-22. Queue element transfer descriptor data structure

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next qTD Pointer																											0				T	03-00
Alternate Next qTD Pointer																											0				T	07-04
dt	Total Bytes to Transfer															ioc	C_Page			Cerr		PID Code		Status								0B-08[1]
Buffer Pointer (page 0)																				Current Offset												0F-0C[1]
Buffer Pointer (page 1)																				Reserved												13-10
Buffer Pointer (page 2)																				Reserved												17-14
Buffer Pointer (page 3)																				Reserved												1B-18
Buffer Pointer (page 4)																				Reserved												1F-1C

1. 08-0F: Transfer Results

Host Controller Read/Write



Host Controller Read Only

Queue Element Transfer Descriptors must be aligned on 32-byte boundaries.

54.5.2.5.1 Next qTD Pointer

The first DWord of an element transfer descriptor is a pointer to another transfer element descriptor.

The following table describes Next qTD pointer fields.

Table 54-23. qTD Next Element Transfer Pointer (DWord 0)

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed. The field corresponds to memory address signals[31:5], respectively.
4-1	Reserved

Table continues on the next page...

Table 54-23. qTD Next Element Transfer Pointer (DWord 0) (continued)

0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

54.5.2.5.2 Alternate Next qTD Pointer

The second DWord of a queue element transfer descriptor is used to support hardware-only advance of the data stream to the next transfer descriptor on short packet. To be more explicit the host controller always uses this pointer when the current qTD is retired due to short packet.

The following table describes the TD Alternate Next Element Transfer Pointer field descriptions.

Table 54-24. TD Alternate Next Element Transfer Pointer (DWord 1)

Bit	Description
31-5	Alternate Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed in the event that the current qTD execution encounters a short packet (for an IN transaction). The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved
0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

54.5.2.5.3 qTD Token

The third DWord of a queue element transfer descriptor contains most of the information the host controller requires to execute a USB transaction (the remaining endpoint-addressing information is specified in the queue head).

NOTE

The field descriptions forward reference fields defined in the queue head. Where necessary, these forward references are preceded with a QH notation.

The following table describes the TD Token fields.

Table 54-25. TD Token (DWord 2)

Bit	Description
31 Data Toggle	This is the data toggle sequence bit. The use of this bit depends on the setting of the *Data Toggle Control* bit in the queue head.

Table continues on the next page...

Table 54-25. TD Token (DWord 2) (continued)

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>30-16
Total Bytes to Transfer</td><td>This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction, only on the successful completion of the transaction. The maximum value software may store in this field is 5 * 4K (5000H). This is the maximum number of bytes 5 page pointers can access. If the value of this field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the host controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for OUT transfers that Total Bytes To Transfer be an even multiple of QHD.Maximum Packet Length. If software builds such a transfer descriptor for an OUT transfer, the last transaction is always less than QHD.Maximum Packet Length.

Although it is possible to create a transfer up to 20K this assumes the 1st offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16 K(4000H).</td></tr>
<tr><td>15
Interrupt On Complete (IOC)</td><td>If this bit is set to a one, it specifies that when this qTD is completed, the Host Controller should issue an interrupt at the next interrupt threshold.</td></tr>
<tr><td>14-12
Current Page (C_Page)</td><td>This field is used as an index into the qTD buffer pointer list. Valid values are in the range 0H to 4H. The host controller is not required to write this field back when the qTD is retired.</td></tr>
<tr><td>11-10
Error Counter (CERR)</td><td>This field is a 2-bit down counter that keeps track of the number of consecutive Errors detected while executing this qTD. If this field is programmed with a non-zero value during set-up, the Host Controller decrements the count and writes it back to the qTD if the transaction fails. If the counter counts from one to zero, the Host Controller marks the qTD inactive, sets the Halted bit to a one, and error status bit for the error that caused CERR to decrement to zero. An interrupt is generated if the USB Error Interrupt Enable bit in the USBINTR register is set to a one. If HCD programs this field to zero during set-up, the Host Controller does not count errors for this qTD and there is no limit on the retries of this qTD. Note that write-backs of intermediate execution state are to the queue head overlay area, not the qTD.

Transaction Error - Decrement

Data Buffer Error - No Decrement[3]

Stalled - No Decrement[1]

Babble Detected - No Decrement[1]

No Error - No Decrement[2]
<table>
<tr><th>Error</th><th>Decrement Counter</th></tr>
<tr><td>1</td><td>Detection of Babble or Stall automatically halts the queue head. Thus, count is not decremented</td></tr>
<tr><td>2</td><td>If the EPS field indicates a HS device or the queue head is in the Asynchronous Schedule (and PIDCode indicates an IN or OUT) and a bus transaction completes and the host controller does not detect a transaction error, then the host controller should reset CERR to extend the total number of errors for this transaction. For example, CERR should be reset with maximum value (3) on each successful completion of a transaction. The host controller must never reset this field if the value at the start of the transaction is 00b.

See Split Transaction Interrupt for CERR adjustment rules when the EPS field indicates a FS or LS device and the queue head is in the Periodic Schedule. See Asynchronous - Do Complete Split for CERR adjustment rules when the EPS field indicates a FS or LS device, the queue head is in the Asynchronous schedule and the PIDCode indicates a SETUP.</td></tr>
</table></td></tr>
</table>

Table continues on the next page...

Table 54-25. TD Token (DWord 2) (continued)

Bit	Description	
	3	Data buffer errors are host problems. They don't count against the device's retries.
	NOTE: Software must not program CERR to a value of zero when the EPS field is programmed with a value indicating a Full- or Low-speed device. This combination could result in undefined behavior.	
9-8 PID Code	This field is an encoding of the token, which should be used for transactions associated with this transfer descriptor. Encodings are:	
	00b	OUT Token generates token (E1H)
	01b	IN Token generates token (69H)
	10b	SETUP Token generates token (2DH) (undefined if endpoint is an interrupt, the queue head is non-zero) transfer type, for example, µ*Frame S-mask* field in.
	11b	Reserved
7-0 Status	This field is used by the Host Controller to communicate individual command execution states back to HCD. This field contains the status of the last transaction performed on this qTD. The bit encodings are:	
	Bit	Status Field Description
	7	Active. Set to one by software to enable the execution of transactions by the Host Controller.
	6	Halted. Set to one by the Host Controller during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this qTD. This can be caused by babble, the error counter counting down to zero, or reception of the STALL handshake from the device during a transaction. Any time that a transaction results in the Halted bit being set to a one, the Active bit is also set to zero.
	5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, the Host Controller forces a timeout condition on the USB, invalidating the transaction at the source. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	4	Babble Detected. Set to a one by the Host Controller during status update when" babble" is detected during the transaction. In addition to setting this bit, the Host Controller also sets the *Halted* bit to a one. Because "babble" is considered a fatal error for the transfer, setting the Halted bit to a one insures that no more transactions occur because of this descriptor.
	3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	2	Missed Micro-Frame. This bit is ignored unless the *QH.EPS* field indicates a full- or low-speed endpoint and the queue head is in the periodic list. This bit is set when the host controller detected that a host-induced hold-off caused the host controller to miss a required complete-split transaction. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	1	Split Transaction State (SplitXstate). This bit is ignored by the host controller unless the *QH.EPS* field indicates a full- or low-speed endpoint. When a Full- or Low-speed device, the host controller uses this bit to track the state of the split-

Table continues on the next page...

Table 54-25. TD Token (DWord 2) (continued)

Bit	Description	
		transaction. The functional requirements of the host controller for managing this state bit and the split transaction protocol depends on whether the endpoint is in the periodic or asynchronous schedule. The bit encodings are: Value Meaning 0b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint. 1b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint.
	0	Ping State (P)/ERR. If the *QH.EPS* field indicates a High-speed device and the *PID_Code* indicates an OUT endpoint, then this is the state bit for the Ping protocol. The bit encodings are: Value Meaning 0b Do OUT. This value directs the host controller to issue an OUT PID to the endpoint. 1b Do Ping. This value directs the host controller to issue a PING PID to the endpoint. If the *QH.EPS* field does not indicate a High-speed device, then this field is used as an error indicator bit. It is set to a one by the host controller whenever a periodic split-transaction receives an ERR handshake.

54.5.2.5.4 qTD Buffer Page Pointer List

The last five DWords of a queue element transfer descriptor is an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

System software initializes Current Offset field to the starting offset into the current page, where current page is selected through the value in the C_Page field.

The following table describes the qTD Buffer Pointer(s) (DWords 3-7) fields.

Table 54-26. qTD Buffer Pointer(s) (DWords 3-7)

Bit	Description
31-12	Buffer Pointer List. Each element in the list is a 4 K page aligned physical memory address. The lower 12 bits in each pointer are reserved (except for the first one), as each memory pointer must reference the start of a 4 K page. The field C_Page specifies the current active pointer. When the transfer element descriptor is fetched, the starting buffer address is selected using C_Page (similar to an array index to select an array element). If a transaction spans a 4K buffer boundary, the host controller must detect the page-span boundary in the data stream, increment C_Page and advance to the next buffer pointer in the list, and conclude the transaction through the new buffer pointer.
11-0	Current Offset (Reserved). This field is reserved in all pointers except the first one (for example Page 0). The host controller should ignore all reserved bits. For the page 0 current offset interpretation, this field is the byte offset into the current page (as selected by C_Page). The host controller is not required to write this field back when the qTD is retired. Software should ensure the Reserved fields are initialized to zero.

54.5.2.6 Queue Head

The table located in this section shows the Queue Head structure layout.

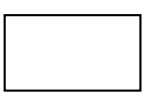
The following table shows the queue head structure layout.

Table 54-27. Queue Head Structure Layout

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Queue Head Horizontal Link Pointer																											0		Typ		T	03-00
RL				C	Maximum Packet Length											H	dtc	EP		EndPt				I	Device Address							07-04[1]
Mult		Port Number[2]							Hub Addr[2]							μFrame C-mask[2]								μFrame S-mask								0B-08[1]
Current qTD Pointer																											0					0F-0C
Next qTD Pointer																											0				T	13-10[3]
Alternate Next qTD pointer																											NakCnt				T	17-14[4]
dt	Total Bytes to Transfer															ioc	C_Page			Cerr		PID Code		Status								1B-18
Buffer Pointer (Page 0)																				Current Offset												1F-1C
Buffer Pointer (Page 1)																				Reserved				C-prog-mask[2]								23-20
Buffer Pointer (Page 2)																				S-bytes[2]										FrameTag[2]		27-24[4]
Buffer Pointer (Page 3)																				Reserved												2B-28
Buffer Pointer (Page 4)																				Reserved												2F-2C[3]

1. 04-0B: Static endpoint state.
2. These fields are used exclusively to support split transactions to USB 2.0 hubs
3. 10-2F: Transfer overlay.
4. 14-27: Transfer results.

Host Controller Read/Write



Host Controller Read Only

54.5.2.6.1 Queue Head Horizontal Link Pointer

The first DWord of a Queue Head contains a link pointer to the next data object to be processed after any required processing in this queue has been completed, as well as the control bits defined below.

This pointer may reference a queue head or one of the isochronous transfer descriptors. It must not reference a queue element transfer descriptor.

The following table describes the Queue head DWord 0 fields.

Table 54-28. Queue Head DWord 0

Bit	Description
31-5	Queue Head Horizontal Link Pointer (QHLP). This field contains the address of the next data object to be processed in the horizontal list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD Select (Typ). This field indicates to the hardware whether the item referenced by the link pointer is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (frame span traversal node)
0	Terminate (T). 1=Last QH (pointer is invalid). 0=Pointer is valid. If the queue head is in the context of the periodic list, a one bit in this field indicates to the host controller that this is the end of the periodic list. This bit is ignored by the host controller when the queue head is in the Asynchronous schedule. Software must ensure that queue heads reachable by the host controller always have valid horizontal link pointers.

54.5.2.6.2 Queue Head Endpoint Capabilities/Characteristics

The second and third DWords of a Queue Head specifies static information about the endpoint. This information does not change over the lifetime of the endpoint.

There are three types of information in this region:

- Endpoint Characteristics. These are the USB endpoint characteristics including addressing, maximum packet size, and endpoint speed.
- Endpoint Capabilities. These are adjustable parameters of the endpoint. They effect how the endpoint data stream is managed by the host controller.
- Split Transaction Characteristics. This data structure is used to manage full- and low-speed data streams for bulk, control, and interrupt via split transactions to USB2.0 Hub Transaction Translator. There are additional fields used for addressing the hub and scheduling the protocol transactions (for periodic).

The host controller must not modify the bits in this region.

The following table describes the Endpoint characteristics: Queue head DWord 1 fields.

Table 54-29. Endpoint Characteristics: Queue Head DWord 1

Bit	Description	
31-28	Nak Count Reload (RL). This field contains a value, which is used by the host controller to reload the Nak Counter field.	
27	Control Endpoint Flag (C). If the *QH.EPS* field indicates the endpoint is not a high-speed device, and the endpoint is a control endpoint, then software must set this bit to a one. Otherwise, it should always set this bit to zero.	
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint (*wMaxPacketSize*). The maximum value this field may contain is 0x400 (1024).	
15	Head of Reclamation List Flag (H). This bit is set by System Software to mark a queue head as being the head of the reclamation list.	
14	Data Toggle Control (DTC). This bit specifies where the host controller should get the initial data toggle on an overlay transition. 0b Ignore DT bit from incoming qTD. Host controller preserves DT bit in the queue head. 1b Initial data toggle comes from incoming qTD DT bit. Host controller replaces DT bit in the queue head from the DT bit in the qTD.	
13-12	Endpoint Speed (EPS). This is the speed of the associated endpoint. Bit combinations are:	
	Value	Meaning
	00b	Full-Speed (12 Mbits/sec)
	01b	Low-Speed (1.5 Mbits/sec)
	10b	High-Speed (480 Mbits/sec)
	11b	Reserved
	This field must not be modified by the host controller.	
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.	
7	Inactivate on Next Transaction (I). This bit is used by system software to request that the host controller set the Active bit to zero. See Rebalancing the periodic schedule , for full operational details. This field is only valid when the queue head is in the Periodic Schedule and the *EPS* field indicates a Full or Low-speed endpoint. Setting this bit to one when the queue head is in the Asynchronous Schedule or the *EPS* field indicates a high-speed device yields undefined results.	
6-0	Device Address. This field selects the specific device serving as the data source or sink.	

The table below describes the Endpoint capabilities: Queue head DWord 2 field descriptions.

Table 54-30. Endpoint Capabilities: Queue Head DWord 2

Bit	Description
31-30	High-Bandwidth Pipe Multiplier (Mult). This field is a multiplier used to key the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution. The host controller makes the simplifying assumption that software properly initializes this field (regardless of location of queue head in the schedules or other run time parameters). The valid values are: Value Meaning

Table continues on the next page...

Table 54-30. Endpoint Capabilities: Queue Head DWord 2 (continued)

	00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro-frame. 10b Two transactions to be issued for this endpoint per micro-frame. 11b Three transactions to be issued for this endpoint per micro-frame.
29-23	Port Number. This field is ignored by the host controller unless the *EPS* field indicates a full- or low-speed device. The value is the port number identifier on the USB 2.0 Hub (for hub at device address *Hub Addr* below), below which the full- or low-speed device associated with this endpoint is attached. This information is used in the split-transaction protocol.
22-16	Hub Addr. This field is ignored by the host controller unless the *EPS* field indicates a full-or low-speed device. The value is the USB device address of the USB 2.0 Hub below which the full- or low-speed device associated with this endpoint is attached. This field is used in the split-transaction protocol.
15-8	Split Completion Mask (µFrame C-Mask). This field is ignored by the host controller unless the *EPS* field indicates this device is a low- or full-speed device and this queue head is in the periodic list. This field (along with the *Active* and *SplitX-state* fields) is used to determine during which micro-frames the host controller should execute a complete-split transaction. When the criteria for using this field are met, a zero value in this field has undefined behavior. This field is used by the host controller to match against the three low-order bits of the FRINDEX register. If the FRINDEX register bits decode to a position where the µ*Frame C- Mask* field is a one, then this queue head is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Interrupt Schedule Mask (µFrame S-mask). This field is used for all endpoint speeds. Software should set this field to a zero when the queue head is on the asynchronous schedule. A non-zero value in this field indicates an interrupt endpoint. The host controller uses the value of the three low-order bits of the FRINDEX register as an index into a bit position in this bit vector. If the µ*Frame S-mask* field has a one at the indexed bit position then this queue head is a candidate for transaction execution. If the *EPS* field indicates the endpoint is a high-speed endpoint, then the transaction executed is determined by the *PID_Code* field contained in the execution area. This field is also used to support split transaction types: Interrupt (IN/OUT). This condition is true when this field is non-zero and the *EPS* field indicates this is either a full- or low-speed device. A zero value in this field, in combination with existing in the periodic frame list has undefined results.

54.5.2.6.3 Transfer Overlay-Queue Head

The nine DWords in this area represent a transaction working space for the host controller. The general operational model is that the host controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it follows the Queue Head Horizontal Link Pointer to the next queue head. The host controller will never follow the Next Transfer Queue Element or Alternate Queue Element pointers unless it is actively attempting to advance the queue. For the duration of the transfer, the host controller keeps the incremental status of the transfer in the overlay area. When the transfer is complete, the results are written back to the original queue element.

The DWord3 of a Queue Head contains a pointer to the source qTD currently associated with the overlay. The host controller uses this pointer to write back the overlay area into the source qTD after the transfer is complete.

The following table describes the current qTD link pointer field descriptions.

Table 54-31. Current qTD Link Pointer

Bit	Description
31-5	Current Element Transaction Descriptor Link Pointer. This field contains the address Of the current transaction being processed in this queue and corresponds to memory address signals [31:5], respectively.
4-0	Reserved (R). These bits are ignored by the host controller when using the value as an address to write data. The actual value may vary depending on the usage.

The DWords 4-11 of a queue head are the transaction overlay area. This area has the same base structure as a Queue Element Transfer Descriptor. The queue head utilizes the reserved fields of the page pointers to implement tracking the state of split transactions.

This area is characterized as an overlay because when the queue is advanced to the next queue element, the source queue element is merged onto this area. This area serves as execution cache for the transfer.

The table below describes the Host-controller rules for bits in overlay.

Table 54-32. Host-Controller Rules for Bits in Overlay (DWords 5, 6, 8 and 9)

DWord	Bit	Description
5	4-1	Nak Counter (NakCnt)µRW. This field is a counter the host controller decrements whenever a transaction for the endpoint associated with this queue head results in a Nak or Nyet response. This counter is reloaded from *RL* before a transaction is executed during the first pass of the reclamation list (relative to an Asynchronous List Restart condition). It is also loaded from RL during an overlay.
6	31	Data Toggle. The *Data Toggle Control* controls whether the host controller preserves this bit when an overlay operation is performed.
6	15	Interrupt On Complete (IOC). The IOC control bit is always inherited from the source qTD when the overlay operation is performed.
6	11-10	Error Counter (C_ERR). This two-bit field is copied from the qTD during the overlay and written back during queue advancement.
6	0	Ping State (P)/ERR. If the *EPS* field indicates a high-speed endpoint, then this field should be preserved during the overlay operation.
8	7-0	Split-transaction Complete-split Progress (C-prog-mask). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	4-0	Split-transaction Frame Tag (Frame Tag). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	11-5	S-bytes. Software must ensure that the *S-bytes* field in a *qTD* is zero before activating the *qTD*. This field is used to keep track of the number of bytes sent or received during an IN or OUT split transaction.

54.5.2.7 Periodic Frame Span Traversal Node (FSTN)

This data structure is to be used only for managing Full- and Low-speed transactions that span a Host-frame boundary.

See Host Controller Operational Model for FSTNs for full operational details. Software must not use an FSTN in the Asynchronous Schedule. An FSTN in the Asynchronous schedule results in undefined behavior. Software must not use the FSTN feature with a host controller whose USB_HCIVERSION register indicates a revision implementation below 0096h. FSTNs are not defined for implementations before 0.96 and their use yields undefined results.

Table 54-33. Frame Span Traversal Node Structure Layout

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Normal Path Link Pointer																											0		Typ		T	03-00
Back Path Link Pointer																											0		Typ[1]		T	07-04

1. Must be set to indicate a queue head

Host Controller Read/Write

Host Controller Read Only

54.5.2.7.1 FSTN Normal Path Pointer

The first DWord of an FSTN contains a link pointer to the next schedule object. This object can be of any valid periodic schedule data type.

The following table describes the FSTN normal path pointer fields.

Table 54-34. FSTN Normal Path Pointer Field Descriptions

Bit	Description
31-5	Normal Path Link Pointer (NPLP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD/FSTN Select (Typ). This field indicates to the Host Controller whether the item referenced is a iTD/siTD, a QH or an FSTN. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (Frame Span Traversal Node)
0	Terminate (T). 1 = Link Pointer field is not valid. 0 = Link Pointer is valid.

54.5.2.7.2 FSTN Back Path Link Pointer

The second DWord of an FTSN node contains a link pointer to a queue head.

If the T-bit in this pointer is zero, then this FSTN is a Save-Place indicator. Its Typ field must be set by software to indicate the target data structure is a queue head. If the T-bit in this pointer is set to one, then this FSTN is the Restore indicator. When the T-bit is one, the host controller ignores the Typ field.

The following table describes the FSTN back path link pointer fields.

Table 54-35. FSTN Back Path Link Pointer Field Descriptions

Bit	Description
31-5	Back Path Link Pointer (BPLP). This field contains the address of a Queue Head. This field corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	Typ. Software must ensure this field is set to indicate the target data structure is a Queue Head. Any other value in this field yields undefined results.
0	Terminate (T). 1=Link Pointer field is not valid (that is the host controller must not use bits [31:5] as a valid memory address). This value also indicates that this FSTN is a Restore indicator. 0=Link Pointer is valid (that is the host controller may use bits [31:5] (in combination with the CTRLDSSEGMENT register if applicable) as a valid memory address). This value also indicates that this FSTN is a Save-Place indicator.

54.5.3 Host Operational Model

The general operational model is for the enhanced interface host controller hardware and enhanced interface host controller driver (generally referred to as system software).

Each significant operational feature of the EHCI host controller is discussed in a separate section. Each section presents the operational model requirements for the host controller hardware. Where appropriate, recommended system software operational models for features are also presented.

54.5.3.1 Host Controller Initialization

After initial power-on or HCReset (hardware or through HCReset bit in the USB_USBCMD register), all of the operational registers are at their default values. After a hardware reset, only the operational registers not contained in the Auxiliary power well are at their default values.

The following table describes the default values of operational registers.

.

Table 54-36. Default Values of Operational Register Space

Operational Register	Default Value (after Reset)
USB_USBCMD	00080000h (00080B00h, if *Asynchronous Schedule Park Capability is one*)
USB_USBSTS	00001000h
USB_USBINTR	00000000h
USB_FRINDEX	00000000h
USB_CTRLDSSEGMENT	00000000h
USB_PERIODICLISTBASE	Undefined
USB_ASYNCLISTADDR	Undefined
USB_CONFIGFLAG	00000000h
USB_PORTSC1	00002000h (w/*PPC* set to one); 00003000h (w/*PPC* set to zero)

To initialize the host controller, software should perform the following steps:

- Write the appropriate value to the USB_USBINTR register to enable the appropriate interrupts.
- Write the base address of the Periodic Frame List to the USB_PERIODICLIST BASE register. If no work items are in the periodic schedule, all elements of the Periodic Frame List should have their T-Bits set to one.
- Write the USB_USBCMD register to set the desired interrupt threshold, frame list size (if applicable) and turn the host controller ON through setting the Run/Stop bit.

At this point, the host controller is up and running and the port registers begin reporting device connects, and so on. System software can enumerate a port through the reset process (where the port is in the enabled state). At this point, the port is active with SOFs occurring down the enabled ports, but the schedules have not enabled. To communicate with devices through the asynchronous schedule, system software must write the USB_ASYNCLISTADDR register with the address of a control or bulk queue head. Software must then enable the asynchronous schedule by writing one to the Asynchronous Schedule Enable bit in the USB_USBCMD register. To communicate with devices through the periodic schedule, system software must enable the periodic schedule by writing one to the Periodic Schedule Enable bit in the USB_USBCMD register.

NOTE

The schedules can be turned on before the first port is reset (and enabled).

When the USB_USBCMD register is written, system software must ensure the appropriate bits are preserved, depending on the intended operation.

54.5.3.2 Port Routing and Control

The EHCI specification defines that a USB 2.0 Host controller is comprised of one high-speed host controller, which implements the EHCI programming interface and 0 to N USB 1.1 companion host controllers.

Companion host controllers (cHCs) may be implementations of either Universal or Open host controller specifications. This configuration is used to deliver the required full USB 2.0-defined port capability; for example, Low-, Full-, and High-speed capability for every port.

NOTE

The USB controllers on do not require nor support companion controllers to support Full and Low Speed device. Full and Low Speed devices are supported within the USB controller by emulating the functionality of a high-speed HUB. Therefore, no port routing is present in the controller. Please refer to Embedded Transaction Translator Function for details.

The following figure illustrates a simple block diagram of the port routing logic and its relationship to the high-speed and companion host controllers within a USB 2.0 host controller.

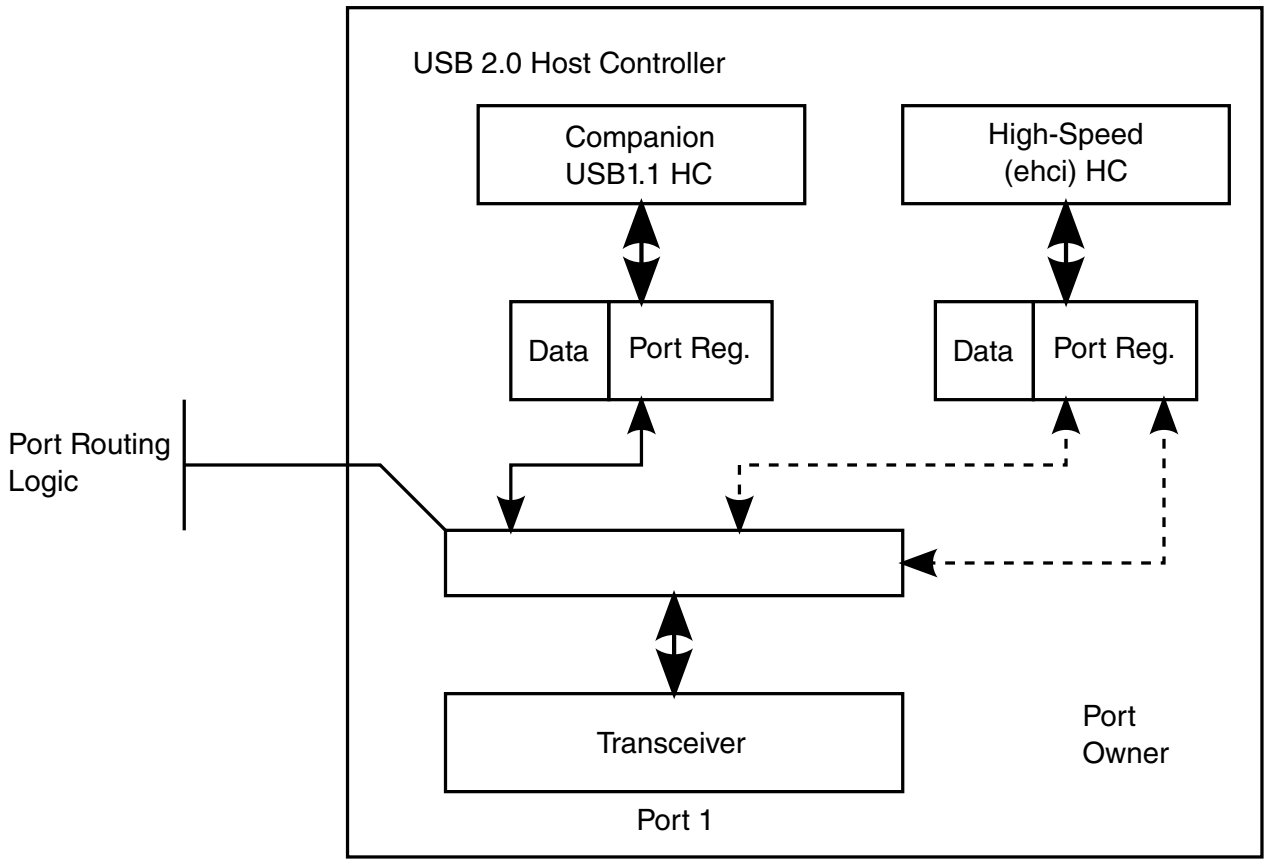


Figure 54-4. Example USB 2.0 Host Controller Port Routing Block Diagram

There exists one transceiver per physical port and each host controller block has its own port status and control registers. The EHCI controller has port status and control registers for every port. Each companion host controller has only the port control and status registers it is required to operate. Either the EHCI host controller or one companion host controller controls each transceiver. Routing logic lies between the transceiver, the port status and control registers.[1]

The port routing logic is controlled from signals originating in the EHCI host controller. The EHCI host controller has a global routing policy control field and per-port ownership control fields. The Configured Flag (CF) bit is the global routing policy control. At power-on or reset, the default routing policy is to the companion controllers (if they exist). If the system does not include a driver for the EHCI host controller and the host controller includes Companion Controllers, then the ports still work in Full- and Low-speed mode (assuming the system includes a driver for the companion controllers). In general, when the EHCI owns the ports, the companion host controllers' port registers do not see a connect indication from the transceiver. Similarly, when a companion host controller owns a port, the EHCI controller's port registers do not see a connect indication

1. The routing logic should not be implemented in the 480 MHz clock domain of the transceiver.

from the transceiver. The details on the rules for the port routing logic are described in the following sections. The USB 2.0 host controller must be implemented as a multi-function PCI device if the implementation includes companion controllers. The companion host controllers' function numbers must be less than the EHCI host controller function number. The EHCI host controller must be a larger function number with respect to the companion host controllers associated with this EHCI host controller. If a PCI device implementation contains only an EHCI controller (that is no companion controllers or other PCI functions), then the EHCI host controller must be function zero, in accordance with the PCI Specification. The N_CC field in the Structural Parameter register (HCSPARAMS) indicates whether the controller implementation includes companion host controllers. When N_CC has a non-zero value there exists companion host controllers. If N_CC has a value of zero, then the host controller implementation does not include companion host controllers. If the host controller root ports are exposed to attachment of full- or low-speed devices, the ports always fails the high-speed chirp during reset and the ports are not enabled. System software can notify the user of the illegal condition. This type of implementation requires a USB 2.0 hub be connected to a root port to provide full and low-speed device connectivity.

System software uses information in the host controller capability registers to determine how the ports are routed to the companion host controllers. See Host Controller Structural Parameters (USB_*n*HCSPARAMS)

54.5.3.2.1 Port Routing Control through EHCI Configured (CF) Bit

Each port in the USB 2.0 host controller are routed either to a single companion host controller or to the EHCI host controller.

The port routing logic is controlled by two mechanisms in the EHCI HC: a host controller global flag and per-port control. The Configured Flag (CF) bit, is used to globally set the policy of the routing logic. Each port register has a Port Owner control bit which allows the EHCI Driver to explicitly control the routing of individual ports. Whenever the CF bit transitions from zero to one (this transition is only available under program control) the port routing unconditionally routes all of the port registers to the EHCI HC (all Port Owner bits go to zero). While the CF-bit is one, the EHCI Driver controls individual ports' routing through the Port Owner control bit. Likewise, whenever the CF bit transitions from one to zero (as a result of Aux power application, HCRESET, or software writing zero to CF-bit), the port routing unconditionally routes all of the port registers to the appropriate companion HC. The default value for the EHCI HC's CF bit (after Aux power application or HCRESET) is zero.

The *view* of the port depends on the current owner. A Universal or Open companion host controller will see port register bits consistent with the appropriate specification. Port bit definitions that are required for EHCI host controllers are not visible to companion host controllers.

The following table summarizes the default routing for all the ports, based on the value of the EHCI HC's CF bit.

Table 54-37. Default Port Routing Depending on EHCI HC CF Bit

HS CF Bit	Default Port Ownership	Explanation
0B	Companion HCs	The companion host controllers own the ports and only Full- and Low-speed devices are supported in the system. The exact port assignments are implementation dependent. The ports behave only as Full- and Low-speed ports in this configuration
1B	EHCI HC	The EHCI host controller has default ownership over all of the ports. The routing logic inhibits device connect events from reaching the companion HCs' port status and control registers when the port owner is the EHCI HC.The EHCI HC has access to the additional port status and control bits defined in this specification (see). The EHCI HC can temporarily release control of the port to a companion HC by setting the *PortOwner* bit in the PORTSC1 register to one.

54.5.3.2.2 Port Routing Control through PortOwner and Disconnect Event

Manipulating the port routing through the CF-bit is an extreme process and not intended to be used during normal operation.

The normal mode of port ownership transferal is on the granularity of individual ports using the Port Owner bit in the EHCI HC's USB_PORTSC1 register (for hand-offs from EHCI to companion host controllers). Individual port ownership is returned to the EHCI controller when the port registers a device disconnect. When the disconnect is detected, the port routing logic immediately returns the port ownership to the EHCI controller. The companion host controller port register detects the device disconnect and operates normally.

Under normal operating conditions (assuming all HC drivers loaded and operational and the EHCI *CF-bit* is set to one), the typical port enumeration sequence proceeds as illustrated below:

- Initial condition is that EHCI is port owner. A device is connected causing the port to detect a connect, set the port connect change bit and issue a port-change interrupt (if enabled).
- EHCI Driver identifies the port with the new connect change bit asserted and sends a change report to the hub driver. Hub driver issues a GetPortStatus() request and

identifies the connect change. It then issues a request to clear the connect change, followed by a request to reset and enable the port.

- When the EHCI Driver receives the request to reset and enable the port, it first checks the value reported by the LineStatus bits in the USB_PORTSC1 register. If they indicate the attached device is a full-speed device (for example, D+ is asserted), then the EHCI Driver sets the PortReset control bit to one (and sets the PortEnable bit to zero) which begins the reset-process. Software times the duration of the reset, then terminates reset signaling by writing zero to the port reset bit. The reset process is actually complete when software reads zero in the PortReset bit. The EHCI Driver checks the PortOwner bit in the USB_PORTSC1 register. If set to one, the connected device is a high-speed device and EHCI Driver (root hub emulator) issues a change report to the hub driver and the hub driver continues to enumerate the attached device.
- At the time the EHCI Driver receives the port reset and enable request the LineStatus bits might indicate a low-speed device. Additionally, when the port reset process is complete, the PortEnable field may indicate that a full-speed device is attached. In either case the EHCI driver sets the PortOwner bit in the USB_PORTSC1 register to one to release port ownership to a companion host controller.
- When the EHCI Driver sets PortOwner bit to one, the port routing logic makes the connection state of the transceiver available to the companion host controller port register and removes the connection state from the EHCI HC port. The EHCI USB_PORTSC1 register observes and reports a disconnect event through the disconnect change bit. The EHCI Driver detects the connection status change (either by polling or by port change interrupt) and then sends a change report to the hub driver. When the hub driver requests that port-state, the EHCI Driver responds with a reset complete change set to one, a connect change set to one and a connect status set to zero. This information is derived directly from the EHCI port register. This allows the hub driver to assume the device was disconnected during reset. It acknowledges the change bits and wait for the next change event. While the EHCI controller does not own the port, it simply remains in a state where the port reports no device connected. The device-connect evaluation circuitry of the companion HC activates and detects the device, the companion Driver detects the connection and enumerates the port.

When a port is routed to a companion HC, it remains under the control of the companion HC until the device is disconnected from the root port (ignoring for now the scenario where EHCI's CF-bit transitions from1b to 0b). When a disconnect occurs, the disconnect event is detected by both the companion HC port control and the EHCI port ownership control. On the event, the port ownership is returned immediately to the EHCI controller. The companion HC stack detects the disconnect and acknowledges as it would in an ordinary standalone implementation. Subsequent connects is detected by the EHCI port register and the process repeats.

54.5.3.2.3 Example Port Routing State Machine

The following figure illustrates an example of how the port ownership should be managed. The following sections describe the entry conditions to each state.

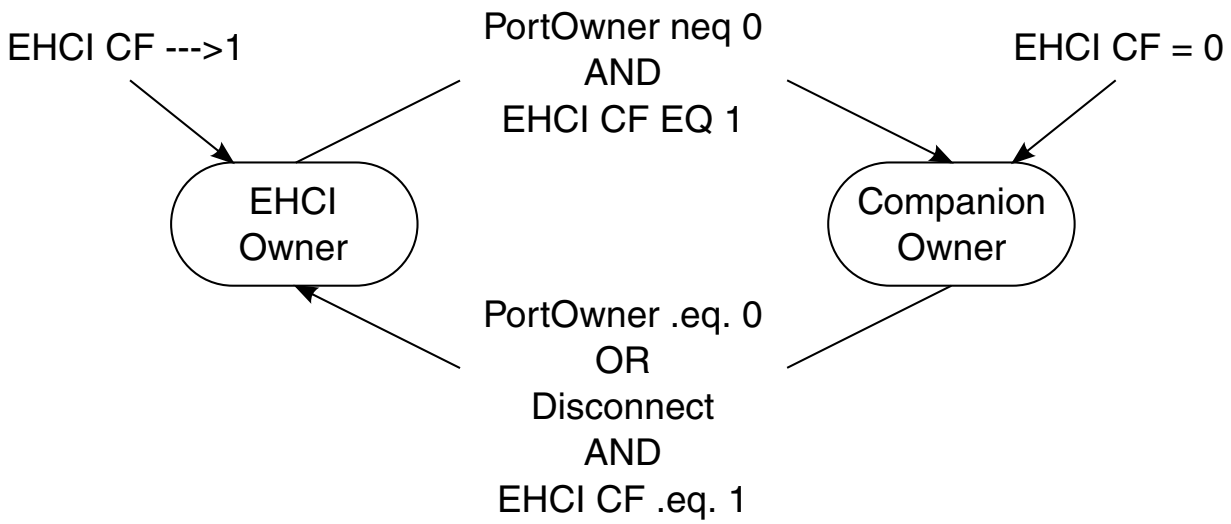


Figure 54-5. Port Owner Handoff State Machine

54.5.3.2.3.1 EHCI HC Owner

Entry to this state occurs when one of the following events occur:

- When the EHCI HC's Configure Flag (CF) bit in the USB_CONFIGFLAG register transitions from zero to one. This signals the fact that the system has a host controller driver for the EHCI HC and that all ports in the USB 2.0 host controller must default route to the EHCI controller.
- When the port is owned by a companion HC and the device is disconnected from the port. The EHCI port routing control logic is notified of the disconnect, and returns port routing to the EHCI controller. The connection state of the companion HC goes immediately to the disconnected state (with appropriate side effect to connect change, enable and enable change). The companion HC driver acknowledges the disconnect by setting the connect status change bit to zero. This allows the companion HC's driver to interact with the port completely through the disconnect process.
- When system software writes zero to the PortOwner bit in the USB_PORTSC1 register. This allows software to take ownership of a port from a companion host controller. When this occurs, the routing logic to the companion HC effectively signals a disconnect to the companion HC's port status and control register.

54.5.3.2.3.2 Companion HC Owner

Entry to this state occurs whenever one of the following events occur:

- When the PortOwner field transitions from zero to one.
- When the HS-mode HC's Configure Flag (CF) is equal to zero.

On entry to this state, the routing logic allows the companion HC port register to detect a device connect. Normal port enumeration proceeds.

54.5.3.2.4 Port Power

The Port Power Control (PPC) bit in the USB_HCSPARAMS register indicates whether the USB 2.0 host controller has port power control (see).

When this bit is zero, then the host controller does not support software control of port power switches. When in this configuration, the port power is always available and the companion host controllers must implement functionality consistent with port power always on. When the *PPC* bit is one, then the host controller implementation includes port power switches. Each available switch has an output enable, which is referred to in this discussion as PortPowerOutputEnable (PPE). PPE is controlled based on the state of the combination bits PPC bit, EHCI Configured (CF)-bit and individual Port Power (PP) bits.

The following table describes the summary behavioral model.

Table 54-38. Port Power Enable Control Rules

CF	CHC[1] (PP)	EHC[2] (PP)	Owner	PPE[3]	Description
0	0	X	CHC	0	When the EHCI controller is not configured, the port is owned by the companion host controller. When the companion HC's port power select is off, then the port power is off.
0	1	X	CHC	1	Similar to previous entry. When the companion HC's port power select is on, then the port power is on.
1	0	0	CHC	0	Port owner has port power turned off, the power to port is off.
1	0	0	EHC	0	Port owner has port power turned off, the power to port is off.
1	0	1	EHC	1	Port owner has port power on, so power to port is on.
1	0	1	CHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	EHC	1	If either HC has port power turned on, the power to the port is on.

Table continues on the next page...

Table 54-38. Port Power Enable Control Rules (continued)

1	1	0	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	EHC	1	Port owner has port power on, so power to port is on.

1. CHC (Companion Host Controller).
2. EHC (EHCI Host Controller).
3. PPE (Port Power Enable). This bit actually turns on the port power switch (if one exists).

54.5.3.2.5 Port Reporting Over-Current

Host controllers are by definition power providers on USB. Whether the ports are considered high- or low-powered is a platform implementation issue. Each EHCI USB_PORTSC1 register has an over-current status and over-current change bit.

The functionality of these bits are specified in the USB Specification Revision 2.0.

The over current detection and limiting logic usually resides outside the host controller logic. This logic may be associated with one or more ports. When this logic detects an over-current condition it is made available to both the companion and EHCI ports. The effect of an over-current status on a companion host controller port is beyond the scope of this document.

The over-current condition effects the following bits in the USB_PORTSC1 register on the EHCI port:

- Over-current Active bits are set to one. When the over-current condition goes away, the Over-current Active bit transitions from one to zero.
- Over-current Change bits are set to one. On every transition of the Over-current Active bit the host controller sets the Over-current Change bit to one. Software sets the Over-current Change bit to zero by writing one to this bit.
- Port Enabled/Disabled bit is set to zero. When this change bit gets set to one, then the Port Change Detect bit in the USB_USBSTS register is set to one.
- Port Power (PP) bits may optionally be set to zero. There is no requirement in USB that a power provider shut off power in an over current condition. It is sufficient to limit the current and leave power applied. When the Over-current Change bit transitions from zero to one, the host controller also sets the Port Change Detect bit in the USB_USBSTS register to one. In addition, if the Port Change Interrupt Enable bit in the USB_USBINTR register is one, then the host controller issues an interrupt to the system. Refer to Table 54-39 for summary behavior for over-current detection when the host controller is halted (suspended from a device component point of view).

54.5.3.3 Suspend/Resume-Host Operational Model

The EHCI host controller provides an equivalent suspend and resume model as that defined for individual ports in a USB 2.0 Hub.

Control mechanisms are provided to allow system software to suspend and resume individual ports. The mechanisms allow the individual ports to be resumed completely through software initiation. Other control mechanisms are provided to parameterize the host controller's response (or sensitivity) to external resume events. In this discussion, host-initiated, or software initiated resumes are called Resume Events/Actions. Bus-initiated resume events are called wake-up events. The classes of wake-up events are:

- Remote-wake-up enabled device asserts resume signaling. In similar kind to USB 2.0 Hubs, EHCI controllers must always respond to explicit device resume signaling and wake-up the system (if necessary).
- Port connect and disconnect and over-current events. Sensitivity to these events can be turned on or off by using the per-port control bits in the USB_PORTSC1 registers.

Selective suspend is a feature supported by every USB_PORTSC1 register. It is used to place specific ports into a suspend mode. This feature is used as a functional component for implementing the appropriate power management policy implemented in a particular operating system. When system software intends to suspend the entire bus, it should selectively suspend all enabled ports, then shut off the host controller by setting the Run/Stop bit in the USB_USBCMD register to zero. The EHCI sub-block can then be placed into a lower device state through the PCI power management interface (see Appendix A, Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com).

When a wake event occurs, the system resumes operation and system software eventually set the Run/Stop bit to one and resume the suspended ports. Software must not set the Run/Stop bit to one until it is confirmed that the clock to the host controller is stable. This is usually confirmed in a system implementation in that all of the clocks in the system are stable before the Arm platform is restarted. So, by definition, if software is running, clocks in the system are stable and the Run/Stop bit in the USB_USBCMD register can be set to one. Minimum system software delays are also defined in the PCI Power Management Specification. Refer to PCI Power Management Specification for more information.

54.5.3.3.1 Port Suspend/Resume

System software places individual ports into suspend mode by writing one into the appropriate USB_PORTSC1 Suspend bit. Software must only set the Suspend bit when the port is in the enabled state (Port Enabled bit is one) and the EHCI is the port owner (PortOwner bit is zero).

The host controller may evaluate the Suspend bit immediately or wait until a micro-frame or frame boundary occurs. If evaluated immediately, the port is not suspended until the current transaction (if one is executing) completes. Therefore, there may be several micro-frames of activity on the port until the host controller evaluates the Suspend bit. The host controller must evaluate the Suspend bit at least every frame boundary.

System software can initiate a resume on a selectively suspended port by writing one to the Force Port Resume bit. Software should not attempt to resume a port unless the port reports that it is in the suspended state (see Port Status & Control (USB_*n*PORTSC1)). If system software sets Force Port Resume bit to one when the port is not in the suspended state, the resulting behavior is undefined. In order to assure proper USB device operation, software must wait for at least 10 ms after a port indicates that it is suspended (Suspend bit is one) before initiating a port resume through the Force Port Resume bit. When Force Port Resume bit is one, the host controller sends resume signaling down the port. System software times the duration of the resume (nominally 20 ms) then sets the Force Port Resume bit to zero. When the host controller receives the write to transition Force Port Resume to zero, it completes the resume sequence as defined in the USB specification, and sets both the Force Port Resume and Suspend bits to zero. Software-initiated port resumes do not affect the Port Change Detect bit in the USB_USBSTS register nor do they cause an interrupt if the Port Change Interrupt Enable bit in the USB_USBINTR register is one. An external USB event may also initiate a resume. The wake events are defined above. When a wake event occurs on a suspended port, the resume signaling is detected by the port and the resume is reflected downstream within 100 µsec. The port's Force Port Resume bit is set to one and the Port Change Detect bit in the USB_USBSTS register is set to one. If the Port Change Interrupt Enable bit in the USB_USBINTR register is one the host controller issues a hardware interrupt.

System software observes the resume event on the port, delays a port resume time (nominally 20 ms), then terminates the resume sequence by writing zero to the Force Port Resume bit in the port. The host controller receives the write of zero to Force Port Resume, terminates the resume sequence and sets Force Port Resume and Suspend port bits to zero. Software can determine that the port is enabled (not suspended) by sampling the USB_PORTSC1 register and observing that the Suspend and Force Port Resume bits are zero. Software must ensure that the host controller is running (that is HCHalted bit in the USB_USBSTS register is zero), before terminating a resume by writing zero to a

port's Force Port Resume bit. If HCHalted is one when Force Port Resume is set to zero, then SOFs do not occur down the enabled port and the device returns to suspend mode in a maximum of 10 msec.

The table below summarizes the wake-up events. Whenever a resume event is detected, the Port Change Detect bit in the USB_USBSTS register is set to one. If the Port Change Interrupt Enable bit is one in the USB_USBINTR register, the host controller generates an interrupt on the resume event. Software acknowledges the resume event interrupt by clearing the Port Change Detect status bit in the USB_USBSTS register.

Table 54-39. Behavior During Wake-up Events

Port Status and Signaling Type	Signaled Port Response	Device State	
		D0	Not D0
Port disabled, resume K-State received	No Effect	N/A	N/A
Port suspended, resume K-State received	Resume reflected downstream on signaled port. Force Port Resume status bit in USB_PORTSC1 register is set to one. Port Change Detect bit in USB_USBSTS register set to one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is one. A disconnect is detected.	Depending in the initial port state, theUSB_ PORTSC1 Connected Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is zero. A disconnect is detected.	Depending on the initial port state, the USB_PORTSC1 Connect and Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]
Port is not connected and the port's WKCNNT_E bit is one. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]
Port is not connected and the port's WKCNNT_E bit is zero. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]
Port is connected and the port's WKOC_E bit is one. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one	[1], [2]	[2]
Port is connected and the port's WKOC_E bit is zero. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]

[1] Hardware interrupt issued if Port Change Interrupt Enable bit in the USB_USBINTR register is one.

[2] PME# asserted if enabled (Note: PME Status must always be set to one).

[3] PME# not asserted.

54.5.3.4 Schedule Traversal Rules

The host controller executes transactions for devices using a simple, shared-memory schedule.

The schedule is comprised of a few data structures, organized into two distinct lists. The data structures are designed to provide the maximum flexibility required by USB, minimize memory traffic and hardware / software complexity.

System software maintains two schedules for the host controller: a periodic schedule and an asynchronous schedule. The root of the periodic schedule is the USB_PERIODICLISTBASE register (see Frame List Base Address (USB_*n*PERIODICLISTBASE))/ Device Address (USB_*n*DEVICEADDR). The USB_PERIODICLISTBASE register is the physical memory base address of the periodic frame list. The periodic frame list is an array of physical memory pointers. The objects referenced from the frame list must be valid schedule data structures as defined in Host Data Structures. In each micro-frame, if the periodic schedule is enabled (see Periodic scheduling threshold) then the host controller must execute from the periodic schedule before executing from the asynchronous schedule. It only executes from the asynchronous schedule after it encounters the end of the periodic schedule. The host controller traverses the periodic schedule by constructing an array offset reference from the USB_PERIODICLISTBASE and the USB_FRINDEX registers (see the following figure). It fetches the element and begins traversing the graph of linked schedule data structures.

The end of the periodic schedule is identified by a next link pointer of a schedule data structure having its T-bit set to one. When the host controller encounters a T-Bit set to one during a horizontal traversal of the periodic list, it interprets this as an End-Of-Periodic-List mark. This causes the host controller to cease working on the periodic schedule and transitions immediately to traversing the asynchronous schedule. After the transition, the host controller executes from the asynchronous schedule until the end of the micro-frame.

The following figure illustrates the derivation of pointer into frame list array.

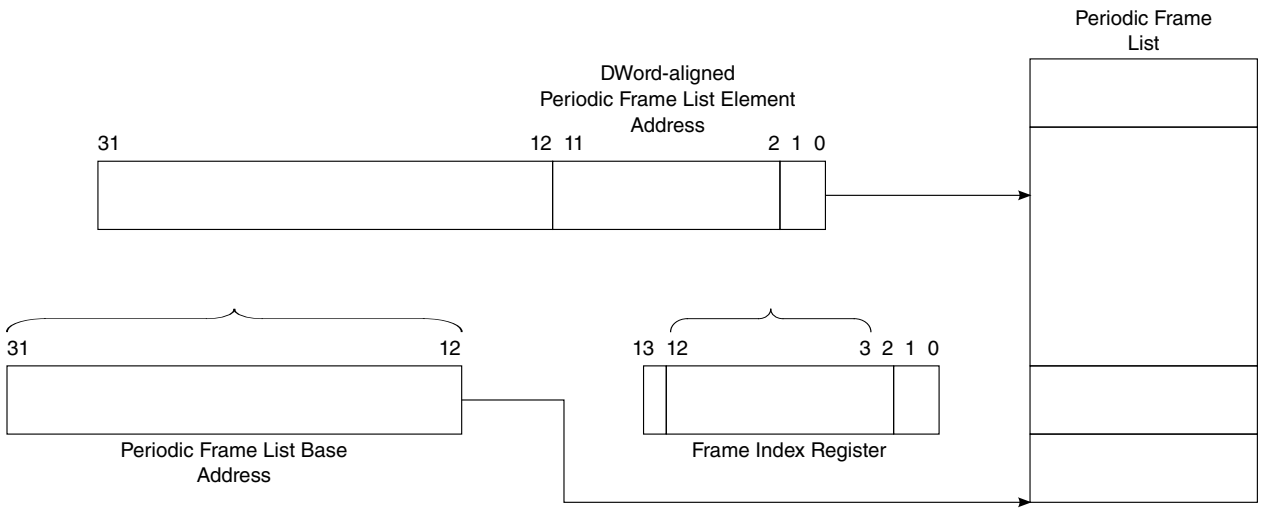


Figure 54-6. Derivation of Pointer into Frame List Array

When the host controller determines that it is the time to execute from the asynchronous list, it uses the operational register USB_ASYNCLISTADDR to access the asynchronous schedule, see the figure below.

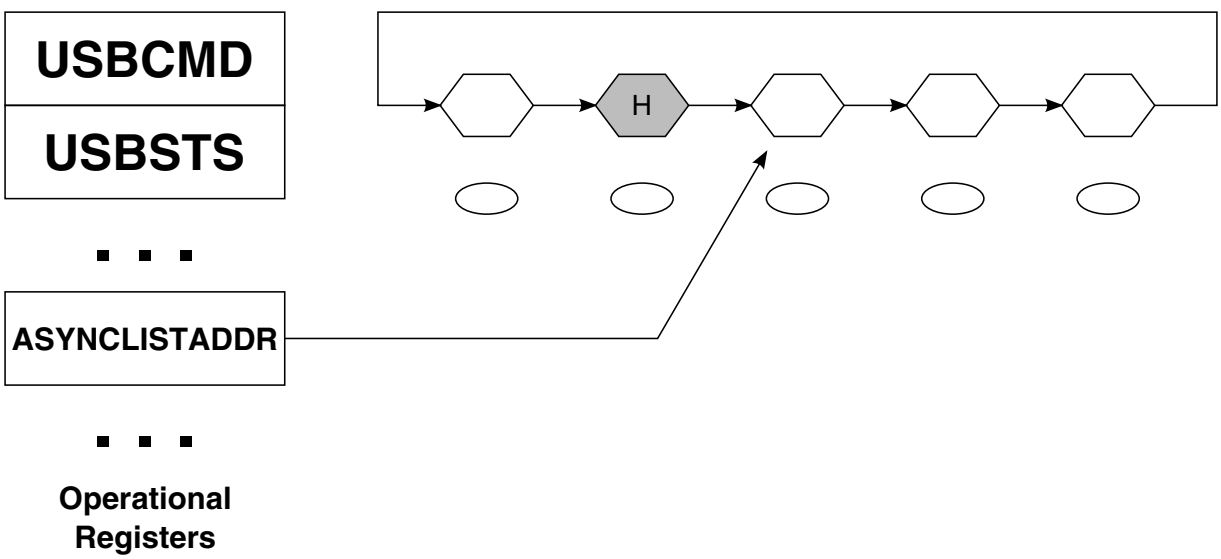


Figure 54-7. General Format of Asynchronous Schedule List

The USB_ASYNCLISTADDR register contains a physical memory pointer to the next queue head. When the host controller makes a transition to executing the asynchronous schedule, it begins by reading the queue head referenced by the USB_ASYNCLISTADDR register. Software must set queue head horizontal pointer T-bits to zero for queue heads in the asynchronous schedule. See Asynchronous Schedule for complete operational details.

54.5.3.4.1 Example - Preserving Micro-Frame Integrity

One of the requirements of a USB host controller is to maintain Frame Integrity. This means that the HC must preserve the micro-frame boundaries.

For example, SOF packets must be generated on time (within the specified allowable jitter), and High-speed EOF1,2 thresholds must be enforced. The end of micro-frame timing points EOF1 and EOF2 are clearly defined in the USB Specification Revision 2.0. One implication of this responsibility is that the HC must ensure that it does not start transactions that do not complete before the end of the micro-frame. More precisely, no transactions should be started by the host controller, which do not complete in their entirety before the EOF1 point. In order to enforce this rule, the host controller must check each transaction before it starts to ensure that it completes before the end of the micro-frame.

So, what exactly needs to be involved in this check? Fundamentally, the transaction data payload, plus bit stuffing, plus transaction overhead must be taken into consideration. It is possible to be extremely accurate on how much time the next transaction takes. Take OUTs for an example. The host controller must fetch all of the OUT data from memory in order to send it onto the USB bus. A host controller implementation could pre-fetch all of the OUT data, and pre-compute the actual number of bits in the token and data packets. In addition, the system knows the depth of the target endpoint, so it could closely estimate turnaround time for handshake. In addition, the host controller knows the size of a handshake packet. Pre-computing effects of bit stuffing and summing up the other overhead numbers can allow the host controller to know exactly whether there is enough bus time, before EOF1 to complete the OUT transaction. To accomplish this particular approach takes an inordinate amount of time and hardware complexity.

The alternative is to make a reasonable guess whether the next transaction can be started. An example approximation algorithm is described below. This example algorithm relies on the EHCI policy that periodic transactions are scheduled first in the micro-frame. It is a reasonable assumption that software never over-commits the micro-frame to periodic transactions greater than the specification allowable 80%. In the available remaining 20% bandwidth, the host controller has some ability (in this example) to decide whether or not to execute a transaction. The result of this algorithm is that sometimes, under some circumstances a transaction is not executed that could have been executed. However, under all circumstances, a transaction is never started unless there is enough time in the frame to complete the transaction.

54.5.3.4.1.1 Transaction Fit - A Best-Fit Approximation Algorithm

A curve is calculated which represents the latest start time for every packet size, at which software schedules the start of a periodic transaction.

This curve is the 80% bandwidth curve. Another curve is calculated which is the absolute, latest permitted start time for every packet size. This curve represents the absolute latest time, that a transaction of each packet size can be started and completed, in the micro-frame. A plot of these two curves are illustrated in Figure 54-8. The plot Y-axis represents the number of byte-times left in a frame.

The space between the 80% and the Last Start plots is bandwidth reclamation area. In this algorithm the host controller may skip transactions during this time if it is prudent.

The Best-Fit Approximation method plots a function ($f(x)$) between the 80% and Last Start curves. The function $f(x)$ adds a constant to every transaction's maximum packet size and the result compared with the number of bytes left in the frame. The constant represents an approximation of the effects of bit stuffing and protocol overhead. The host controller starts transactions whose results land above the function curve. The host controller will not start transactions whose results land below the function curve.

The following figure illustrates the Best-Fit Approximation.

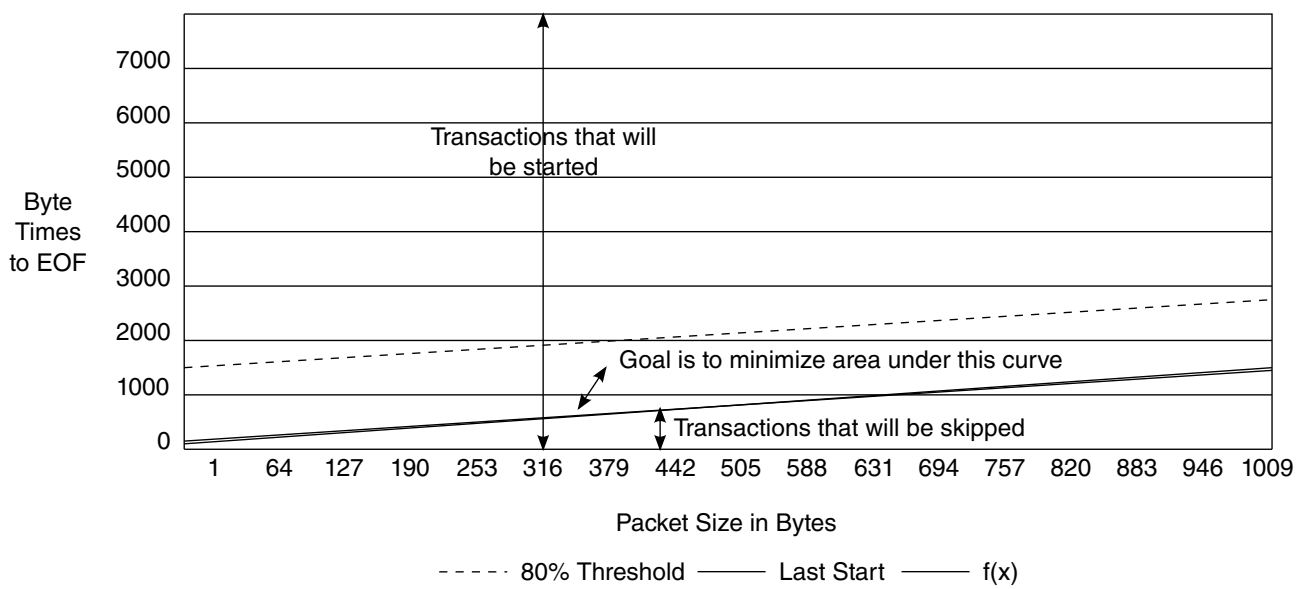


Figure 54-8. Best Fit Approximation

The LastStart line was calculated in this example to assume the absolute worst-case bus overhead per transaction. The particular transaction used is a start-split, zero-length OUT transaction with a handshake. Summaries of the component parts are listed in the table below. The component times were derived from the protocol timings defined in the USB Specification Revision 2.0.

Table 54-40. Example Worse-case Transaction Timing Components

Component	Bit time	Byte Time	Explanation

Table continues on the next page...

Table 54-40. Example Worse-case Transaction Timing Components (continued)

Split Token	76	9.5	Split token as defined in USB core specification. Includes sync, token, eop, and so on.
Host 2 Host IPG	88	11	Number of bit times required between consecutive host packets.
Token	67	8.375	Token as defined in USB core specification. Includes sync, token, eop, and so on.
Host 2 Host IPG	88	11	Token as defined in USB core specification. Includessync, token, eop, and so on.
Data Packet (0 data bytes)	66.7	8.34	Zero-length data packet. Includes sync, PID, crc16, eop, and so on.
Turnaround time	721	90.125	Time for packet initiator (Host) to see the beginning of a response to a transmitted packet.
Handshake packet	48	6	Handshake packet as defined in USB core specification. Includes sync, PID, eop, and so on.
		144	Total

The exact details of the function (f(x)) are up to the particular implementation. However, it should be obvious that the goal is to minimize the area under the curve between the approximation function and the Last Start curve, without dipping below the LastStart line, while at the same time keeping the check as simple as possible for hardware implementation. The f(x) in Figure 54-8 was constructed using the following pseudo-code test on each transaction size data point. This algorithm assumes that the host controller keeps track of the remaining bits in the frame.

```
Alorithm CheckTransactionWillFit (MaximumPacketSize, HC_BytesLeftInFrame)
Begin
Local Temp = MaximumPacketSize + 192
Local rvalue = TRUE
If MaximumPacketSize >= 128 then
     Temp += 128
End If
If Temp > HC_BytesLeftInFrame then
     Rvalue = FALSE
End If
Return rvalue
End
```

This algorithm takes two inputs, the current maximum packet size of the transaction and the hardware counter of the number of bytes left in the current micro-frame. It unconditionally adds a simple constant of 192 to the maximum packet size to account for a first-order effect of transaction overhead and bit stuffing. If the transaction size is greater than or equal to 128 bytes, then an additional constant of 128 is added to the running sum to account for the additional worst-case bit stuffing of payloads larger than 128. An inflection point was inserted at 128 because the $f(x)$ plot was getting close to the LastStart line.

54.5.3.5 Periodic Schedule Frame Boundaries vs Bus Frame Boundaries

The USB Specification Revision 2.0 requires that the frame boundaries (SOF frame number changes) of the high-speed bus and the full- and low-speed bus(s) below USB 2.0 Hubs be strictly aligned.

Super-imposed on this requirement is that USB 2.0 Hubs manage full- and low-speed transactions through a micro-frame pipeline (see start- (SS) and complete- (CS) splits illustrated in the following figure). A simple, direct projection of the frame boundary model into the host controller interface schedule architecture creates tension (complexity for both hardware and software) between the frame boundaries and the scheduling mechanisms required to service the full- and low-speed transaction translator periodic pipelines.

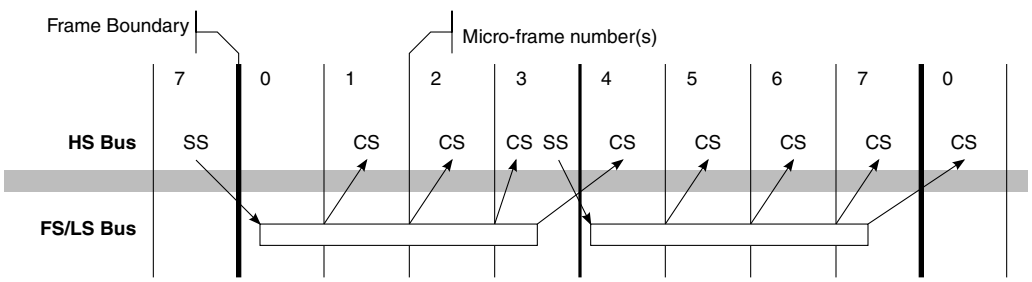


Figure 54-9. Frame Boundary Relationship between HS bus and FS/LS Bus

The simple projection, as the above figure illustrates, introduces frame-boundary wrap conditions for scheduling on both the beginning and end of a frame. In order to reduce the complexity for hardware and software, the host controller is required to implement one micro-frame phase shift for its view of frame boundaries. The phase shift eliminates the beginning of frame and frame-wrap scheduling boundary conditions.

The implementation of this phase shift requires that the host controller use one register value for accessing the periodic frame list and another value for the frame number value included in the SOF token. These two values are separate, but tightly coupled. The periodic frame list is accessed through the Frame List Index Register (USB_FRINDEX) documented in USB Frame Index (USB_*n*FRINDEX) and initially illustrated in Schedule Traversal Rules. Bits FRINDEX[2:0], represent the micro-frame number. The SOF value is coupled to the value of FRINDEX[13:3]. Both FRINDEX[13:3] and the SOF value are increment based on FRINDEX[2:0]. It is required that the SOF value be delayed from the FRINDEX value by one micro-frame. The one micro-frame delay yields host controller periodic schedule and bus frame boundary relationship as illustrated in the following figure. This adjustment allows software to trivially schedule the periodic start and

complete-split transactions for full-and low-speed periodic endpoints, using the natural alignment of the periodic schedule interface. The reasons for selecting this phase-shift are beyond the scope of this specification.

The following figure illustrates how periodic schedule data structures relate to schedule frame boundaries and bus frame boundaries. To aid the presentation, two terms are defined: The host controller's view of the 1 msec boundaries is called H-Frames. The high-speed bus's view of the 1 msec boundaries is called B-Frames.

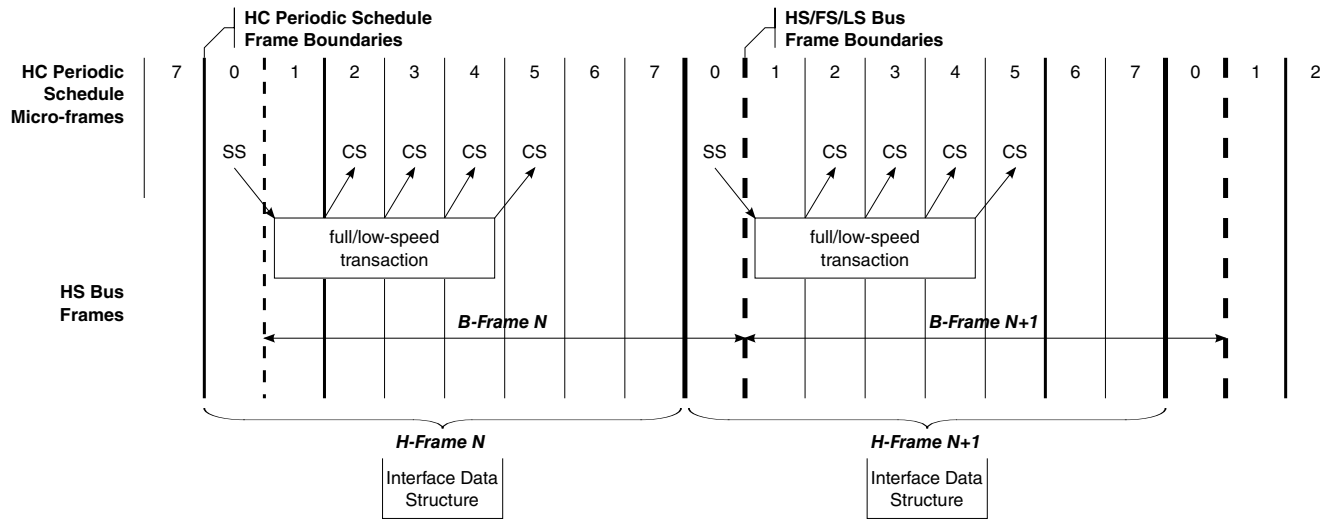


Figure 54-10. Relationship of Periodic Schedule Frame Boundaries to Bus Frame Boundaries

H-Frame boundaries for the host controller correspond to increments of FRINDEX[13:3]. Micro-frame numbers for the H-Frame are tracked by FRINDEX[2:0]. B-Frame boundaries are visible on the high-speed bus through changes in the SOF token's frame number. Micro-frame numbers on the high-speed bus are only derived from the SOF token's frame number (that is the high-speed bus sees eight SOFs with the same frame number value). H-Frames and B-Frames have the fixed relationship (that is B-Frames lag H-Frames by one micro-frame time) illustrated in the figure above. The host controller's periodic schedule is naturally aligned to H-Frames. Software schedules transactions for full- and low-speed periodic endpoints relative the H-Frames. The result is these transactions execute on the high-speed bus at exactly the right time for the USB 2.0 Hub periodic pipeline. As described in USB Frame Index (USB_*n*FRINDEX), the SOF Value can be implemented as a shadow register (in this example, called SOFV), which lags the FRINDEX register bits [13:3] by one micro-frame count. This lag behavior can be accomplished by incrementing FRINDEX[13:3] based on carry-out on the 7 to 0 increment of FRINDEX[2:0] and incrementing SOFV based on the transition of 0 to 1 of FRINDEX[2:0].

Software is allowed to write to FRINDEX. USB Frame Index (USB_*n*FRINDEX) provides the requirements that software should adhere when writing a new value in FRINDEX.

The table below illustrates the required relationship between the value of FRINDEX and the value of SOFV.

Table 54-41. Operation of FRINDEX and SOFV (SOF Value Register)

Current			Next		
FRINDEX[F]	SOFV	FRINDEX[mF]	FRINDEX[F]	SOFV	FRINDEX[mF]
N	N	111b	N+1	N	000b
N+1	N	000b	N+1	N+1	001b
N+1	N+1	001b	N+1	N+1	010b
N+1	N+1	010b	N+1	N+1	011b
N+1	N+1	011b	N+1	N+1	100b
N+1	N+1	100b	N+1	N+1	101b
N+1	N+1	101b	N+1	N+1	110b
N+1	N+1	110b	N+1	N+1	111b

NOTE

Where [F] = [13:3]; [µF] = [2:0]

54.5.3.6 Periodic Schedule

The periodic schedule traversal is enabled or disabled through the Periodic Schedule Enable bit in the USB_USBCMD register. If the Periodic Schedule Enable bit is set to zero, then the host controller simply does not try to access the periodic frame list through the USB_PERIODICLISTBASE register. Likewise, when the Periodic Schedule Enable bit is one, then the host controller does use the USB_PERIODICLISTBASE register to traverse the periodic schedule.

The host controller will not react to modifications to the Periodic Schedule Enable immediately. In order to eliminate conflicts with split transactions, the host controller evaluates the Periodic Schedule Enable bit only when FRINDEX[2:0] is zero. System software must not disable the periodic schedule if the schedule contains an active split transaction work item that spans the 000b micro-frame. These work items must be removed from the schedule before the Periodic Schedule Enable bit is written to zero.

The Periodic Schedule Status bit in the USB_USBSTS register indicates status of the periodic schedule. System software enables (or disables) the periodic schedule by writing one (or zero) to the Periodic Schedule Enable bit in the USB_USBCMD register. Software then can poll the Periodic Schedule Status bit to determine when the periodic

schedule has made the desired transition. Software must not modify the Periodic Schedule Enable bit unless the value of the Periodic Schedule Enable bit equals that of the Periodic Schedule Status bit.

The periodic schedule is used to manage all isochronous and interrupt transfer streams. The base of the periodic schedule is the periodic frame list. Software links schedule data structures to the periodic frame list to produce a graph of scheduled data structures. The graph represents an appropriate sequence of transactions.

The following figure illustrates isochronous transfers (using iTDs and siTDs) with a period of one are linked directly to the periodic frame list. Interrupt transfers (are managed with queue heads) and isochronous streams with periods other than one are linked following the period-one iTD/siTDs. Interrupt queue heads are linked into the frame list ordered by poll rate. Longer poll rates are linked first (for example, closest to the periodic frame list), followed by shorter poll rates, with queue heads with a poll rate of one, on the very end.

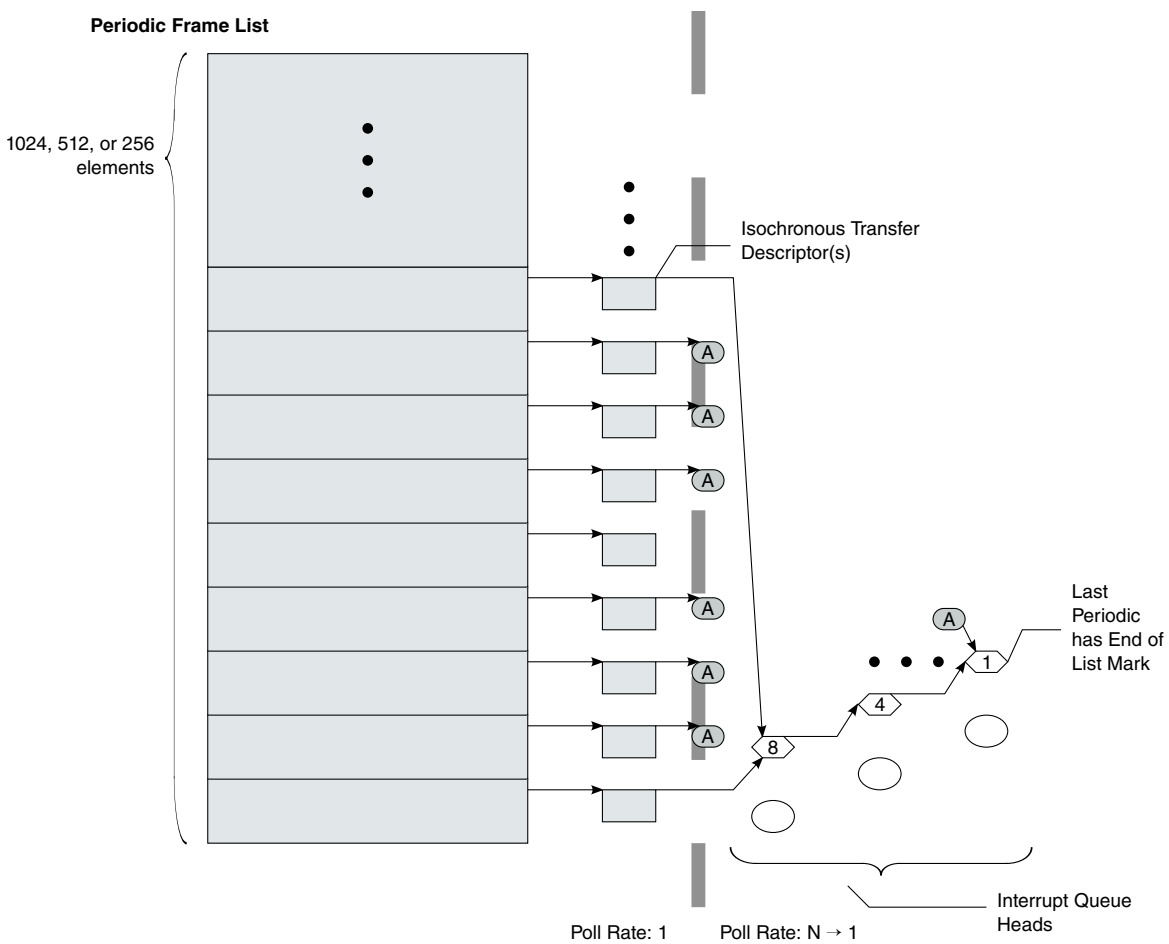


Figure 54-11. Example Periodic Schedule

54.5.3.7 Managing Isochronous Transfers Using iTDs

The structure of an iTD is presented in Isochronous (High-Speed) Transfer Descriptor (iTD). The four distinct sections to an iTD:

- The first field is the Next Link Pointer. This field is for schedule linkage purposes only.
- Transaction description array. This area is an eight-element array. Each element represents control and status information for one micro-frame's worth of transactions for a single high-speed isochronous endpoint.
- The buffer page pointer array is a 7-element array of physical memory pointers to data buffers. These are 4 K aligned pointers to physical memory.
- Endpoint capabilities. This area utilizes the unused low-order 12 bits of the buffer page pointer array. The fields in this area are used across all transactions executed for this iTD, including endpoint addressing, transfer direction, maximum packet size and high-bandwidth multiplier.

54.5.3.7.1 Host Controller Operational Model for iTDs

The host controller uses FRINDEX register bits [12:3] to index into the periodic frame list. This means that the host controller visits each frame list element eight consecutive times before incrementing to the next periodic frame list element. Each iTD contains eight transaction descriptions, which map directly to FRINDEX register bits [2:0]. Therefore, each transaction descriptor corresponds to one micro-frame. Each iTD can span 8 micro-frames worth of transactions.

When the host controller fetches an iTD, it uses FRINDEX register bits [2:0] to index into the transaction description array.

If the active bit in the Status field of the indexed transaction description is set to zero, the host controller ignores the iTD and follows the Next pointer to the next schedule data structure.

When the indexed active bit is one, the host controller continues to parse the iTD. It stores the indexed transaction description and the general endpoint information (device address, endpoint number, maximum packet size, and so on.). It also uses the Page Select (PG) field to index the buffer pointer array, storing the selected buffer pointer and the next sequential buffer pointer. For example, if PG field is 0, then the host controller stores Page 0 and Page 1.

The host controller constructs a physical data buffer address by concatenating the current buffer pointer (as selected using the current transaction description's PG field) and the transaction description's Transaction Offset field. The host controller uses the endpoint addressing information and I/O-bit to execute a transaction to the appropriate endpoint. When the transaction is complete, the host controller clears the active bit and writes back any additional status information to the Status field in the currently selected transaction description.

The data buffer associated with the iTD must be virtually contiguous memory. Seven page pointers are provided to support eight high-bandwidth transactions regardless of the starting packet's offset alignment into the first page. A starting buffer pointer (physical memory address) is constructed by concatenating the page pointer (for example, page 0 pointer) selected by the active transaction descriptions' PG (for example, value: 00B) field with the transaction offset field. As the transaction moves data, the host controller must detect when an increment of the current buffer pointer crosses a page boundary. When this occurs the host controller simply replaces the current buffer pointer's page portion with the next page pointer (for example, page 1 pointer) and continues to move data. The size of each bus transaction is determined by the value in the Maximum Packet Size field. An iTD supports high-bandwidth pipes through the Mult (multiplier) field. When the Mult field is 1, 2, or 3, the host controller executes the specified number of Maximum Packet sized bus transactions for the endpoint in the current micro-frame. In other words, the Mult field represents a transaction count for the endpoint in the current micro-frame. If the Mult field is zero, the operation of the host controller is undefined. The transfer description is used to service all transactions indicated by the Mult field.

For OUT transfers, the value of the Transaction X Length field represents the total bytes to be sent during the micro-frame. The Mult field must be set by software to be consistent with Transaction X Length and Maximum Packet Sixe. The host controller sends the bytes in Maximum Packet Size'd portions. After each transaction, the host controller decrements its local copy of Transaction X Length by Maximum Packet Size. The number of bytes the host controller sends is always Maximum Packet Size or Transaction X Length, whichever is less. The host controller advances the transfer state in the transfer description, updates the appropriate record in the iTD and moves to the next schedule data structure. The maximum sized transaction supported is 3 x 1024 bytes.

For IN transfers, the host controller issues Mult transactions. It is assumed that software has properly initialized the iTD to accommodate all of the possible data. During each IN transaction, the host controller must use Maximum Packet Size to detect packet babble errors. The host controller keeps the sum of bytes received in the Transaction X Length field. After all transactions for the endpoint have completed for the micro-frame, Transaction X Length contains the total bytes received. If the final value of Transaction X Length is less than the value of Maximum Packet Size, then less data than was allowed for was received from the associated endpoint. This short packet condition does not set

the USBINT bit in the USB_USBSTS register to one. The host controller will not detect this condition. If the device sends more than Transaction X Length or Maximum Packet Size bytes (whichever is less), then the host controller sets the Babble Detected bit to one and set the Active bit to zero. Note, that the host controller is not required to update the iTD field Transaction X Length in this error scenario. If the Mult field is greater than one, then the host controller automatically executes the value of Mult transactions. The host controller will not execute all Mult transactions if:

- The endpoint is an OUT and Transaction X Length goes to zero before all the Mult transactions have executed (ran out of data), or
- The endpoint is an IN and the endpoint delivers a short packet, or an error occurs on a transaction before Mult transactions have been executed. The end of micro-frame may occur before all of the transaction opportunities have been executed. When this happens, the transfer state of the transfer description is advanced to reflect the progress that was made, the result written back to the iTD and the host controller proceeds to processing the next micro-frame. Refer to Appendix D for a table summary of the host controller required behavior for all the high-bandwidth transaction cases.

54.5.3.7.2 Software Operational Model for iTDs

A client buffer request to an isochronous endpoint may span 1 to N micro-frames. When N is larger than one, system software may have to use multiple iTDs to read or write data with the buffer (if N is larger than eight, it must use more than one iTD).

The following figure illustrates the simple model of how a client buffer is mapped by system software to the periodic schedule (that is the periodic frame list and a set of iTDs). On the right is the client description of its request. The description includes a buffer base address plus additional annotations to identify which portions of the buffer should be used with each bus transaction. In the middle is the iTD data structures used by the system software to service the client request. Each iTD can be initialized to service up to 24 transactions, organized into eight groups of up to three transactions each. Each group maps to one micro-frame's worth of transactions. The EHCI controller does not provide per-transaction results within a micro-frame. It treats the per-micro-frame transactions as a single logical transfer. On the left is the host controller's frame list. System software establishes references from the appropriate locations in the frame list to each of the appropriate iTDs. If the buffer is large, then system software can use a small set of iTDs to service the entire buffer. System software can activate the transaction description records (contained in each iTD) in any pattern required for the particular data stream.

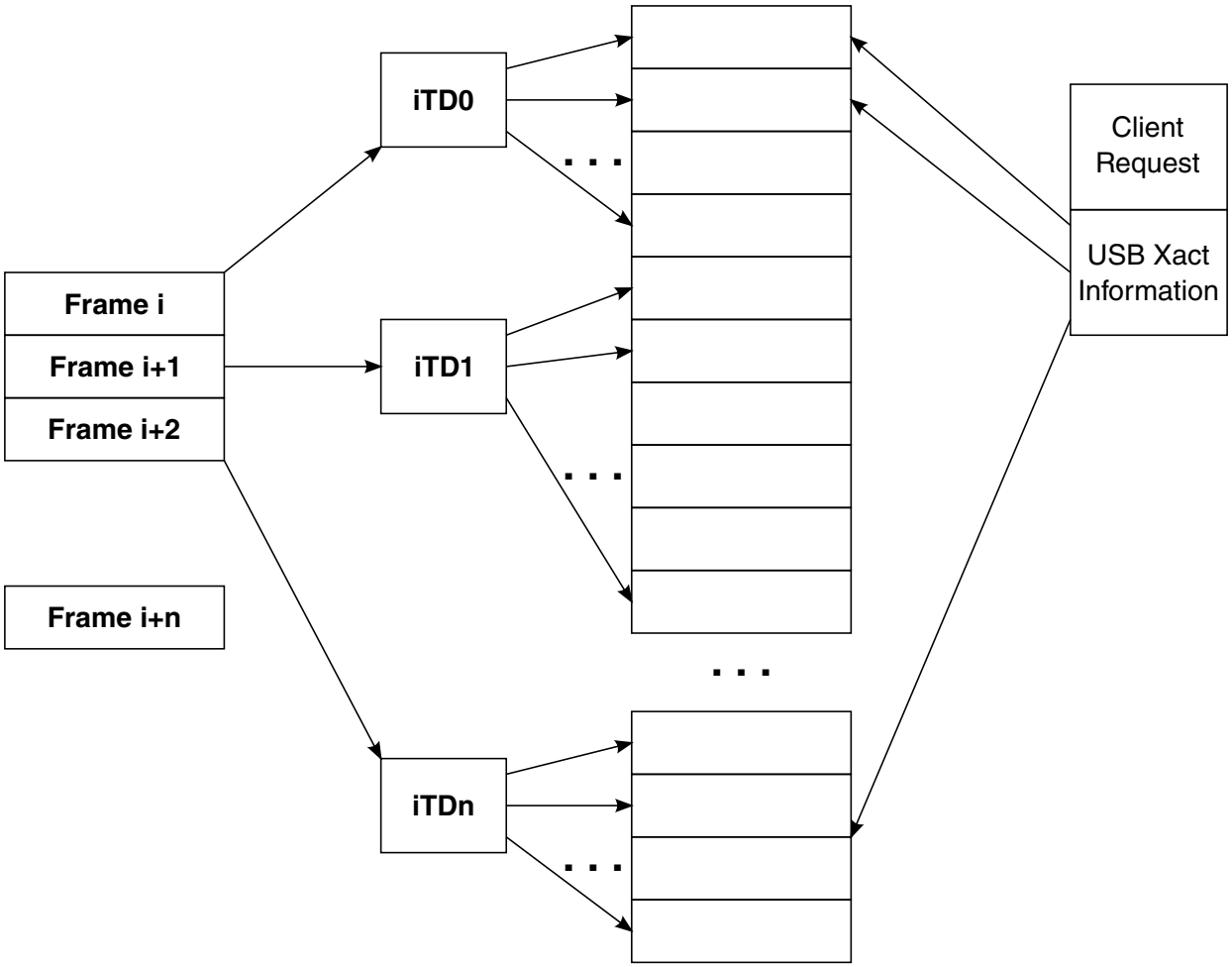


Figure 54-12. Example Association of iTDs to Client Request Buffer

As noted above, the client request includes a pointer to the base of the buffer and offsets into the buffer to annotate which buffer sections are to be used on each bus transaction that occurs on this endpoint. System software must initialize each transaction description in an iTD to ensure it uses the correct portion of the client buffer. For example, for each transaction description, the PG field is set to index the correct physical buffer page pointer and the Transaction Offset field is set relative to the correct buffer pointer page (for example, the same one referenced by the PG field). When the host controller executes a transaction it selects a transaction description record based on FRINDEX[2:0]. It then uses the current Page Buffer Pointer (as selected by the PG field) and concatenates to the transaction offset field. The result is a starting buffer address for the transaction. As the host controller moves data for the transaction, it must watch for a page wrap condition and properly advance to the next available Page Buffer Pointer. System software must not use the Page 6 buffer pointer in a transaction description where the length of the transfer wraps a page boundary. Doing so yields undefined behavior. The host controller hardware is not required to 'alias' the page selector to Page zero. USB 2.0 isochronous

endpoints can specify a period greater than one. Software can achieve the appropriate scheduling by linking iTDs into the appropriate frames (relative to the frame list) and by setting appropriate transaction description elements active bits to one.

54.5.3.7.2.1 Periodic scheduling threshold

The Isochronous Scheduling Threshold field in the USB_HCCPARAMS capability register is an indicator to system software as to how the host controller pre-fetches and effectively caches schedule data structures.

It is used by system software when adding isochronous work items to the periodic schedule. The value of this field indicates to system software the minimum distance it can update isochronous data (relative to the current location of the host controller execution in the periodic list) and still have the host controller process them.

The iTD and siTD data structures each describe 8 micro-frames worth of transactions. The host controller is allowed to cache one (or more) of these data structures in order to reduce memory traffic. Three basic caching models that account for the fact the isochronous data structures span 8 micro-frames. The three caching models are: no caching, micro-frame caching and frame caching.

When software is adding new isochronous transactions to the schedule, it always performs a read of the USB_FRINDEX register to determine the current frame and micro-frame the host controller is currently executing. Of course, there is no information about where in the micro-frame the host controller is, so a constant uncertainty-factor of one micro-frame has to be assumed. Combining the knowledge of where the host controller is executing with the knowledge of the caching model allows the definition of simple algorithms for how closely software can reliably work to the executing host controller.

No caching is indicated with a value of zero in the Isochronous Scheduling Threshold field. The host controller may pre-fetch data structures during a periodic schedule traversal (per micro-frame) but always dumps any accumulated schedule state at the end of the micro-frame. At the appropriate time relative to the beginning of every micro-frame, the host controller always begins schedule traversal from the frame list. Software can use the value of the USB_FRINDEX register (plus the constant 1 uncertainty-factor) to determine the approximate position of the executing host controller. When no caching is selected, software can add an isochronous transaction as near as 2 micro-frames in front of the current executing position of the host controller.

Frame caching is indicated with a non-zero value in bit [7] of the Isochronous Scheduling Threshold field. In the frame-caching model, system software assumes that the host controller caches one (or more) isochronous data structures for an entire frame (8 micro-frames). Software uses the value of the USB_FRINDEX register (plus the constant 1

uncertainty) to determine the current micro-frame/frame (assume modulo 8 arithmetic in adding the constant 1 to the micro-frame number). For any current frame N, if the current micro-frame is 0 to 6, then software can safely add isochronous transactions to Frame N + 1. If the current micro-frame is 7, then software can add isochronous transactions to Frame N + 2.

Micro-frame caching is indicated with a non-zero value in the least-significant 3 bits of the Isochronous Scheduling Threshold field. System software assumes the host controller caches one or more periodic data structures for the number of micro-frames indicated in the Isochronous Scheduling Threshold field. For example, if the count value were 2, then the host controller keeps a window of 2 micro-frames worth of state (current micro-frame, plus the next) on-chip. On each micro-frame boundary, the host controller releases the current micro-frame state and begins accumulating the next micro-frame state.

54.5.3.8 Asynchronous Schedule

The Asynchronous schedule traversal is enabled or disabled through the Asynchronous Schedule Enable bit in the USB_USBCMD register.

If the Asynchronous Schedule Enable bit is set to zero, then the host controller simply does not try to access the asynchronous schedule through the USB_ASYNCLISTADDR register. Likewise, when the Asynchronous Schedule Enable bit is one, then the host controller does use the USB_ASYNCLISTADDR register to traverse the asynchronous schedule. Modifications to the Asynchronous Schedule Enable bit are not necessarily immediate. Rather the new value of the bit is taken into consideration the next time the host controller needs to use the value of the USB_ASYNCLISTADDR register to get the next queue head.

The Asynchronous Schedule Status bit in the USB_USBSTS register indicates status of the asynchronous schedule. System software enables (or disables) the asynchronous schedule by writing one (or zero) to the Asynchronous Schedule Enable bit in the USB_USBCMD register. Software then can poll the Asynchronous Schedule Status bit to determine when the asynchronous schedule has made the desired transition. Software must not modify the Asynchronous Schedule Enable bit unless the value of the Asynchronous Schedule Enable bit equals that of the Asynchronous Schedule Status bit.

The asynchronous schedule is used to manage all Control and Bulk transfers. Control and Bulk transfers are managed using queue head data structures. The asynchronous schedule is based at the USB_ASYNCLISTADDR register. The default value of the USB_ASYNCLISTADDR register after reset is undefined and the schedule is disabled when the Asynchronous Schedule Enable bit is zero.

Software may only write this register with defined results when the schedule is disabled. For example, Asynchronous Schedule Enable bit in the USB_USBCMD and the Asynchronous Schedule Status bit in the USB_USBSTS register are zero. System software enables execution from the asynchronous schedule by writing a valid memory address (of a queue head) into this register. Then software enables the asynchronous schedule by setting the Asynchronous Schedule Enable bit to one. The asynchronous schedule is actually enabled when the Asynchronous Schedule Status bit is one.

When the host controller begins servicing the asynchronous schedule, it begins by using the value of the USB_ASYNCLISTADDR register. It reads the first referenced data structure and begins executing transactions and traversing the linked list as appropriate. When the host controller completes processing the asynchronous schedule, it retains the value of the last accessed queue head's horizontal pointer in the USB_ASYNCLISTADDR register. Next time the asynchronous schedule is accessed, this is the first data structure that is serviced. This provides round-robin fairness for processing the asynchronous schedule.

A host controller completes processing the asynchronous schedule when one of the following events occur:

- The end of a micro-frame occurs.
- The host controller detects an empty list condition (see Empty Asynchronous Schedule Detection)
- The schedule has been disabled through the Asynchronous Schedule Enable bit in the USB_USBCMD register.

The queue heads in the asynchronous list are linked into a simple circular list as shown in Figure 54-7. Queue head data structures are the only valid data structures that may be linked into the asynchronous schedule. An isochronous transfer descriptor (iTD or siTD) in the asynchronous schedule yields undefined results.

The maximum packet size field in a queue head is sized to accommodate the use of this data structure for all non-isochronous transfer types. The USB Specification, Revision 2.0 specifies the maximum packet sizes for all transfer types and transfer speeds. System software should always parameterize the queue head data structures according to the core specification requirements.

54.5.3.8.1 Adding Queue Heads to Asynchronous Schedule

This is a software requirement section.

There are two independent events for adding queue heads to the asynchronous schedule. The first is the initial activation of the asynchronous list. The second is inserting a new queue head into an activated asynchronous list.

Activation of the list is simple. System software writes the physical memory address of a queue head into the USB_ASYNCLISTADDR register, then enables the list by setting the Asynchronous Schedule Enable bit in the USB_USBCMD register to one.

When inserting a queue head into an active list, software must ensure that the schedule is always coherent from the host controllers' point of view. This means that the system software must ensure that all queue head pointer fields are valid. For example, qTD pointers have T-Bits set to one or reference valid qTDs and the Horizontal Pointer references a valid queue head data structure. The following algorithm represents the functional requirements:

```
InsertQueueHead (pQHeadCurrent, pQueueHeadNew)
          --
          -- Requirement: all inputs must be properly initialized.
          --
          -- pQHeadCurrent is a pointer to a queue head that is
          -- already in the active list
          -- pQHeadNew is a pointer to the queue head to be added
          --
          -- This algorithm links a new queue head into a existing
          -- list
          --
          pQueueHeadNew.HorizontalPointer  =  pQueueHeadCurrent.HorizontalPointer
          pQueueHeadCurrent.HorizontalPointer =  physicalAddressOf(pQueueHeadNew)
End InsertQueueHead
```

54.5.3.8.2 Removing Queue Heads from Asynchronous Schedule

This is a software requirement section.

There are two independent events for removing queue heads from the asynchronous schedule. The first is shutting down (deactivating) the asynchronous list. The second is extracting a single queue head from an activated list.

Software deactivates the asynchronous schedule by setting the Asynchronous Schedule Enable bit in the USB_USBCMD register to zero. Software can determine when the list is idle when the Asynchronous Schedule Status bit in the USB_USBSTS register is zero. The normal mode of operation is that software removes queue heads from the asynchronous schedule without shutting it down. Software must not remove an active queue head from the schedule. Software should first deactivate all active qTDs, wait for the queue head to go inactive, then remove the queue head from the asynchronous list. Software removes a queue head from the asynchronous list through the following algorithm. As illustrated, the unlinking is quite easy. Software merely must ensure all of the link pointers reachable by the host controller are kept consistent.

```
UnlinkQueueHead (pQHeadPrevious, pQueueHeadToUnlink, pQHeadNext)
          --
          -- Requirement: all inputs must be properly initialized.
          --
          -- pQHeadPrevious is a pointer to a queue head that
          -- references the queue head to remove
          -- pQHeadToUnlink is a pointer to the queue head to be
```

```
        -- removed
        -- pQheadNext is a pointer to a queue head still in the
        -- schedule. Software provides this pointer with the
        -- following strict rules:
        --          if the host software is one queue head, then
        --          pQHeadNext must be the same as
        --          QueueheadToUnlink.HorizontalPointer. If the host
        --          software is unlinking a consecutive series of
        --          queue heads, QHeadNext must be set by software to
        --          the  queue head remaining in the schedule.
        --
        -- This algorithm unlinks a queue head from a circular list
        --
        pQueueHeadPrevious.HorizontalPointer =  pQueueHeadToUnlink.HorizontalPointer
                          pQueueHeadToUnlink.HorizontalPointer = pQHeadNext
End UnlinkQueueHead
```

If software removes the queue head with the H-bit set to one, it must select another queue head still linked into the schedule and set its H-bit to one. This should be completed before removing the queue head. The requirement is that software keep one queue head in the asynchronous schedule, with its H-bit set to one. At the point software has removed one or more queue heads from the asynchronous schedule, it is unknown whether the host controller has a cached pointer to them. Similarly, it is unknown how long the host controller might retain the cached information, as it is implementation dependent and may be affected by the actual dynamics of the schedule load. Therefore, once software has removed a queue head from the asynchronous list, it must retain the coherency of the queue head (link pointers, and so on). It cannot disturb the removed queue heads until it knows that the host controller does not have a local copy of a pointer to any of the removed data structures.

The method software uses to determine when it is safe to modify a removed queue head is to handshake with the host controller. The handshake mechanism allows software to remove items from the asynchronous schedule, then execute a simple, lightweight handshake that is used by software as a key that it can free (or reuse) the memory associated the data structures it has removed from the asynchronous schedule.

The handshake is implemented with three bits in the host controller. The first bit is a command bit (Interrupt on Async Advance Doorbell bit in the USB_USBCMD register) that allows software to inform the host controller that something has been removed from its asynchronous schedule. The second bit is a status bit (Interrupt on Async Advance bit in the USB_USBSTS register) that the host controller sets after it has released all on-chip state that may potentially reference one of the data structures just removed. When the host controller sets this status bit to one, it also sets the command bit to zero. The third bit is an interrupt enable (Interrupt on Async Advance bit in the USB_USBINTR register) that is matched with the status bit. If the status bit is one and the interrupt enable bit is one, then the host controller asserts a hardware interrupt.

The figure below illustrates a general example. In this example, consecutive queue heads (B and C) are unlinked from the schedule using the algorithm above. Before the unlink operation, the host controller has a copy of queue head A.

The unlink algorithm requires that as software unlinks each queue head, the unlinked queue head is loaded with the address of a queue head that remains in the asynchronous schedule.

When the host controller observes that doorbell bit being set to one, it makes a note of the local reachable schedule information. In this example, the local reachable schedule information includes both queue heads (A and B). It is sufficient that the host controller can set the status bit (and clear the doorbell bit) as soon as it has traversed beyond current reachable schedule information (that is traversed beyond queue head (B) in this example). The following figure illustrates the generic queue head unlink scenario.

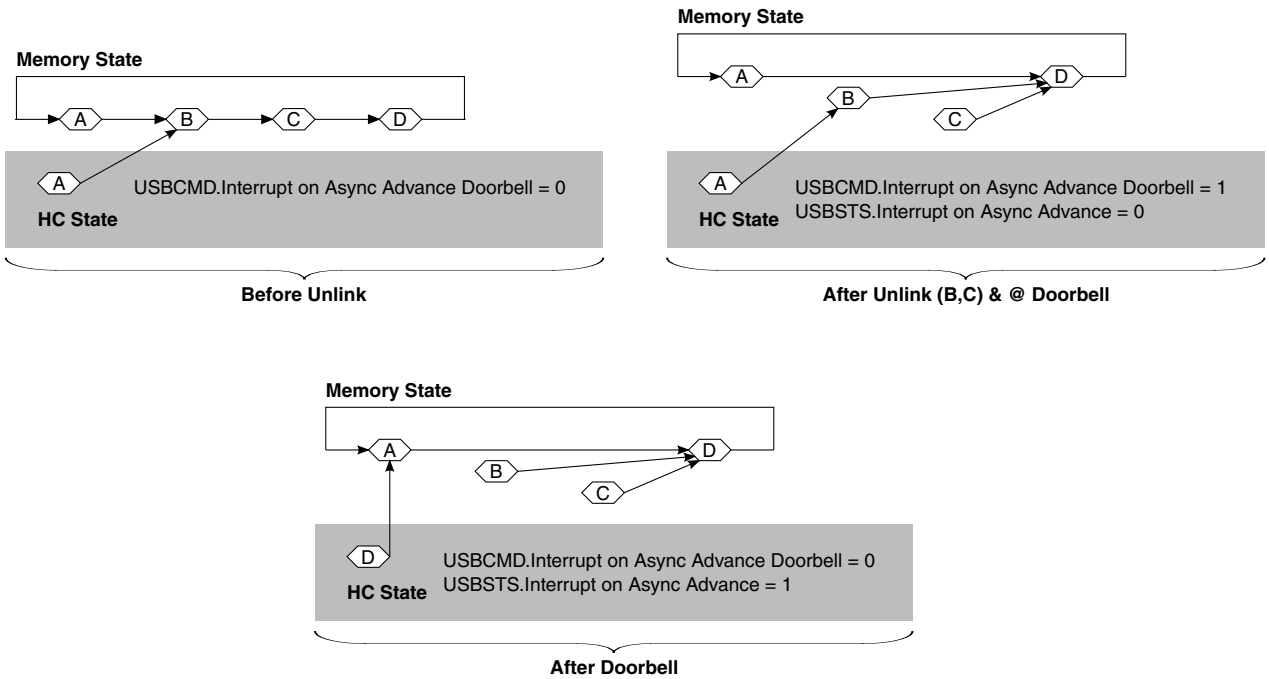


Figure 54-13. Generic Queue Head Unlink Scenario

Alternatively, a host controller implementation is allowed to traverse the entire asynchronous schedule list (for example, observed the head of the queue (twice)) before setting the Advance on Async status bit to one.

Software may re-use the memory associated with the removed queue heads after it observes the Interrupt on Async Advance status bit is set to one, following assertion of the doorbell. Software should acknowledge the Interrupt on Async Advance status as indicated in the USB_USBSTS register, before using the doorbell handshake again.

54.5.3.8.3 Empty Asynchronous Schedule Detection

The Enhanced Host Controller Interface uses two bits to detect when the asynchronous schedule is empty.

The queue head data structure (see Table 54-27) defines an *H-bit* in the queue head, which allows software to mark a queue head as being the *head* of the reclaim list. The Enhanced Host Controller Interface also keeps a 1-bit flag in the USB_USBSTS register (*Reclamation*) that is set to zero when the Enhanced Interface Host Controller observes a queue head with the H-bit set to one. The reclamation flag in the status register is set to one when any USB transaction from the asynchronous schedule is executed (or whenever the asynchronous schedule starts, see Asynchronous schedule traversal: *Start Event*).

If the Enhanced Host Controller Interface ever encounters an *H-bit* of one and a *Reclamation* bit of zero, the EHCI controller simply stops traversal of the asynchronous schedule.

An example illustrating the H-bit in a schedule is shown in the following figure.

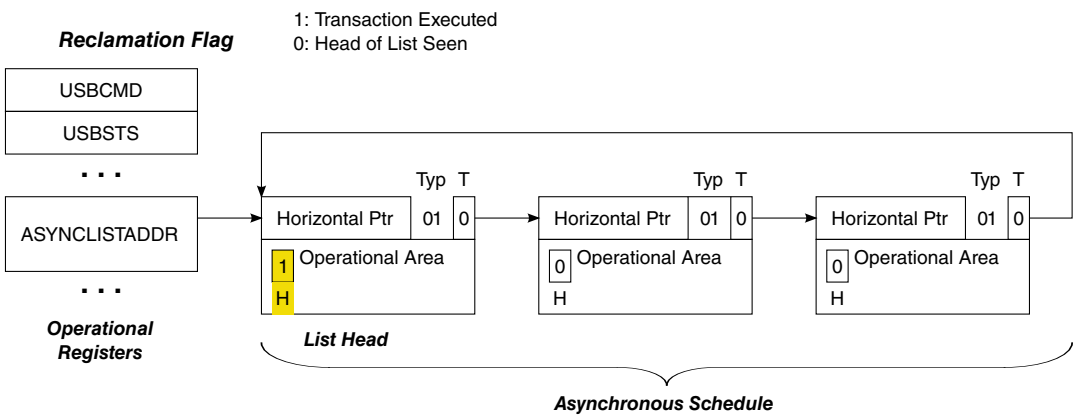


Figure 54-14. Asynchronous Schedule List w/Annotation to Mark Head of List

Software must ensure there is at most one queue head with the *H-bit* set to one, and that it is always coherent with respect to the schedule.

54.5.3.8.4 Restarting Asynchronous Schedule Before EOF

There are many situations where the host controller will detect an empty list *long* before the end of the micro-frame.

It is important to remember that under many circumstances the schedule traversal has stopped due to Nak/Nyet responses from all endpoints.

An example of particular interest is when a start-split for a bulk endpoint occurs early in the micro-frame. Given the EHCI simple traversal rules, the complete-split for that transaction may Nak/Nyet out very quickly. If it is the only item in the schedule, then the host controller ceases traversal of the Asynchronous schedule very early in the micro-frame. In order to provide reasonable service to this endpoint, the host controller should issue the complete-split before the end of the current micro-frame, instead of waiting

until the next micro-frame. When the reason for host controller idling asynchronous schedule traversal is because of empty list detection, it is mandatory the host controller implement a 'waking' method to resume traversal of the asynchronous schedule. An example method is described below.

54.5.3.8.4.1 Example Method for Restarting Asynchronous Schedule Traversal

The reason for idling the host controller when the list is empty is to keep the host controller from unnecessarily occupying too much memory bandwidth. The question is: *how long should the host controller stay idle before restarting*?

The answer in this example is based on deriving a manifest constant, which is the amount of time the host controller will stay idle before restarting traversal. In this example, the manifest constant is called *AsyncSchedSleepTime*, and has a value of 10 µsec. The value is derived based on the analysis in Example Derivation for *AsyncSchedSleepTime*, The traversal algorithm is simple:

- Traverse the Asynchronous schedule until the either an End-Of-micro-Frame event occurs, or an empty list is detected. If the event is an End-of-micro-Frame, go attempt to traverse the Periodic schedule. If the event is an empty list, then set a sleep timer and go to a *schedule sleep* state.
- When the sleep timer expires, set working context to the Asynchronous Schedule start condition and go to *schedule active* state. The start context allows the HC to reload *Nakcnt* fields, and so on. So the HC has a chance to run for more than one iteration through the schedule.

This process simply repeats itself each micro-frame. The figure below illustrates a sample state machine to manage the active and sleep states of the Asynchronous Schedule traversal policy. There are three states: Actively traversing the Asynchronous schedule, Sleeping, and Not Active. The last two are similar in terms of interaction with the Asynchronous schedule, but the Not Active state means that the host controller is busy with the Periodic schedule or the Asynchronous schedule is not enabled. The Sleeping state is specifically a special state where the host controller is just waiting for a period of time before resuming execution of the Asynchronous schedule.

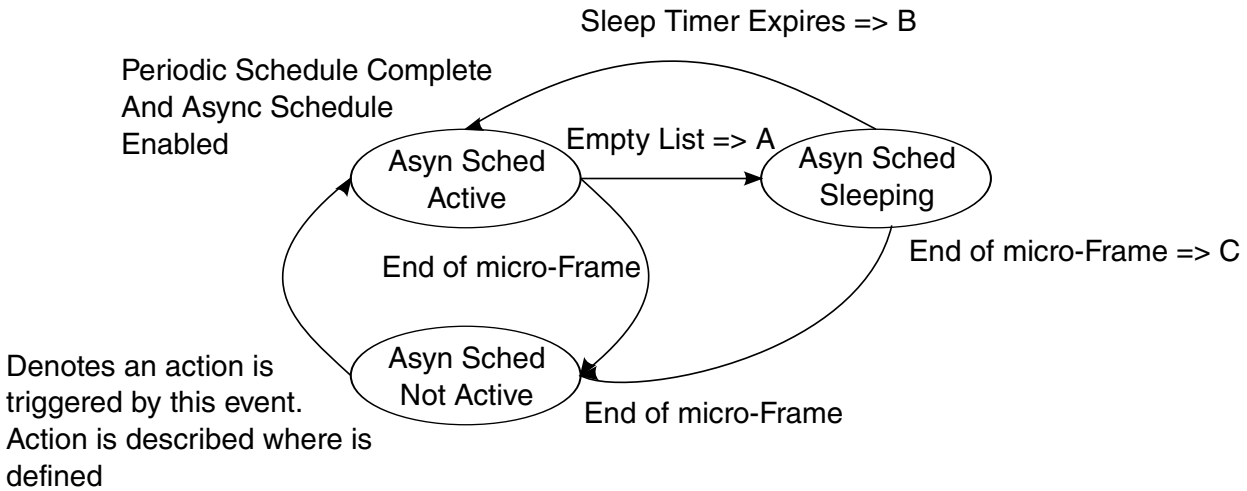


Figure 54-15. Example State Machine for Managing Asynchronous Schedule Traversal

The actions referred to in the figure above are defined in the following table.

Table 54-42. Asynchronous Schedule SM Transition Actions

Action	Action Description Label
A	On detection of the empty list, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.
B	When the *AsynchronousTraversalSleepTimer* expires, the host controller sets the *Reclamation* bit in the USBSTS register to one and moves the Nak Counter reload state machine to WaitForListHead (see Nak Count Reload Control).
C	The host controller cancels the sleep timer (*AsynchronousTraversalSleepTimer*).

54.5.3.8.4.2 Async Sched Not Active

This is the initial state of the traversal state machine after a host controller reset. The traversal state machine does not leave this state when the *Asynchronous Schedule Enable* bit in the USB_USBCMD register is zero.

This state is entered from Async Sched Active or Async Sched Sleeping states when the end-of-micro-frame event is detected.

54.5.3.8.4.3 Async Sched Active

This state is entered from the Async Sched Not Active state when the periodic schedule is not active. It is also entered from the Async Sched Sleeping states when the *AsyncrhonousTraversalSleepTimer* expires. On every transition into this state, the host controller sets the *Reclamation* bit in the USB_USBSTS register to one.

While in this state, the host controller continually traverses the asynchronous schedule until either the end of micro-frame or an empty list condition is detected.

54.5.3.8.4.4 Async Sched Sleeping

The state is entered from the Async Sched Active state when a schedule empty condition is detected. On entry to this state, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.

54.5.3.8.4.5 Example Derivation for *AsyncSchedSleepTime*

The derivation is based on analysis of what work the host controller could be doing next.

It assumes the host controller does not keep any state about what work is possibly pending in the asynchronous schedule. The schedule could contain any mix of the possible combinations of high- full- or low-speed control and bulk requests.

The table below summarizes some of the typical 'next transactions' that could be in the schedule, and the amount of time (for example *footprint*, or *wall clock*) the transaction takes to complete.

Table 54-43. Typical Low-/Full-speed Transaction Times

Transaction Attributes		Footprint (time)	Description
Speed	HS	11.9 ms	Maximum foot print for a worst-case, full-sized bulk data transaction.
Size	512	9.45 ms	Maximum footprint for an approximate best-case, full-sized bulk data transaction.
Type	Bulk		
Speed	FS	~50 ms	Approximate typical for full-sized bulk data. An 8-byte low-speed is about 2x, or between 90 and 100 ms.
Size	64		
Type	Bulk		
Speed	FS	~12 ms	Approximate typical for 8-byte bulk/control (that is setup)
Size	8		
Type	Cntrl		

A *AsyncSchedSleepTime* value of 10 µs provides a reasonable relaxation of the system memory load and still provides a good level of service for the various transfer types and payload sizes. For example, say we detect an empty list after issuing a start-split for a 64-byte full-speed bulk request. Assuming this is the only thing in the list, the host controller gets the results of the full-speed transaction from the hub during the fifth complete-split request. If the full-speed transaction was an IN and it nak'd, the 10 µs sleep period would allow the host controller to get the NAK results on the first complete-split.

54.5.3.8.5 Asynchronous schedule traversal: *Start Event*

Once the HC has *idled* itself through the empty schedule detection (Section 0), it will naturally *activate* and begin processing from the Periodic Schedule at the beginning of each micro-frame.

In addition, it may have idled itself early in a micro-frame. When this occurs (idles early in the micro-frame) the HC must occasionally *re-activate* during the micro-frame and traverse the asynchronous schedule to determine whether any progress can be made. The requirements and method for this restart are described in Restarting Asynchronous Schedule Before EOF . Asynchronous schedule *Start Event*s are defined to be:

- Whenever the host controller transitions from the periodic schedule to the asynchronous schedule. If the periodic schedule is disabled and the asynchronous schedule is enabled, then the beginning of the micro-frame is equivalent to the transition from the periodic schedule, or
- The asynchronous schedule traversal restarts from a sleeping state (see Restarting Asynchronous Schedule Before EOF).

54.5.3.8.6 Reclamation Status Bit (USBSTS Register)

The operation of the empty asynchronous schedule detection feature (see Empty Asynchronous Schedule Detection) depends on the proper management of the *Reclamation* bit in the USB_USBSTS register. The host controller tests for an empty schedule just after it fetches a new queue head while traversing the asynchronous schedule (see Fetch Queue Head).

The host controller is required to set the *Reclamation* bit to one whenever an asynchronous schedule traversal *Start Event*, as documented in Asynchronous schedule traversal: *Start Event*, occurs. The *Reclamation* bit is also set to one whenever the host controller executes a transaction while traversing the asynchronous schedule (see Execute Transaction). The host controller sets the *Reclamation* bit to zero whenever it finds a queue head with its *H-bit* set to one. Software should only set a queue head's *H-bit* if the queue head is in the asynchronous schedule. If software sets the *H-bit* in an interrupt queue head to one, the resulting behavior is undefined. The host controller may set the *Reclamation* bit to zero when executing from the periodic schedule.

54.5.3.9 Operational Model for Nak Counter

This section describes the operational model for the *NakCnt* field defined in a queue head.

See Queue Head Initialization for more information. Software should not use this feature for interrupt queue heads. This rule is not required to be enforced by the host controller.

USB protocol has built-in flow control through the Nak response by a device. There are several scenarios, beyond the Ping feature, where an endpoint may naturally Nak or Nyet the majority of the time. An example is the host controller management of the split transaction protocol for control and bulk endpoints. All bulk endpoints (High- or Full-speed) are serviced through the same asynchronous schedule. The time between the *Start-split* transaction and the first *Complete-split* transaction could be very short (that is like when the endpoint is the only one in the asynchronous schedule). The hub NYETs (effectively Naks) the *Complete-split* transaction until the classic transaction is complete. This could result in the host controller thrashing memory, repeatedly fetching the queue head and executing the transaction to the Hub, which does not complete until after the transaction on the classic bus completes.

The two component fields in a queue head to support the throttling feature: a counter field (*NakCnt*), and a counter reload field (*RL*). *NakCnt* is used by the host controller as one of the criteria to determine whether or not to execute a transaction to the endpoint. The two operational modes associated with this counter:

- Not Used- This mode is set when the *RL* field is zero. The host controller ignores the *NakCnt* field for any execution of transactions through a queue head with an *RL* field of zero. Software must use this selection for interrupt endpoints.
- Nak Throttle Mode- This mode is selected when the *RL* field is non-zero. In this mode, the value in the *NakCnt* field represents the maximum number of Nak or Nyet responses the host controller tolerates on each endpoint. In this mode, the HC decrements the *NakCnt* field based on the token/handshake criteria listed in the table below. The host controller must reload *NakCnt* when the endpoint successfully moves data (for example, policy to reward device for moving data).

The following table describes the NakCnt field adjustment rules.

Table 54-44. NakCnt Field Adjustment Rules

Token	Handshake	
	Handshake NAK	NYET
IN/PING	decrement *NakCnt*	N/A (protocol error)
OUT	decrement *NakCnt*	No Action[1] Start
Split	decrement *NakCnt*	N/A (protocol error)
Complete Split	No Action	Decrement *NakCnt*

1. Recommended behavior on this response is to reload *NakCnt*

In summary, system software enables the counter by setting the reload field (*RL*) to a non-zero value. The host controller may execute a transaction if *NakCnt* is non-zero. The host controller does not execute a transaction if *NakCnt* is zero. The reload mechanism is described in detail in Nak Count Reload Control .

NOTE

When all queue heads in the Asynchronous Schedule either exhausts all transfers or all NakCnt's go to zero, then the host controller detects an empty Asynchronous Schedule and idle schedule traversal (see Empty Asynchronous Schedule Detection).

Any time the host controller begins a new traversal of the Asynchronous Schedule, a *Start Event* is assumed, see Asynchronous schedule traversal: *Start Event*. Every time a Start-Event occurs, the Nak Count reload procedure is enabled.

54.5.3.9.1 Nak Count Reload Control

When the host controller reaches the *Execute Transaction* state for a queue head (meaning that it has an active operational state), it checks to determine whether the *NakCnt* field should be reloaded from *RL* (see Execute Transaction). If the answer is yes, then *RL* is copied into *NakCnt*. After the reload or if the reload is not active, the host controller evaluates whether to execute the transaction.

The host controller must reload nak counters (*NakCnt* see Table 54-27) in queue heads during the first pass through the reclamation list after an asynchronous schedule Start Event (see Asynchronous schedule traversal: *Start Event* for the definition of the Start Event). The Asynchronous Schedule should have at most one queue head marked as the head (see Figure 54-14).

The following figure illustrates an example state machine that satisfies the operational requirements of the host controller detecting the first pass through the Asynchronous Schedule. This state machine is maintained internal to the host controller and is only used to gate reloading of the nak counter during the queue head traversal state: Execute Transaction (see the figure below). The host controller does not perform the nak counter reload operation if the RL field (see Table 54-27) is set to zero.

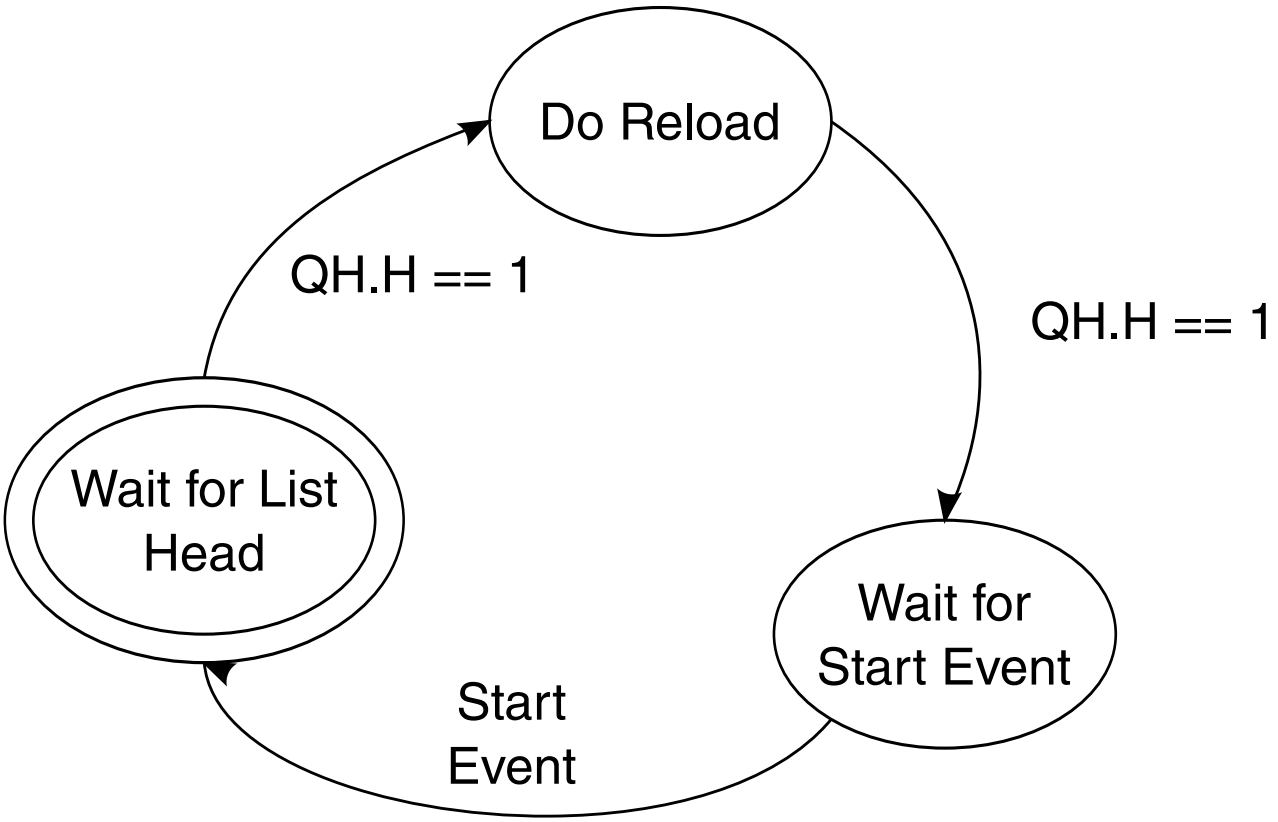


Figure 54-16. Example HC State Machine for Controlling Nak Counter Reloads

54.5.3.9.1.1 Wait for List Head

This is the initial state.

The state machine enters this state from Wait for Start Event when a start event as defined in Asynchronous schedule traversal: *Start Event* occurs.

The purpose of this state is to wait for the first observation of the head of the Asynchronous Schedule.

This occurs when the host controller fetches a queue head whose *H-bit* is set to one.

54.5.3.9.1.2 Do Reload

This state is entered from the Wait for List Head state when the host controller fetches a queue head with the *H-bit* set to one. While in this state, the host controller performs nak counter reloads for every queue head visited that has a non-zero nak reload value (*RL*) field.

54.5.3.9.1.3 Wait for Start Event

This state is entered from the *Do Reload* state when a queue head with the *H-bit* set to one is fetched. While in this state, the host controller does not perform nak counter reloads.

54.5.3.10 Managing Control/Bulk/Interrupt Transfers through Queue Heads

This section presents an overview of how the host controller interacts with queuing data structures.

Queue heads use the Queue Element Transfer Descriptor (qTD) structure. One queue head is used to manage the data stream for one endpoint. The queue head structure contains static endpoint characteristics and capabilities. It also contains a working area from where individual bus transactions for an endpoint are executed (see Overlay area defined in Table 54-27). Each qTD represents one or more bus transactions, which is defined in the context of this specification as a *transfer*.

The general processing model for the host controller's use of a queue head is simple:

- read a queue head,
- execute a transaction from the overlay area,
- write back the results of the transaction to the overlay area,
- move to the next queue head.

If the host controller encounters errors during a transaction, the host controller sets one (or more) of the error reporting bits in the queue head's *Status* field. The *Status* field accumulates all errors encountered during the execution of a qTD (for example, the error bits in the queue head *Status* field are 'sticky' until the transfer (qTD) has completed). This state is always written back to the source qTD when the transfer is complete. On transfer (for example, buffer or halt conditions) boundaries, the host controller must auto-advance (without software intervention) to the next qTD. Additionally, the hardware must be able to halt the queue so no additional bus transactions occurs for the endpoint and the host controller does not advance the queue.

An example host controller operational state machine of a queue head traversal is illustrated in the following figure. This state machine is a model for how a host controller should traverse a queue head. The host controller must be able to advance the queue from the *Fetch QH* state in order to avoid all hardware/software race conditions. This simple mechanism allows software to simply link qTDs to the queue head and *activate* them, then the host controller always *find* them if/when they are reachable. The figure below illustrates the Host Controller Queue Head Traversal State Machine.

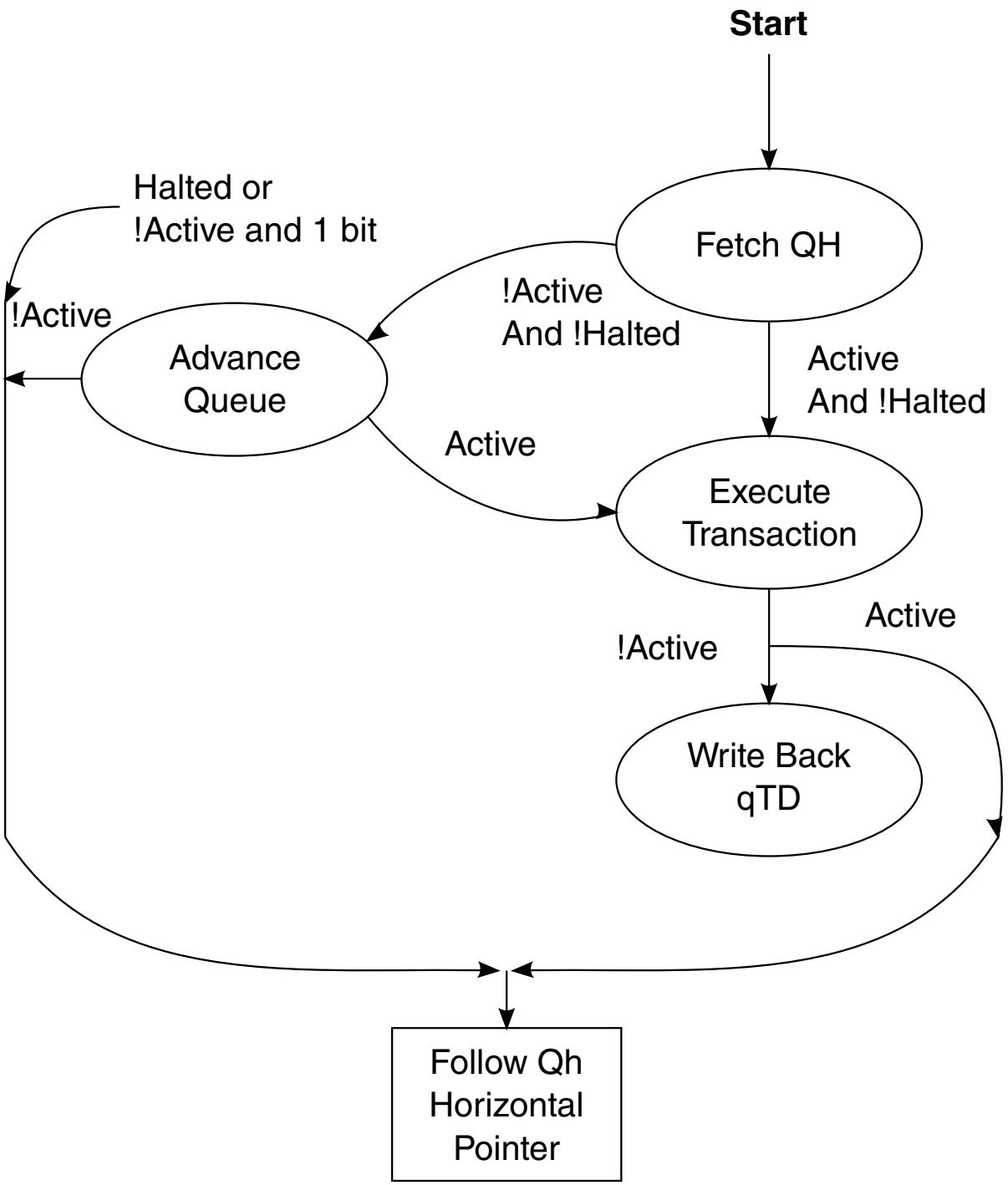


Figure 54-17. Host Controller Queue Head Traversal State Machine

This traversal state machine applies to all queue heads, regardless of transfer type or whether split transactions are required. The following sections describe each state. Each state description describes the entry criteria. The Execute Transaction state (see Execute

Transaction) describes the basic requirements for all endpoints. Split Transactions for Asynchronous Transfers and Split Transaction Interrupt describe details of the required extensions to the Execute Transaction state for endpoints requiring split transactions.

NOTE

Prior to software placing a queue head into either the periodic or asynchronous list, software must ensure the queue head is properly initialized. Minimally, the queue head should be initialized to the following (see Section Queue Head for layout of a queue head):

Valid static endpoint state.

- For the very first use of a queue head, software may zero-out the queue head transfer overlay, then set the *Next qTD Pointer* field value to reference a valid qTD.

54.5.3.10.1 Fetch Queue Head

A queue head can be referenced from the physical address stored in the ASYNCLISTADDR Register (see Next Asynch. Address (USB_*n*ASYNCLISTADDR))/ Endpoint List Address (USB_*n*ENDPTLISTADDR) Additionally, it may be referenced from the *Next LinkPointer* field of an iTD, siTD, FSTN or another Queue Head. If the referencing link pointer has the *Typ* field set to indicate a queue head, it is assumed to reference a queue head structure as defined in Table 54-27.

While in this state, the host controller performs operations to implement empty schedule detection (see Empty Asynchronous Schedule Detection) and Nak Counter reloads (see Operational Model for Nak Counter). After the queue head has been fetched, the host controller conducts the following queries for empty schedule detection:

- If queue head is not an interrupt queue head (that is *S-mask* is zero), and
- The *H-bit* is one, and
- The *Reclamation* bit in the USBSTS register is zero.

When these criteria are met, the host controller stops traversing the asynchronous list (as described in Empty Asynchronous Schedule Detection). When the criteria are not met, the host controller continues schedule traversal. If the queue head is not an interrupt and the *H-bit* is one and the *Reclamation* bit is one, then the host controller sets the *Reclamation* bit in the USBSTS register to zero before completing this state. The operations for reloading of the Nak Counter are described in detail in Operational Model for Nak Counter.

This state is complete when the queue head has been read on-chip.

54.5.3.10.2 Advance Queue

To advance the queue, the host controller must find the next qTD, adjust pointers, perform the overlay and write back the results to the queue head.

This state is entered from the FetchQHD state if the overlay *Active* and *Halt* bits are set to zero. On entry to this state, the host controller determines which next pointer to use to fetch a qTD, fetches a qTD and determines whether or not to perform an overlay.

NOTE

If the *I-bit* is one and the *Active* bit is zero, the host controller immediately skips processing of this queue head, exits this state and uses the horizontal pointer to the next schedule data structure. If the field *Bytes to Transfer* is not zero and the *T-bit* in the *Alternate Next qTD Pointer* is set to zero, then the host controller uses the *Alternate Next qTD Pointer*. Otherwise, the host controller uses the *NextqTD Pointer*. If *NextqTD Pointer's T-bit* is set to one, then the host controller exits this state and uses the horizontal pointer to the next schedule data structure.

Using the selected pointer the host controller fetches the referenced qTD. If the fetched qTD has its *Active* bit set to one, the host controller moves the pointer value used to reach the qTD (*Next* or *Alternate Next*) to the *Current qTD Pointer* field, then performs the overlay. If the fetched qTD has its *Active* bit set to zero, the host controller aborts the queue advance and follows the queue head's horizontal pointer to the next schedule data structure.

The host controller performs the overlay based on the following rules:

- The value of the data toggle (*dt*) field in the overlay area depends on the value of the *data toggle control* (*dtc*) bit (see Table 54-29).
- If the *EPS* field indicates the endpoint is a high-speed endpoint, the *Ping* state field is preserved by the host controller. The value of this field is not changed as a result of the overlay.
- *C-prog-mask* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *Frame Tag* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *NakCnt* field in the overlay area is loaded from the *RL* field in the queue head's Static Endpoint State.
- All other areas of the overlay are set by the incoming qTD.

The host controller exits this state when it has committed the write to the queue head.

54.5.3.10.3 Execute Transaction

The host controller enters this state from the Fetch Queue Head state only if the *Active* bit in *Status* field of the queue head is set to one.

On entry to this state, the host controller executes a few pre-operations, then checks some pre-condition criteria before committing to executing a transaction for the queue head.

The pre-operations performed and the pre-condition criteria depend on whether the queue head is an interrupt endpoint. The host controller can determine that a queue head is an interrupt queue head when the queue head's *S-mask* field contains a non-zero value. It is the responsibility of software to ensure the *S-mask* field is appropriately initialized based on the transfer type. There are other criteria that must be met if the *EPS* field indicates that the endpoint is a low- or full-speed endpoint, see Split Transactions for Asynchronous Transfers and Split Transaction Interrupt .

54.5.3.10.3.1 Interrupt Transfer Pre-condition Criteria

If the queue head is for an interrupt endpoint (for example, non-zero *S-mask* field), then the FRINDEX[2:0] field must identify a bit in the *S-mask* field that has one in it.

For example, an *S-mask* value of 00100000b would evaluate to true only when FRINDEX[2:0] is equal to 101b. If this condition is met then the host controller considers this queue head for a transaction.

54.5.3.10.3.2 Asynchronous Transfer Pre-operations and Pre-condition Criteria

If the queue head is not for an interrupt endpoint (for example, zero *S-mask* field), then the host controller performs one pre-operation and then evaluates one pre-condition criteria.

The pre-operation is:

Checks the Nak counter reload state (Operational Model for Nak Counter). It may be necessary for the host controller to reload the Nak Counter field. The reload is performed at this time.

The pre-condition evaluated is:

- Whether or not the *NakCnt* field has been reloaded, the host controller checks the value of the *NakCnt* field in the queue head. If *NakCnt* is non-zero, or if the *Reload Nak Counter* field is zero, then the host controller considers this queue head for a transaction.

54.5.3.10.3.3 Transfer Type Independent Pre-operations

Regardless of the transfer type, the host controller always performs at least one pre-operation and evaluates one pre-condition. The pre-operation is:

- A host controller internal transaction (down) counter *qHTransactionCounter* is loaded from the queue head's *Mult* field. A host controller implementation is allowed to ignore this for queue heads on the asynchronous list. It is mandatory for interrupt queue heads. Software should ensure that the *Mult* field is set appropriately for the transfer type.

The pre-conditions evaluated are:

- The host controller determines whether there is enough time in the micro-frame to complete this transaction (see Transaction Fit - A Best-Fit Approximation Algorithm for an example evaluation method). If there is not enough time to complete the transaction, the host controller exits this state.
- If the value of *qHTransactionCounter* for an interrupt endpoint is zero, then the host controller exits this state.

When the pre-operations are complete and pre-conditions are met, the host controller sets the *Reclamation* bit in the USBSTS register to one and then begins executing one or more transactions using the endpoint information in the queue head. The host controller iterates *qHTransactionCounter* times in this state executing transactions. After each transaction is executed, *qHTransactionCounter* is decremented by one. The host controller exits this state when one of the following events occurs:

- The *qHTransactionCounter* decrements to zero, or
- The endpoint responds to the transaction with any handshake other than an ACK,[4] or
- The transaction experiences a transaction error, or
- The *Active* bit in the queue head goes to zero, or
- There is not enough time in the micro-frame left to execute the next transaction(see Transaction Fit - A Best-Fit Approximation Algorithm) for example method for implementing the frame boundary test).

NOTE

For a high-bandwidth interrupt OUT endpoint, the host controller may optionally immediately retry the transaction if it fails.

The results of each transaction is recorded in the on-chip overlay area. If data was successfully moved during the transaction, the transfer state in the overlay area is advanced. To advance queue head's transfer state, the *Total Bytes to Transfer* field is decremented by the number of bytes moved in the transaction, the data toggle bit (*dt*) is toggled, the current page offset is advanced to the next appropriate value (for example,

advanced by the number of bytes successfully moved), and the *C_Page* field is updated to the appropriate value (if necessary). See Buffer Pointer List Use for Data Streaming with qTDs .

NOTE

The *Total Bytes To Transfer* field may be zero when all the other criteria for executing a transaction are met. When this occurs, the host controller executes zero-length transaction to the endpoint. If the *PID_Code* field indicates an IN transaction and the device delivers data, the host controller detects a packet babble condition, set the *babble* and *halted* bits in the *Status* field, set the *Active* bit to zero, write back the results to the source qTD, then exit this state.

In the event an IN token receives a data PID mismatch response, the host controller must ignore the received data (for example not advance the transfer state for the bytes received). Additionally, if the endpoint is an interrupt IN, then the host controller must record that the transaction occurred (for example, decrement *qHTransactionCounter*). It is recommended (but not required) the host controller continue executing transactions for this endpoint if the resultant value of *qHTransactionCounter* is greater than one.

If the response to the IN bus transaction is a Nak (or Nyet) and *RL* is non-zero, *NakCnt* is decremented by one. If *RL* is zero, then no write-back by the host controller is required (for a transaction receiving a Nak or Nyet response and the value of *CErr* did not change). Software should set the *RL* field to zero if the queue head is an interrupt endpoint. Host controller hardware is not required to enforce this rule or operation.

After the transaction has finished and the host controller has completed the post processing of the results (advancing the transfer state and possibly *NakCnt*, the host controller writes back the results of the transaction to the queue head's overlay area in main memory).

The number of bytes moved during an IN transaction depends on how much data the device endpoint delivers. The maximum number of bytes a device can send is *MaximumPacket Size*. The number of bytes moved during an OUT transaction is either *Maximum Packet Length* bytes or *Total Bytes to Transfer*, whichever is less.

If there was a transaction error during the transaction, the transfer state (as defined above) is not advanced by the host controller. The *CErr* field is decremented by one and the status field is updated to reflect the type of error observed. Transaction errors are summarized in Transaction Error .

The following events causes the host controller to clear the *Active* bit in the queue head's overlay status field. When the *Active* bit transitions from one to zero, the transfer in the overlay is considered complete. The reason for the transfer completion (clearing the *Active* bit) determines the next state.

- *CErr* field decrements to zero. When this occurs the *Halted* bit is set to one and *Active* is set to zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The device responds to the transaction with a STALL PID. When this occurs, the *Halted* bit is set to one and the *Active* bit is set to zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The *Total Bytes to Transfer* field is zero after the transaction completes.
  - For a zero length transaction, it was zero before the transaction was started. When this condition occurs, the *Active* bit is set to zero.
- The PID code is an IN, and the number of bytes moved during the transaction is less than the *Maximum Packet Length*. When this occurs, the *Active* bit is set to zero and a short packet condition exists. The short-packet condition is detected during the Advance Queue state. Refer to Split Transactions for additional rules for managing low- and full-speed transactions.

With the exception of a NAK response (when *RL* field is zero), the host controller always writes the results of the transaction back to the overlay area in main memory. This includes when the transfer completes. For a high-speed endpoint, the queue head information written back includes minimally the following fields: The *PID Code* field indicates an IN and the device sends more than the expected number of bytes (for example *Maximum Packet Length* or *Total Bytes to Transfer* bytes, whichever is less) (for example a packet babble). This results in the host controller setting the *Halted* bit to one.

- NakCnt, dt, Total Bytes to Transfer, C_Page, Status, CERR, and Current Offset

For a low- or full-speed device the queue head information written back also includes the fields:

- C-prog-mask, FrameTag and S-bytes.

The duration of this state depends on the time it takes to complete the transaction(s) and the status write to the overlay is committed.

54.5.3.10.3.4 Halting a Queue Head

A halted endpoint is defined only for the transfer types that are managed through queue heads (control, bulk and interrupt).

The following events indicate that the endpoint has reached a condition where no more activity can occur without intervention from the driver:

- An endpoint may return a STALL handshake during a transaction,
- A transaction had three consecutive error conditions, or
- A Packet Babble error occurs on the endpoint.

When any of these events occur (for a queue head) the Host Controller halts the queue head and set the USBERRINT status bit in the USB_n_USBSTS register to one. To halt the queue head, the *Active* bit is set to zero and the *Halted* bit is set to one. There may be other error status bits that are set when a queue is halted. The host controller always writes back the overlay area to the source qTD when the transfer is complete, regardless of the reason (normal completion, short packet or halt). The host controller does not advance the transfer state on a transaction that results in a *Halt* condition (for example no updates necessary for *Total Bytes to Transfer*, *C_Page*, *Current Offset,* and *dt*). The host controller must update *CErr* as appropriate. When a queue head is halted, the *USB Error Interrupt* bit in the USB_n_USBSTS register is set to one. If the *USB Error Interrupt Enable* bit in the USB_n_USBINTR register is set to one, a hardware interrupt is generated at the next interrupt threshold.

54.5.3.10.3.5 Asynchronous Schedule Park Mode

Asynchronous Schedule Park mode is a special execution mode that can be enabled by system software, where the host controller is permitted to execute more than one bus transaction from a high-speed queue head in the Asynchronous schedule before continuing horizontal traversal of the Asynchronous schedule.

This feature has no effect on queue heads or other data structures in the Periodic schedule. This feature is similar in intent as the *Mult* feature that is used in the Periodic schedule. Where-as the *Mult* feature is a characteristic that is tunable for each endpoint; park-mode is a policy that is applied to all high-speed queue heads in the asynchronous schedule. It is essentially the specification of an iterator for consecutive bus transactions to the same endpoint. All of the rules for managing bus transactions and the results of those as defined in Execute Transaction apply. This feature merely specifies how many consecutive times the host controller is permitted to execute from the same queue head before moving to the next queue head in the Asynchronous List. This feature should allow the host controller to attain better bus utilization for those devices that are capable of moving data at maximum rate, while at the same time providing a fair service to all endpoints.

A host controller exports its capability to support this feature to system software by setting the *Asynchronous Schedule Park Capability* bit in the USB_n_HCCPARAMs register to one. This information keys system software that the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USB_n_USBCMD register are modifiable. System software enables the feature by writing a one to the *Asynchronous Schedule Park Mode Enable* bit.

When park-mode is not enabled (for example *Asynchronous Schedule Park Mode Enable* bit in the USB_n_USBCMD register is zero), the host controller must not execute more than one bus transaction per high-speed queue head, per traversal of the asynchronous schedule. When park-mode is enabled, the host controller must not apply the feature to a queue head whose *EPS* field indicates a Low/Full-speed device (for example only one bus transaction is allowed from each Low/Full-speed queue head per traversal of the asynchronous schedule). Park-mode may only be applied to queue heads in the Asynchronous schedule whose *EPS* field indicates that it is a high-speed device.

The host controller must apply park mode to queue heads whose *EPS* field indicates a high-speed endpoint. The maximum number of consecutive bus transactions a host controller may execute on a high-speed queue head is determined by the value in the *Asynchronous Schedule Park Mode Count* field in the USB_n_USBCMD register. Software must not set *Asynchronous Schedule Park Mode Enable* bit to one and also set *Asynchronous Schedule Park Mode Count* field to zero. The resulting behavior is not defined. An example behavioral example describes the operational requirements for the host controller implementing park-mode. This feature does not affect how the host controller handles the bus transaction as defined in Execute Transaction . It only effects how many consecutive bus transactions for the current queue head can be executed. All boundary conditions, error detection and reporting applies as usual. This feature is similar in concept to the use of the *Mult* field for high-bandwidth Interrupt for queue heads in the Periodic Schedule.

The host controller effectively loads an internal down-counter *PM-Count* from *Asynchronous Schedule Park Mode Count* when *Asyncrhonous Schedule Park Mode Enable* bit is one, and a high-speed queue head is first fetched and meets all the criteria for executing a bus transaction. After the bus transaction, *PM-Count* is decremented. The host controller may continue to execute bus transactions from the current queue head until *PM-Count* goes to zero, an error is detected, the buffer for the current transfer is exhausted or the endpoint responds with a flow-control or STALL handshake.

The following table summarizes the responses that effect whether the host controller continues with another bus transaction for the current queue head.

Table 54-45. Actions for Park Mode, based on Endpoint Response and Residual Transfer State

PID	Endpoint Response	Transfer State after Transaction		Action
		PM-Count	Bytes to Transfer	
IN	DATA[0,1] w/Maximum Packet sized data	Not zero	Not Zero	Allowed to perform another bus transaction.[1, 2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH.

Table continues on the next page...

Table 54-45. Actions for Park Mode, based on Endpoint Response and Residual Transfer State (continued)

	DATA[0,1] w/short packet	Don't care	Don't care	Retire qTD and move to next QH.
	NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH.
OUT	ACK	Not zero	Not Zero	Allowed to perform another bus transaction.[2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't' care	Move to next QH.
	NYET, NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH
PING	ACK	Not Zero	Not Zero	Allowed to perform another bus transaction. [2]
	NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH

1. The host controller may continue to execute bus transactions from the current high-speed queue head (if *PM-Count* is not equal to zero), if a PID mismatch is detected (for example expected DATA1 and received DATA0, or visa-versa).
2. This specification does not *require* that the host controller execute another bus transaction when *PM-Count* is non-zero. Implementations are encouraged to make appropriate complexity and performance trade-offs.

54.5.3.10.4 Write Back qTD

This state is entered from the Execute Transaction state when the *Active* bit is set to zero.

The source data for the write-back is the transfer results area of the queue head overlay area (see Table 54-45).

The host controller uses the *Current qTD Pointer* field as the target address for the qTD.

The queue head transfer result area is written back to the transfer result area of the target qTD. This state is also referred to as: qTD retirement. The fields that must be written back to the source qTD include *Total Bytes to Transfer*, *Cerr*, and *Status*.

The duration of this state depends on when the qTD write-back is committed.

54.5.3.10.5 Follow Queue Head Horizontal Pointer

The host controller must use the horizontal pointer in the queue head to the next schedule data structure when any of the following conditions exist:

- If the *Active* bit is one on exit from the Execute Transaction state, or
- When the host controller exits the Write Back qTD state, or
- If the Advance Queue state fails to advance the queue because the target qTD is not active, or
- If the *Halted* bit is one on exit from the Fetch QH state.

There is no functional requirement that the host controller wait until the current transaction is complete before using the horizontal pointer to read the next linked data structure. However, it must wait until the current transaction is complete before executing the next data structure.

54.5.3.10.6 Buffer Pointer List Use for Data Streaming with qTDs

A qTD has an array of buffer pointers, which is used to reference the data buffer for a transfer. This specification requires that the buffer associated with the transfer be *virtually contiguous*.

This means: if the buffer spans more than one physical page, it must obey the following rules (the figure below illustrates an example):

- The first portion of the buffer must begin at some offset in a page and extend through the end of the page.
- The remaining buffer cannot be allocated in small chunks scattered around memory. For each 4 K chunk beyond the first page, each buffer portion matches to a full 4 K page. The final portion, which may only be large enough to occupy a portion of a page, must start at the top of the page and be contiguous within that page.

The buffer pointer list in the qTD is long enough to support a maximum transfer size of 20 K bytes. This case occurs when all five buffer pointers are used and the first offset is zero. A qTD handles a 16 Kbyte buffer with any starting buffer alignment.

The host controller uses the field *C_Page* field as an index value to determine which buffer pointer in the list should be used to start the current transaction. The host controller uses a different buffer pointer for each physical page of the buffer. This is always true, even if the buffer is physically contiguous.

The host controller must detect when the current transaction spans a page boundary and automatically move to the next available buffer pointer in the page pointer list. The next available pointer is reached by incrementing *C_Page* and pulling the next page pointer from the list. Software must ensure there are sufficient buffer pointers to move the amount of data specified in the *Bytes to Transfer* field.

The following figure illustrates a nominal example of how System software would initialize the buffer pointers list and the *C_Page* field for a transfer size of 16383 bytes. *C_Page* is set to zero. The upper 20-bits of Page 0 references the start of the physical page. *Current Offset* (the lower 12-bits of queue head Dword 7) holds the offset in the page for example 2049 (for example 4096-2047). The remaining page pointers are set to reference the beginning of each subsequent 4 K page.

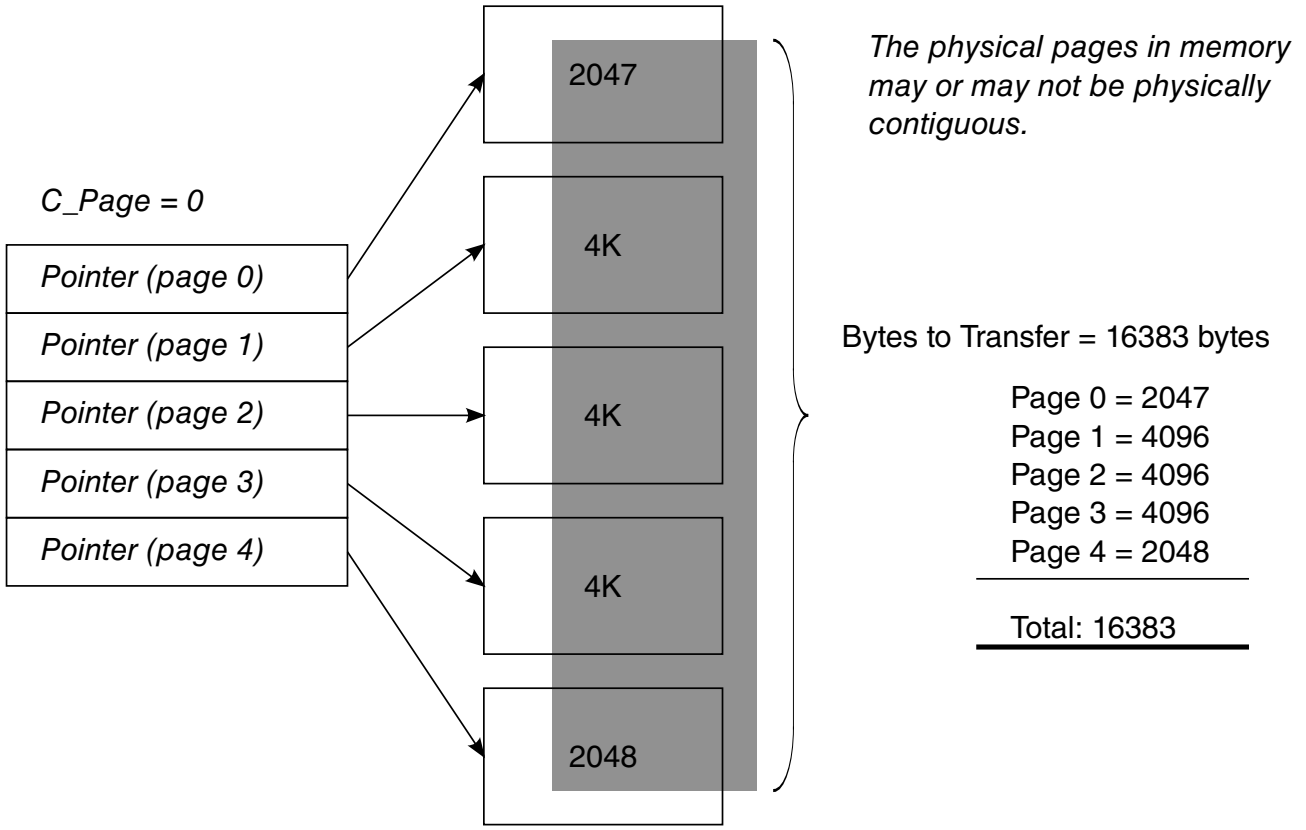


Figure 54-18. Example Mapping of qTD Buffer Pointers to Buffer Pages

For the first transaction on the qTD (assuming a 512-byte transaction), the host controller uses the first buffer pointer (page 0 because *C_Page* is set to zero) and concatenates the *Current Offset* field. The 512 bytes are moved during the transaction, the *Current Offset* and *Total Bytes to Transfer* are adjusted by 512 and written back to the queue head working area.

During the 4th transaction, the host controller needs 511 bytes in page 0 and one byte in page 1. The host controller increments *C_Page* (to 1) and use the page 1 pointer to move the final byte of the transaction. After the 4th transaction, the active page pointer is the page 1 pointer and *Current Offset* has rolled to one, and both are written back to the overlay area. The transactions continue for the rest of the buffer, with the host controller automatically moving to the next page pointer (that is *C_Page*) when necessary. The three conditions for how the host controller handles *C_Page*:

- The current transaction does not span a page boundary. The value of *C_Page* is not adjusted by the host controller.

- The current transaction does span a page boundary. The host controller must detect the page cross condition and advance to the next buffer while streaming data to/from the USB.
- The current transaction completes on a page boundary (that is the last byte moved for the current transaction is the last byte in the page for the current page pointer). The host controller must increment *C_Page* before writing back status for the transaction.

NOTE

The only valid adjustment the host controller may make to *C_Page* is to increment by one.

54.5.3.10.7 Adding Interrupt Queue Heads to the Periodic Schedule

The link path(s) from the periodic frame list to a queue head establishes in which frames a transaction can be executed for the queue head. Queue heads are linked into the periodic schedule so they are polled at the appropriate rate.

System software sets a bit in a queue head's *S-Mask* to indicate which micro-frame within 1 msec period a transaction should be executed for the queue head. Software must ensure that all queue heads in the periodic schedule have *S-Mask* set to a non-zero value. An *S-mask* with zero value in the context of the periodic schedule yields undefined results.

If the desired poll rate is greater than one frame, system software can use a combination of queue head linking and *S-Mask* values to spread interrupts of equal poll rates through the schedule so that the periodic bandwidth is allocated and managed in the most efficient manner possible. Some examples are illustrated in the following table.

Table 54-46. Example Periodic Reference Patterns for Interrupt Transfers with 2ms Poll Rate

Frame # Reference Sequence	Description
0, 2, 4, 6, 8, and so on *S-Mask* = 01h	A queue head for the *bInterval* of 2 msec (16 micro-frames) is linked into the periodic schedule so that it is reachable from the periodic frame list locations indicated in the previous column. In addition, the *S-Mask* field in the queue head is set to 01h, indicating that the transaction for the endpoint should be executed on the bus during micro-frame 0 of the frame.
0, 2, 4, 6, 8, and so on *S-Mask* = 02h	Another example of a queue head with a *bInterval* of 2 msec is linked into the periodic frame list at exactly the same interval as the previous example. However, the *S-Mask* is set to 02h indicating that the transaction for the endpoint should be executed on the bus during micro-frame 1 of the frame.

54.5.3.10.8 Managing Transfer Complete Interrupts from Queue Heads

The host controller sets an interrupt to be signaled at the next interrupt threshold when the completed transfer (qTD) has an *Interrupt on Complete* (*IOC*) bit set to one, or whenever a transfer (qTD) completes with a short packet.

If system software needs multiple qTDs to complete a client request (that is like a control transfer) the intermediate qTDs do not require interrupts. System software may only need a single interrupt to notify it that the complete buffer has been transferred. System software may set IOC's to occur more frequently. A motivation for this may be that it wants early notification so that interface data structures can be re-used in a timely manner.

54.5.3.11 Ping Control

USB 2.0 defines an addition to the protocol for high-speed devices called Ping. Ping is required for all USB 2.0 High-speed bulk and control endpoints.

Ping is not allowed for a split-transaction stream. This extension to the protocol eliminates the bad side-effects of Naking OUT endpoints. The *Status* field has a *Ping State* bit, which the host controller uses to determine the *next* actual PID it uses in the next transaction to the endpoint (see the table below).

The Ping State bit is only managed by the host controller for queue heads that meet the following criteria:

- Queue head is not an interrupt and
- *EPS* field equals High-Speed and
- *PIDCode* field equals OUT

The following table illustrates the state transition table for the host controller's responsibility for maintaining the PING protocol. Refer to Chapter 8 in the USB Specification Revision 2.0 for detailed description on the Ping protocol.

Table 54-47. Ping Control State Transition Table

Event			
Current	Host	Device	Next
Do Ping	PING	Nak	Do Ping
Do Ping	PING	Ack	Do OUT
Do Ping	PING	XactErr [1]	Do Ping
Do Ping	PING	Stall	N/C [2] Do
OUT	OUT	Nak	Do Ping
Do OUT	OUT	Nyet	Do Ping

Table continues on the next page...

Table 54-47. Ping Control State Transition Table (continued)

Do OUT	OUT	Ack	Do OUT
Do OUT	OUT	XactErr[1]	Do Ping
Do OUT	OUT	Stall	N/C [2]

1. Transaction Error (XactErr) is any time the host misses the handshake.
2. No transition change required for the Ping State bit. The Stall handshake results in the endpoint being halted (for example Active set to zero and Halt set to one). Software intervention is required to restart queue. 3 A Nyet response to an OUT means that the device has accepted the data, but cannot receive any more at this time. Host must advance the transfer state and additionally, transition the Ping State bit to Do Ping. The Ping State bit has the following encoding:

Table 54-48. Ping State bit Encoding

Value	Meaning
0B	Do OUT The host controller uses an OUT PID during the next bus transaction to this endpoint.
1B	Do Ping The host controller uses a PING PID during the next bus transaction to this endpoint.

The defined ping protocol (see USB 2.0 Specification, Chapter 8) allows the host to be *imprecise* on the initialization of the ping protocol (that is start in *Do OUT* when we don't know whether there is space on the device or not). The host controller manages the *Ping State* bit. System software sets the initial value in the queue head when it initializes a queue head. The host controller preserves the *Ping State* bit across all queue advancements. This means that when a new qTD is written into the queue head overlay area, the previous value of the *Ping State* bit is preserved.

54.5.3.12 Split Transactions

USB 2.0 defines extensions to the bus protocol for managing USB 1.x data streams through USB 2.0 Hubs.

This section describes how the host controller uses the interface data structures to manage data streams with full- and low-speed devices, connected below USB 2.0 hub, utilizing the split transaction protocol.

Refer to USB 2.0 Specification for the complete definition of the split transaction protocol. Full- and Low-speed devices are enumerated identically as high-speed devices, but the transactions to the Full- and Low-speed endpoints use the split-transaction protocol on the high-speed bus. The split transaction protocol is an encapsulation of (or wrapper around) the Full- or Low-speed transaction. The high-speed wrapper portion of the protocol is addressed to the USB 2.0 Hub and Transaction Translator below which the Full- or Low-speed device is attached.

The EHCI interface uses dedicated data structures for managing full-speed isochronous data streams (see Split Transaction Isochronous Transfer Descriptor (siTD)). Control, Bulk and Interrupt are managed using the queuing data structures (see Queue Head). The interface data structures need to be programmed with the device address and the Transaction Translator number of the USB 2.0 Hub operating as the Low-/Full-speed host controller for this link. The following sections describe the details of how the host controller must process and manage the split transaction protocol.

54.5.3.12.1 Split Transactions for Asynchronous Transfers

A queue head in the asynchronous schedule with an *EPS* field indicating a full-or low-speed device indicates to the host controller that it must use split transactions to stream data for this queue head.

All full-speed bulk and full-, low-speed control are managed through queue heads in the asynchronous schedule.

Software must initialize the queue head with the appropriate device address and port number for the transaction translator that is serving as the full/low-speed host controller for the links connecting the endpoint. Software must also initialize the split transaction state bit (*SplitXState*) to Do-Start-Split. Finally, if the endpoint is a control endpoint, then system software must set the *Control Transfer Type* (*C*) bit in the queue head to one. If this is not a control transfer type endpoint, the *C* bit must be initialized by software to be zero. This information is used by the host controller to properly set the Endpoint Type (ET) field in the split transaction bus token. When the *C* bit is zero, the split transaction token's ET field is set to indicate a bulk endpoint. When the *C* bit is one, the split transaction token's ET field is set to indicate a control endpoint. Refer to Chapter 8 of USB Specification Revision 2.0 for details.

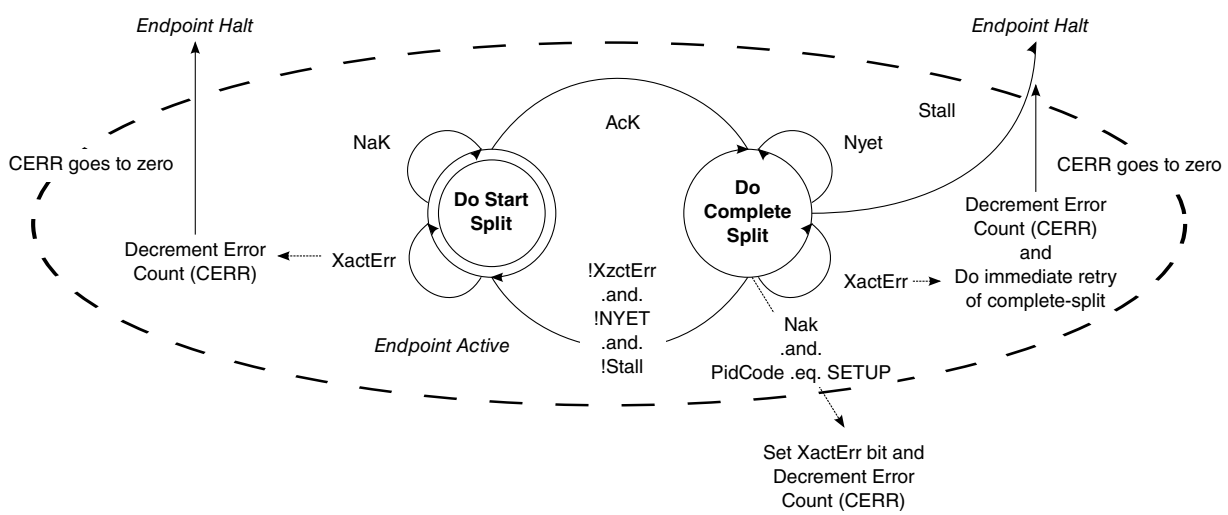


Figure 54-19. Host Controller Asynchronous Schedule Split-Transaction State Machine

54.5.3.12.1.1 Asynchronous - Do Start Split

This is the state which software must initialize a full- or low-speed asynchronous queue head. This state is entered from the Do Complete Split state only after a complete-split transaction receives a valid response from the transaction translator that is not a Nyet handshake.

For queue heads in this state, the host controller executes a start-split transaction to the appropriate transaction translator. If the bus transaction completes without an error and *PidCode* indicates an IN or OUT transaction, then the host controller reloads the error counter (*CErr*). If it is a successful bus transaction and the *PidCode* indicates a SETUP, the host controller does not reload the error counter. If the transaction translator responds with a Nak, the queue head is left in this state, and the host controller proceeds to the next queue head in the asynchronous schedule.

If the host controller times out the transaction (no response, or bad response) the host controller decrements *Cerr* and proceeds to the next queue head in the asynchronous schedule.

54.5.3.12.1.2 Asynchronous - Do Complete Split

This state is entered from the Do Start Split state only after a start-split transaction receives an Ack handshake from the transaction translator.

For queue heads in this state, the host controller executes a complete-split transaction to the appropriate transaction translator. If the transaction translator responds with a Nyet handshake, the queue head is left in this state, the error counter is reset and the host controller proceeds to the next queue head in the asynchronous schedule. When a Nyet handshake is received for a bus transaction where the queue head's *PidCode* indicates an IN or OUT, the host controller reloads the error counter (*CErr*). When a Nyet handshake is received for a complete-split bus transaction where the queue head's *PidCode* indicates a SETUP, the host controller must not adjust the value of *CErr*.

Independent of *PIDCode*, the following responses have the effects:

- Transaction Error (XactErr). Timeout or data CRC failure, and so on. The error counter (*Cerr*) is decremented by one and the complete split transaction is *immediately* retried (if possible). If there is not enough time in the micro-frame to execute the retry, the host controller MUST ensure that the next time the host controller begins executing from the Asynchronous schedule, it must begin executing from this queue head. If another start-split (for some other endpoint) is sent to the transaction translator before the complete-split is really completed, the transaction translator could dump the results (which were never delivered to the host). This is why the core specification states the retries must be immediate. A method to

accomplish this behavior is to not advance the asynchronous schedule. When the host controller returns to the asynchronous schedule in the next micro-frame, the first transaction from the schedule is the retry for this endpoint.

If *Cerr* went to zero, the host controller must halt the queue.

- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced and the state is exited. If the *PidCode* is a SETUP, then the Nak response is a protocol error. The *XactErr* status bit is set to one and the *CErr* field is decremented.
- STALL. The target endpoint responded with a STALL handshake. The host controller sets the *halt* bit in the status byte, retires the qTD but does not attempt to advance the queue.

If the *PidCode* indicates an IN, then any of following responses are expected:

- DATA0/1. On reception of data, the host controller ensures the PID matches the expected data toggle and checks CRC. If the packet is *good*, the host controller advances the state of the transfer, for example move the data pointer by the number of bytes received, decrement *BytesToTransfer* field by the number of bytes received, and toggle the *dt* bit. The host controller then exit this state. The response and advancement of transfer may trigger other processing events, such as retirement of the qTD and advancement of the queue.

If the data sequence PID does not match the expected, the data is ignored, the transfer state is not advanced and this state is exited. If the *PidCode* indicates an OUT/SETUP, then any of following responses are expected:

- ACK. The target endpoint accepted the data, so the host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount and the data toggle bit (*dt*) is toggled. The host controller then exit this state.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Managing Control/Bulk/Interrupt Transfers through Queue Heads).

54.5.3.12.2 Split Transaction Interrupt

Split-transaction Interrupt-IN/OUT endpoints are managed through the same data structures used for high-speed interrupt endpoints. They both co-exist in the periodic schedule.

Queue heads/qTDs offer the set of features required for reliable data delivery, which is characteristic to interrupt transfer types. The split-transaction protocol is managed completely within this defined functional transfer framework. For example, for a high-speed endpoint, the host controller visits a queue head, execute a high-speed transaction (if criteria are met) and advance the transfer state (or not) depending on the results of the entire transaction. For low- and full-speed endpoints, the details of the *execution* phase are different (that is takes more than one bus transaction to complete), but the remainder of the operational framework is intact. This means that the transfer advancement, and so on, occurs as defined in Managing Control/Bulk/Interrupt Transfers through Queue Heads, but only occurs on the completion of a split transaction.

54.5.3.12.2.1 Split Transaction Scheduling Mechanisms for Interrupt

Full- and low-speed Interrupt queue heads have an *EPS* field indicating full- or low-speed and have a non-zero *S-mask* field.

The host controller can detect this combination of parameters and assume the endpoint is a periodic endpoint. Low- and full-speed interrupt queue heads require the use of the split transaction protocol. The host controller sets the Endpoint Type (ET) field in the split token to indicate the transaction is an interrupt. These transactions are managed through a transaction translator's periodic pipeline. Software should not set these fields to indicate the queue head is an interrupt unless the queue head is used in the periodic schedule.

System software manages the per/transaction translator periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each endpoint occurs. The characteristics of the transaction translator are such that the high-speed transaction protocol must execute during explicit micro-frames, or the data or response information in the pipeline is lost.

The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule and queue head data structure. The S and ^{C}X labels indicate micro-frames where software can schedule start-splits and complete splits (respectively).

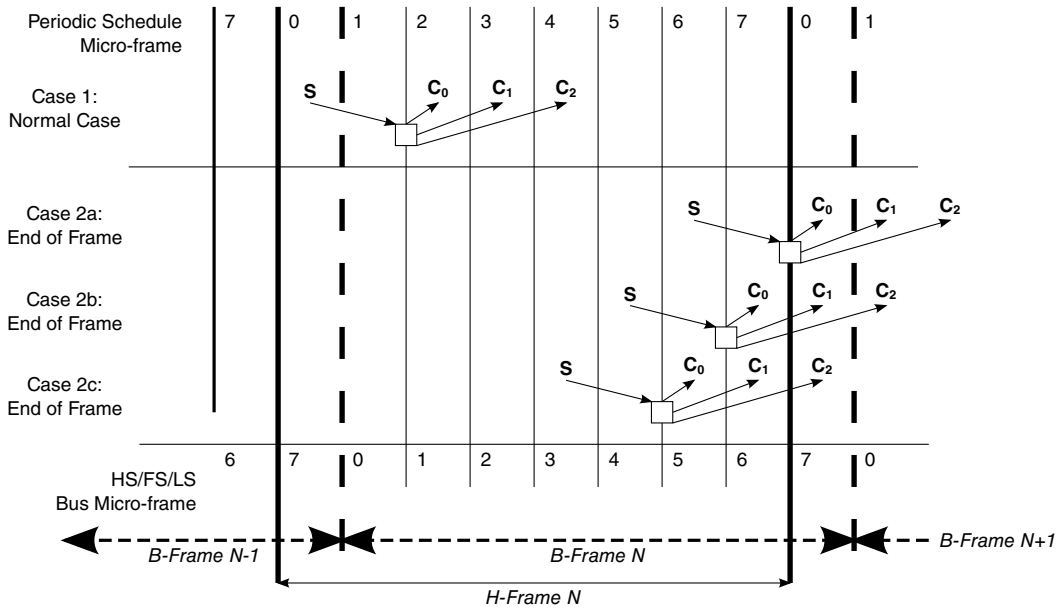


Figure 54-20. Split Transaction, Interrupt Scheduling Boundary Conditions

The scheduling cases are:

- Case 1: The normal scheduling case is where the entire split transaction is completely bounded by a frame (*H-Frame* in this case).
- Case 2a through Case 2c: The USB 2.0 Hub pipeline rules states clearly, when and how many complete-splits must be scheduled to account for earliest to latest execution on the full/low-speed link. The complete-splits may span the *H-Frame* boundary when the start-split is in micro-frame 4 or later. When this occurs, the *H-Frame* to *B-Frame* alignment requires that the queue head be reachable from consecutive periodic frame list locations. System software cannot build an efficient schedule that satisfies this requirement unless it uses FSTNs.

The figure below illustrates the general layout of the periodic schedule.

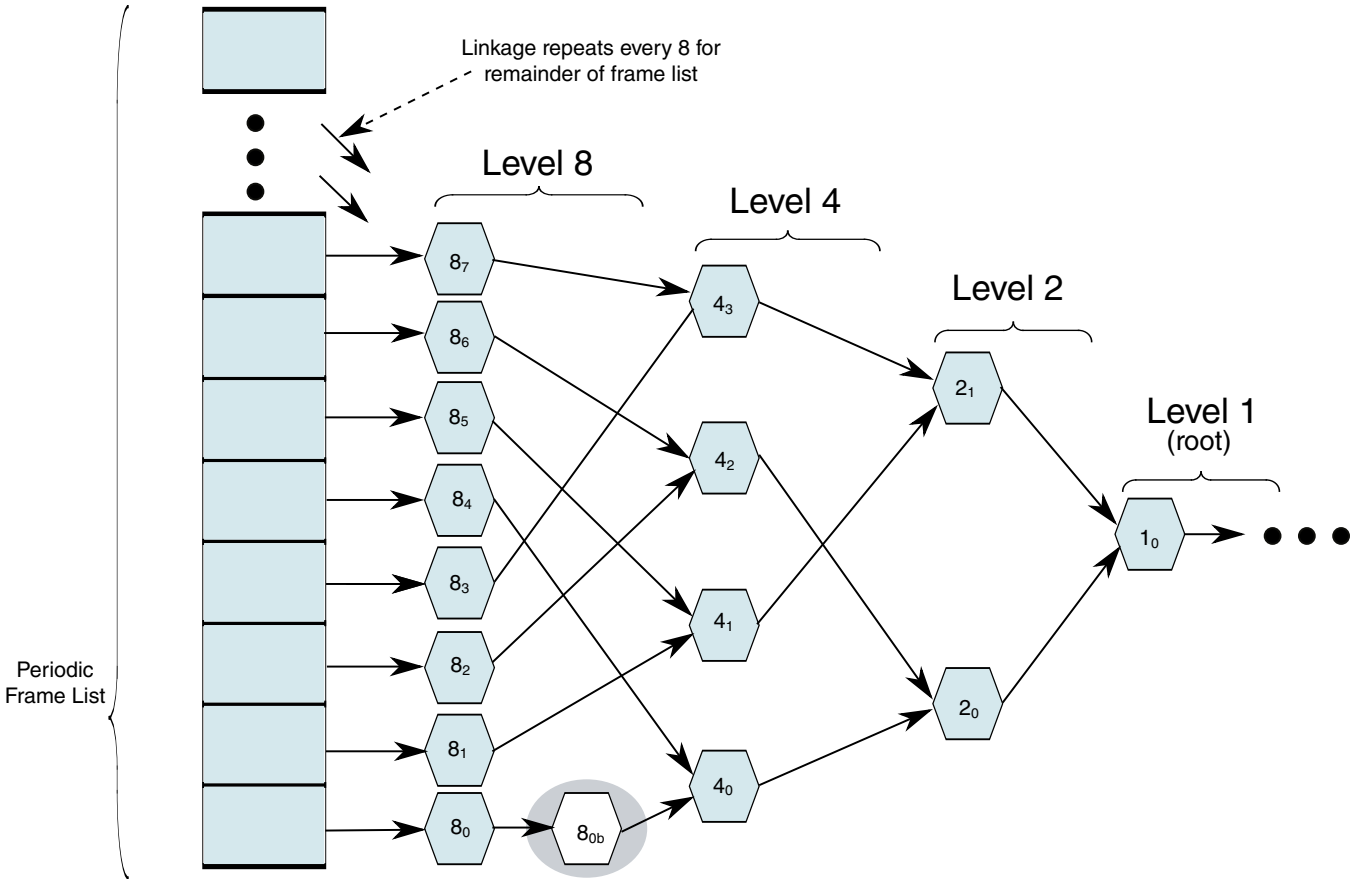


Figure 54-21. General Structure of EHCI Periodic Schedule Utilizing Interrupt Spreading

The periodic frame list is effectively the leaf level a binary tree, which is always traversed leaf to root. Each level in the tree corresponds to a 2^N poll rate. Software can efficiently manage periodic bandwidth on the USB by *spreading* interrupt queue heads that have the same poll rate requirement across all the available paths from the frame list. For example, system software can schedule eight poll rate 8 queue heads and account for them once in the high-speed bus bandwidth allocation.

When an endpoint is allocated an execution footprint that spans a frame boundary, the queue head for the endpoint must be reachable from consecutive locations in the frame list. An example would be if 8_{0b} where such an endpoint. Without additional support on the interface, to get 8_{0b} reachable at the correct time, software would have to link 8_1 to 8_{0b}. It would then have to move 4_1 and everything linked after into the same path as 4_0. This upsets the integrity of the binary tree and disallows the use of the spreading technique.

FSTN data structures are used to preserve the integrity of the binary-tree structure and enable the use of the spreading technique. Host Controller Operational Model for FSTNs defines the hardware and software operational model requirements for using FSTNs.

The following queue head fields are initialized by system software to instruct the host controller when to execute portions of the split-transaction protocol:

- *SplitXState*. This is single bit residing in the *Status* field of a queue head (see Table 54-25). This bit is used to track the current state of the split transaction.
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 54-20, case one, the *S-mask* would have a value of 00000001b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do_Start, and the current micro-frame as indicated by FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask.* This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 54-20, case one, the *C-mask* would have a value of 00011100b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do_Complete, and the current micro-frame as indicated by FRINDEX[2:0] is 2, 3, or 4, then execute a complete-split transaction. It is software's responsibility to ensure that the translation between *H-Frames* and *B-Frames* is correctly performed when setting bits in *S-mask* and *C-mask*

54.5.3.12.2.2 Host Controller Operational Model for FSTNs

The FSTN data structure is used to manage Low/Full-speed interrupt queue heads that need to be reached from consecutive frame list locations (that is boundary cases 2a through 2c).

An FSTN is essentially a *back pointer*, similar in intent to the back pointer field in the siTD data structure (see siTD Back Link Pointer).

This feature provides software a simple primitive to save a schedule position, redirect the host controller to traverse the necessary queue heads in the previous frame, then restore the original schedule position and complete normal traversal.

The four components to the use of FSTNs:

- FSTN data structure.
- A *Save Place* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to zero.

- A *Restore* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to one.
- Host controller FSTN traversal rules.

When the host controller encounters an FSTN during micro-frames 2 through 7 it simply follows the node's *Normal Path Link Pointer* to access the next schedule data structure.

NOTE

The FSTN's *Normal Path Link Pointer.T-bit* may set to one, which the host controller must interpret as the end of periodic list mark.

When the host controller encounters a *Save-Place* FSTN in micro-frames 0 or 1, it saves the value of the *Normal Path Link Pointer* and set an internal flag indicating that it is executing in *Recovery Path* mode. *Recovery Path* mode modifies the host controller's rules for how it traverses the schedule and limits which data structures is considered for execution of bus transactions. The host controller continues executing in *Recovery Path* mode until it encounters a *Restore* FSTN or it determines that it has reached the end of the micro-frame (see details in the list below).

The rules for schedule traversal and limited execution while in *Recovery Path* mode are:

- Always follow the *Normal Path Link Pointer* when it encounters an FSTN that is a *Save-Place* indicator. The host controller must not recursively follow *Save-Place* FSTNs. Therefore, while executing in *Recovery Path* mode, it must never follow an FSTN's *Back Path Link Pointer*.
- Do not process an siTD or, iTD data structure. Simply follow its *Next Link Pointer*.
- Do not process a QH (Queue Head) whose *EPS* field indicates a high-speed device. Simply follow its *Horizontal Link Pointer*.
- When a QH's *EPS* field indicates a Full/Low-speed device, the host controller considers only it for execution if its *SplitXState* is DoComplete (note: this applies whether the *PID Code* indicates an IN or an OUT). See Execute Transaction and Tracking Split Transaction Progress for Interrupt Transfers for a complete list of additional conditions that must be met in general for the host controller to issue a bus transaction.
  - The host controller must not execute a Start-split transaction while executing in *Recovery Path* mode. See Periodic Isochronous - Do Complete Split for special handling when in *Recovery Path* mode.
- Stop traversing the *recovery path* when it encounters an FSTN that is a *Restore* indicator. The host controller unconditionally uses the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* when returning to the normal path traversal. The

host controller must clear the context of executing a *Recovery Path* when it restores schedule traversal to the *Save-Place* FSTN's *Normal Path Link Pointer*.

- If the host controller determines that there is not enough time left in the micro-frame to complete processing of the periodic schedule, it abandons traversal of the recovery path, and clears the context of executing a recovery path. The result is that at the start of the next consecutive micro-frame, the host controller starts traversal at the frame list.

An example traversal of a periodic schedule that includes FSTNs is illustrated in the following figure.

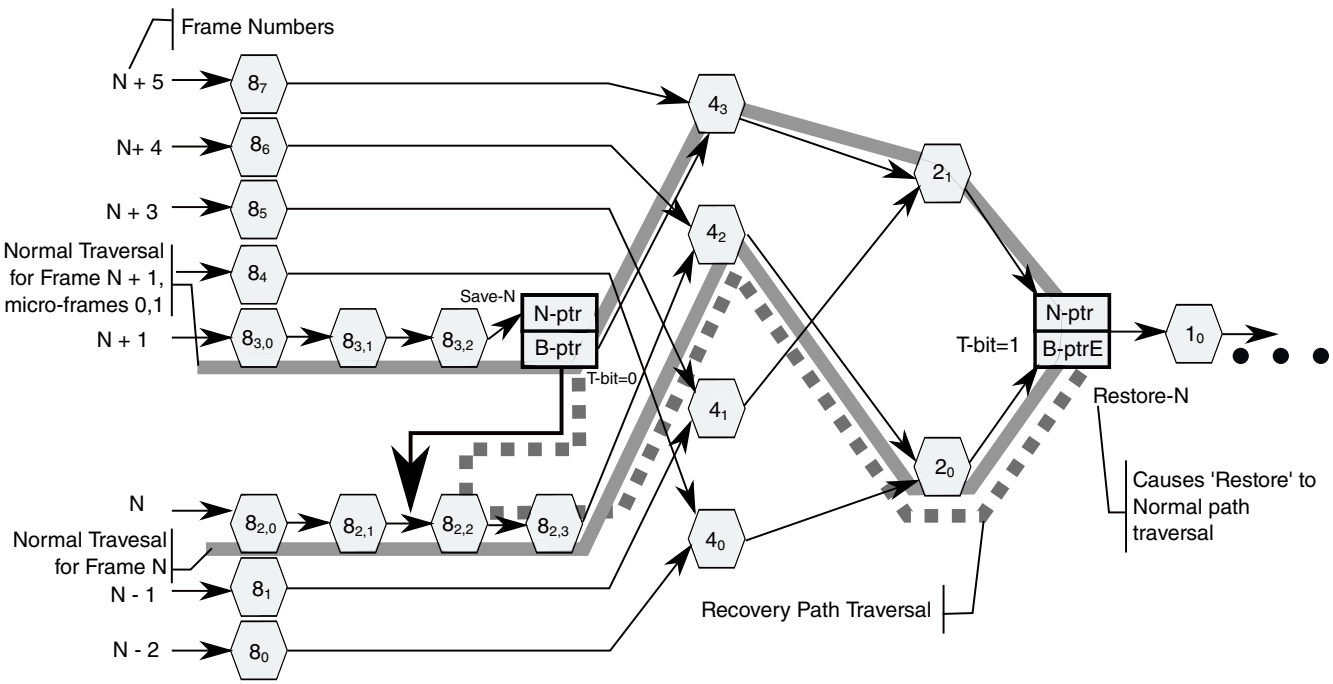


Figure 54-22. Example Host Controller Traversal of Recovery Path via FSTNs

In frame N+1 (micro-frames 0 and 1), when the host controller encounters Save-Path FSTN (Save-N), it observes that Save-N.Back Path Link Pointer.T-bit is zero (definition of a Save-Path indicator). The host controller saves the value of Save-N.Normal Path Link Pointer and follows Save-N.Back Path Link Pointer. At the same time, it sets an internal flag indicating that it is now in *Recovery Path* mode (the recovery path is annotated in the figure above with a large dashed line). The host controller continues traversing data structures on the recovery path and executing only those bus transactions as noted above, on the recovery path until it reaches Restore FSTN (Restore-N). Restore-N.Back Path Link Pointer.T-bit is set to one (definition of a Restore indicator), so the host controller exits *Recovery Path* mode by clearing the internal *Recovery Path* mode flag and commences (restores) schedule traversal using the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* (for example Save-N.Normal Path Link Pointer). The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, $8_{2.2}$, $8_{2.3}$, 4_2,

2_0, Restore-N, 4_3, 2_1, Restore-N, 1_0 …}. The nodes on the recovery-path are in bold. In frame N (micro-frames 0-7), for this example, the host controller traverses all of the schedule data structures utilizing the *Normal Path Link Pointers* in any FSTNs it encounters. This is because the host controller has not yet encountered a *Save-Place* FSTN so it not executing in *Recovery Path* mode. When it encounters the *Restore* FSTN, (Restore-N), during micro-frames 0 and 1, it uses Restore-N.Normal Path Link Pointer to traverse to the next data structure (that is normal schedule traversal). This is because the host controller must use a Restore FSTN's *Normal Path Link Pointer* when not executing in a *Recovery-Path* mode. The nodes traversed during frame N include: {$8_{2.0}$, $8_{2.1}$, $8_{2.2}$, $8_{2.3}$, 4_2, 2_0, Restore-N, 1_0 …}.

In frame N+1 (micro-frames 2-7), when the host controller encounters Save-Path FSTN Save-N, it unconditionally follows Save-N.Normal Path Link Pointer. The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, 4_3, 2_1, Restore-N, 1_0 …}.

54.5.3.12.2.3 Software Operational Model for FSTNs

Software must create a consistent, coherent schedule for the host controller to traverse.

When using FSTNs, system software must adhere to the following rules:

- Each *Save-Place* indicator requires a matching *Restore* indicator.
  - The *Save-Place* indicator is an FSTN with a valid *Back Path Link Pointer and T-bit* equal to zero.
    - *Back Path Link Pointer.Typ* field must be set to indicate the referenced data structure is a queue head. The *Restore* indicator is an FSTN with its *Back Path Link Pointer.T-bit* set to one.
  - A *Restore* FSTN may be matched to one or more *Save-Place* FSTNs. For example, if the schedule includes a poll-rate 1 level, then system software only needs to place a *Restore* FSTN at the beginning of this list in order to match all possible *Save-Place* FSTNs.
- If the schedule does not have elements linked at a poll-rate level of one, and one or more *Save-Place* FSTNs are used, then System Software must ensure the *Restore* FSTN's *Normal Path Link Pointer*'s *T-bit* is set to one, as this is used to mark the end of the periodic list.
- When the schedule does have elements linked at a poll rate level of one, a *Restore* FSTN must be the first data structure on the poll rate one list. All traversal paths from the frame list converge on the poll-rate one list. System software must ensure that *Recovery Path* mode is exited before the host controller is allowed to traverse the poll rate level one list.
- A *Save-Place* FSTN's *Back Path Link Pointer* must reference a queue head data structure. The referenced queue head must be reachable from the previous frame list

location. In other words, if the *Save-Place* FSTN is reachable from frame list offset N, then the FSTN's *Back Path Link Pointer* must reference a queue head that is reachable from frame list offset N-1.

Software should make the schedule as efficient as possible. What this means in this context is that software should have no more than one *Save-Place* FSTN reachable in any single frame. Note there is times when two (or more, depending on the implementation) could exist as full/low-speed footprints change with bandwidth adjustments. This could occur, for example when a bandwidth re-balance causes system software to move the *Save-Place* FSTN from one poll rate level to another. During the transition, software must preserve the integrity of the previous schedule until the new schedule is in place.

54.5.3.12.2.4 Tracking Split Transaction Progress for Interrupt Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where data is lost.

For interrupt-IN transfers, data is lost when it makes it into the USB 2.0 hub, but the USB 2.0 host system is unable to get it from the USB 2.0 Hub and into the system before it expires from the transaction translator pipeline.

When a lost data condition is detected, the queue must be halted, thus signaling system software to recover from the error. A data-loss condition exists whenever a start-split is issued, accepted and successfully executed by the USB 2.0 Hub, but the complete-splits get unrecoverable errors on the high-speed link, or the complete-splits do not occur at the correct times. One reason complete-splits might not occur at the right time would be due to host-induced system hold-offs that cause the host controller to miss bus transactions because it cannot get timely access to the schedule in system memory.

The same condition can occur for an interrupt-OUT, but the result is not an endpoint halt condition, but rather effects only the progress of the transfer. The queue head has the following fields to track the progress of each split transaction. These fields are used to keep incremental state about which (and when) portions have been executed.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets one of the *C-prog*-mask bits for each complete-split executed. The bit position is determined by the micro-frame number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a

complete-split transaction. If the previous complete-splits have not been executed then it means one (or more) have been skipped and data has potentially been lost.
- *FrameTag*. This field is used by the host controller during the complete-split portion of the split transaction to tag the queue head with the frame number (*H-Frame* number) when the next complete split must be executed.
- *S-bytes*. This field can be used to store the number of data payload bytes sent during the start-split (if the transaction was an OUT). The *S-bytes* field must be used to accumulate the data payload bytes received during the complete-splits (for an IN).

54.5.3.12.2.5 Split Transaction Execution State Machine for Interrupt

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame.*

As with asynchronous Full- and Low-speed endpoints, a split-transaction state machine is used to manage the split transaction sequence.

Aside from the fields defined in the queue head for scheduling and tracking the split transaction, the host controller calculates one internal mechanism that is also used to manage the split transaction. The internal calculated mechanism is:

- *cMicroFrameBit* is a single-bit encoding of the current micro-frame number. It is an eight-bit value calculated by the host controller at the beginning of every micro-frame. It is calculated from the three least significant bits of the *FRINDEX* register (that is, *cMicroFrameBit* = (1 shifted-left(*FRINDEX*[2:0])). The *cMicroFrameBit* has at most one bit asserted, which always corresponds to the current micro-frame number. For example, if the current micro-frame is 0, then *cMicroFrameBit* will equal 00000001b. The variable *cMicroFrameBit* is used to compare against the *S-mask* and *C-mask* fields to determine whether the queue head is marked for a start- or complete-splt transaction for the current micro-frame.

The following figure illustrates the state machine for managing a complete interrupt split transaction. There are two phases to each split transaction. The first is a single start-split transaction, which occurs when the *SplitXState* is at Do_Start and the single bit in *cMicroFrameBit* has a corresponding bit active in *QH.S-mask*. The transaction translator does not acknowledge the receipt of the periodic start-split, so the host controller unconditionally transitions the state to Do_Complete. Due to the available jitter in the transaction translator pipeline, there will be more than one complete-split transaction scheduled by software for the Do_Complete state. This translates simply to the fact that there are multiple bits set to a one in the *QH.C-mask* field.

The host controller keeps the queue head in the Do_Complete state until the split transaction is complete (see definition below), or an error condition triggers the *three-strikes-rule* (for example, after the host tries the same transaction three times, and each

encounters an error, the host controller will stop retrying the bus transaction and halt the endpoint, thus requiring system software to detect the condition and perform system-dependent recovery).

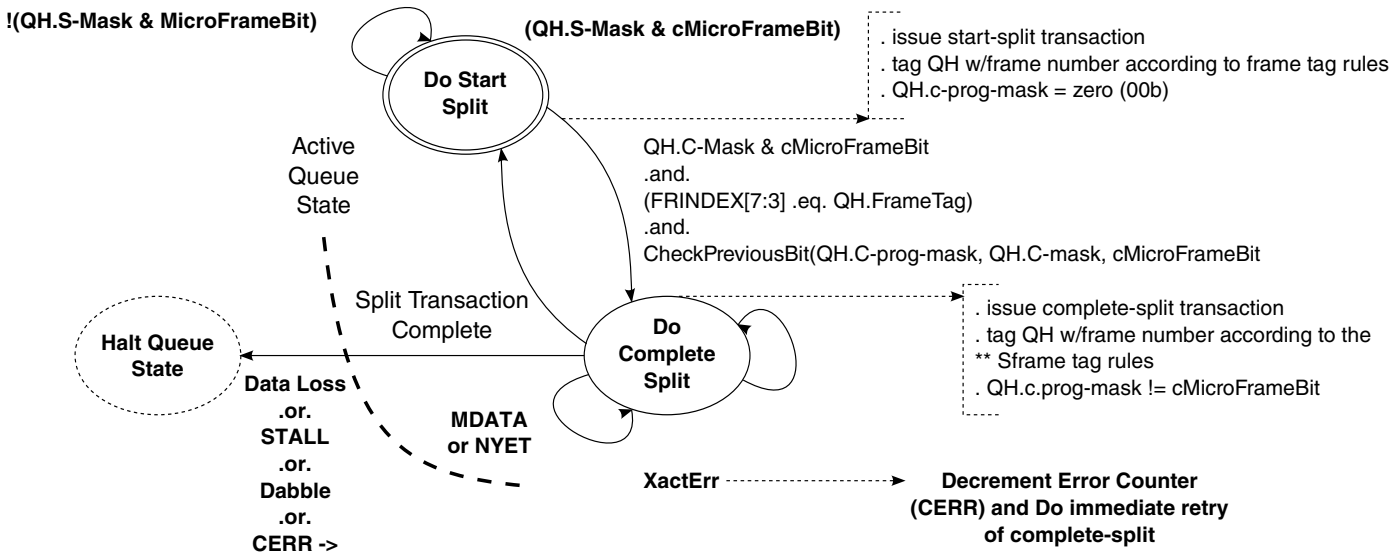


Figure 54-23. Split Transaction State Machine for Interrupt

See Previous Section for the frame tag management rules.

Periodic Interrupt - Do Start Split

This is the state software must initialize a full- or low-speed interrupt queue head *StartXState* bit. This state is entered from the Do_Complete Split state only after the split transaction is complete. This occurs when one of the following events occur: The transaction translator responds to a complete-split transaction with one of the following:

- NAK. A NAK response is a propagation of the full- or low-speed endpoint's NAK response.
- ACK. An ACK response is a propagation of the full- or low-speed endpoint's ACK response. Only occurs on an OUT endpoint.
- DATA 0/1. Only occurs for INs. Indicates that this is the last of the data from the endpoint for this split transaction.
- ERR. The transaction on the low-/full-speed link below the transaction translator had a failure (for example, timeout, bad CRC, etc.).
- NYET (and Last). The host controller issued the last complete-split and the transaction translator responded with a NYET handshake. This means that the start-split was not correctly received by the transaction translator, so it never executed a transaction to the full- or low-speed endpoint, see Section Periodic Isochronous - Do Complete Split for the definition of 'Last'.

Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it performs the following test to determine whether to execute a start-split.

- *QH.S-mask* is bit-wise anded with *cMicroFrameBit*.

If the result is non-zero, then the host controller will issue a start-split transaction. If the *PIDCode* field indicates an IN transaction, the host controller must zero-out the *QH.S-bytes* field. After the split-transaction has been executed, the host controller sets up state in the queue head to track the progress of the complete-split phase of the split transaction. Specifically, it records the expected frame number into *QH.FrameTag* field (see Section), set *C-prog-mask* to zero (00h), and exits this state. Note that the host controller must not adjust the value of *CErr* as a result of completion of a start-split transaction.

Periodic Interrupt - Do Complete Split

This state is entered unconditionally from the Do Start Split state after a start-split transaction is executed on the bus. Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it checks to determine whether a complete-split transaction should be executed now.

There are four tests to determine whether a complete-split transaction should be executed.

- Test A. *cMicroFrameBit* is bit-wise anded with *QH.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame.
- Test B. *QH.FrameTag* is compared with the current contents of *FRINDEX*[7:3]. An equal indicates a match.
- Test C. The complete-split progress bit vector is checked to determine whether the previous bit is set, indicating that the previous complete-split was appropriately executed. An example algorithm for this test is provided below:

```
Algorithm Boolean CheckPreviousBit(QH.C-prog-mask, QH.C-mask, cMicroFrameBit)
Begin
-- Return values:
-- TRUE - no error
-- FALSE - error
--
Boolean rvalue = TRUE;
previousBit = cMicroframeBit logical-rotate-right(1)
-- Bit-wise anding previousBit with C-mask indicates
-- whether there was an intent
-- to send a complete split in the previous micro-frame. So,
-- if the
-- 'previous bit' is set in C-mask, check C-prog-mask to
-- make sure it
-- happened.
If (previousBit bitAND QH.C-mask)then
                                If not(previousBit bitAND QH.C-prog-mask) then
                      rvalue = FALSE;
          End if
End If
-- If the C-prog-mask already has a one in this bit position,
```

```
-- then an aliasing
-- error has occurred. It will probably get caught by the
-- FrameTag Test, but
-- at any rate it is an error condition that as detectable here
-- should not allow
-- a transaction to be executed.
If (cMicroFrameBit bitAND QH.C-prog-mask) then
 rvalue = FALSE;
End if
return (rvalue)
End Algorithm
```

- Test D. Check to see if a start-split should be executed in this micro-frame. Note this is the same test performed in the Do Start Split state (see Section Periodic Isochronous - Do Start Split). Whenever it evaluates to TRUE and the controller is NOT processing in the context of a *Recovery Path* mode, it means a start-split should occur in this micro-frame. Test D and Test A evaluating to TRUE at the same time is a system software error. Behavior is undefined.

If (A .and. B .and. C .and. not(D)) then the host controller will execute a complete-split transaction. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. On completion of the complete-split transaction, the host controller records the result of the transaction in the queue head and sets *QH.FrameTag* to the expected *H-Frame* number (see Section). The effect to the state of the queue head and thus the state of the transfer depends on the response by the transaction translator to the complete-split transaction. The following responses have the effects (note that any responses that result in decrementing of the *CErr* will result in the queue head being halted by the host controller if the result of the decrement is zero):

- NYET (and Last). On each NYET response, the host controller checks to determine whether this is the last complete-split for this split transaction. Last is defined in this context as the condition where all of the scheduled complete-splits have been executed. If it is the last complete-split (with a NYET response), then the transfer state of the queue head is not advanced (never received any data) and this state exited. The transaction translator must have responded to all the clompete-splits with NYETs, meaning that the start-split issued by the host controller was not received. The start-split should be retried at the next poll period.
- The test for whether this is the Last complete split can be performed by XOR *QH.C-mask* with *QH.C-prog-mask*. If the result is all zeros then all complete-splits have been executed. When this condition occurs, the *XactErr* status bit is set to a one and the *CErr* field is decremented.
- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask* and *FrameTag*) and stay in this state. The host controller must not adjust *CErr* on this response.

- Transaction Error (XactErr). Timeout, data CRC failure, etc. The *CErr* field is decremented and the *XactErr* bit in the *Status* field is set to a one. The complete split transaction is *immediately* retried (if *Cerr* is non-zero).If there is not enough time in the micro-frame to complete the retry and the endpoint is an IN, or *CErr* is decremented to a zero from a one, the queue is halted. If there is not enough time in the micro-frame to complete the retry and the endpoint is an OUT and *CErr* is not zero, then this state is exited (that is, return to Do Start Split). This results in a retry of the entire OUT split transaction, at the next poll period. Refer to Chapter 11 Hubs (specifically the section full- and low-speed Interrupts) in the USB Specification Revision 2.0 for detailed requirements on why these errors must be immediately retried.
- ACK. This can only occur if the target endpoint is an OUT. The target endpoint ACK'd the data and this response is a propagation of the endpoint ACK up to the host controller. The host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount. And the data toggle bit (*dt*) is toggled. The host controller will then exit this state for this queue head. The host controller must reload *CErr* with maximum value on this response. Advancing the transfer state may cause other process events such as retirement of the qTD and advancement of the queue (see Section Managing Control/Bulk/Interrupt Transfers through Queue Heads).
- MDATA. This response will only occur for an IN endpoint. The transaction translator responded with zero or more bytes of data and an MDATA PID. The incremental number of bytes received is accumulated in *QH.S-bytes*. The host controller must not adjust *CErr* on this response.
- DATA0/1. This response may only occur for an IN endpoint. The number of bytes received is added to the accumulated byte count in *QH.S-bytes*. The state of the transfer is advanced by the result and the host controller will exit this state for this queue head.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Section Managing Control/Bulk/Interrupt Transfers through Queue Heads).
- If the data sequence PID does not match the expected, the entirety of the data received in this split transaction is ignored, the transfer state is not advanced and this state is exited.
- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced, and this state is exited. The host controller must reload *CErr* with maximum value on this response.

- ERR. There was an error during the full- or low-speed transaction. The ERR status bit is set to a one, *Cerr* is decremented, the state of the transfer is not advanced, and this state is exited.
- STALL. The queue is halted (an exit condition of the Execute Transaction state). The status field bits: *Active* bit is set to zero and the *Halted* bit is set to a one and the qTD is retired. Responses which are not enumerated in the list or which are received out of sequence are illegal and may result in undefined host controller behavior. The other possible combinations of tests A, B, C, and D may indicate that data or response was lost. The table below lists the possible combinations and the appropriate action.

Table 54-49. Interrupt IN/OUT Do Complete Split State Execution Criteria

Condition	Action	Description
not(A) not(D)	Ignore QHD	Neither a start nor complete-split is scheduled for the current micro-frame.Host controller should continue walking the schedule.
A not(C)	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	Progress bit check failed. These means a complete-split has been missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CER*R. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A not(B) C	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	*QH.FrameTag* test failed. This means that exactly one or more *H-Frames* have been skipped. This means complete-splits and have missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CER*R. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A B C not(D)	Execute complete-split	This is the non-error case where the host controller executes a complete-split transaction.
D	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	This is a degenerate case where the start-split was issued, but all of the complete-splits were skipped and all possible intervening opportunities to detect the missed data failed to fire. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect *CER*R. In either case, set the *Missed Micro-frame* bit in the status field to a one. Note: When executing in the context of a *Recovery Path* mode, the host controller is allowed to process the queue head and take the actions indicated above, or it may wait until the queue head is visited in the

Table 54-49. Interrupt IN/OUT Do Complete Split State Execution Criteria

		normal processing mode. Regardless, the host controller must not execute a start-split in the context of a executing in a *Recovery Path* mode.

Managing QH.FrameTag Field

The *QH.FrameTag* field in a queue head is completely managed by the host controller. The rules for setting *QH.FrameTag* are simple:

- Rule 1: If transitioning from Do Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is 6 *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates split transactions whose start-split and complete-splits are in different *H-Frames* (case 2a, see Figure 54-20).
- Rule 2: If the current value of *FRINDEX*[2:0] is 7, *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates staying in Do Complete Split for cases 2a, 2b, and 2c (Figure 54-20).
- Rule 3: If transitioning from Do_Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is not 6, or currently in Do Complete Split and the current value of (*FRINDEX*[2:0]) is not 7, *FrameTag* is set to *FRINDEX*[7:3]. This accommodates all other cases (Figure 54-20).

54.5.3.12.2.6 Rebalancing the periodic schedule

System software must occasionally adjust a periodic queue head's S-mask and C-mask fields during operation.

This need occurs when adjustments to the periodic schedule create a new bandwidth budget and one or more queue head's are assigned new execution footprints (that is, new S-mask and C-mask values).

It is imperative that System software must not update these masks to new values in the midst of a split transaction. In order to avoid any race conditions with the update, the EHCI host controller provides a simple assist to system software. System software sets the *Inactivate-on-next-Transaction* (*I*) bit to a one to signal the host controller that it intends to update the S-mask and C-mask on this queue head. System software will then wait for the host controller to observe the *I-bit* is a one and transition the *Active* bit to a zero. The rules for how and when the host controller sets the *Active* bit to zero are enumerated below:

- If the *Active* bit is a zero, no action is taken. The host controller does not attempt to advance the queue when the *I-bit* is a one.
- If the *Active* bit is a one and the *SplitXState* is DoStart (regardless of the value of *S-mask*), the host controller will simply set *Active* bit to a zero. The host controller is

not required to write the transfer state back to the *current* qTD. Note that if the *S-mask* indicates that a start-split is scheduled for the current micro-frame, the host controller must not issue the start-split bus transaction. It must set the *Active* bit to zero.

System software must save transfer state before setting the *I-bit* to a one. This is required so that it can correctly determine what transfer progress (if any) occurred after the *I-bit* was set to a one and the host controller executed its final bus-transaction and set *Active* to a zero.

After system software has updated the S-mask and C-mask, it must then reactivate the queue head. Because the *Active* bit and the *I-bit* cannot be updated with the same write, system software needs to use the following algorithm to coherently re-activate a queue head that has been stopped via the *I-bit*.

1. Set the *Halted* bit to a one, then
2. Set the *I-bit* to a zero, then
3. Set the *Active* bit to a one and the *Halted* bit to a zero in the same write.

Setting the *Halted* bit to a one inhibits the host controller from attempting to advance the queue between the time the *I-bit* goes to a zero and the *Active* bit goes to a one.

54.5.3.12.3 Split Transaction Isochronous

Full-speed isochronous transfers are managed using the split-transaction protocol through a USB 2.0 transaction translator in a USB2.0 Hub. The EHCI controller utilizes siTD data structure to support the special requirements of isochronous split-transactions.

This data structure uses the scheduling model of isochronous TDs (iTD, Section Isochronous (High-Speed) Transfer Descriptor (iTD)) (see Section Managing Isochronous Transfers Using iTDs for the operational model of iTDs) with the contiguous data feature provided by queue heads. This simple arrangement allows a single isochronous scheduling model and adds the additional feature that all data received from the endpoint (per split transaction) must land into a contiguous buffer.

54.5.3.12.3.1 Split Transaction Scheduling Mechanisms for Isochronous

Full-speed isochronous transactions are managed through a transaction translator's periodic pipeline. As with full- and low-speed interrupt, system software manages each transaction translator's periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each full-speed isochronous endpoint occur.

The requirements described in Section Split Transaction Scheduling Mechanisms for Interrupt apply. The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule. The ^{S}X and ^{C}X labels indicate micro-frames where software can schedule start- and complete-splits (respectively). The *H-Frame* boundaries are marked with a large, solid bold vertical line. The *B-Frame* boundaries are marked with a large, bold, dashed line. The bottom of the figure illustrates the relationship of an siTD to the *H-Frame*.

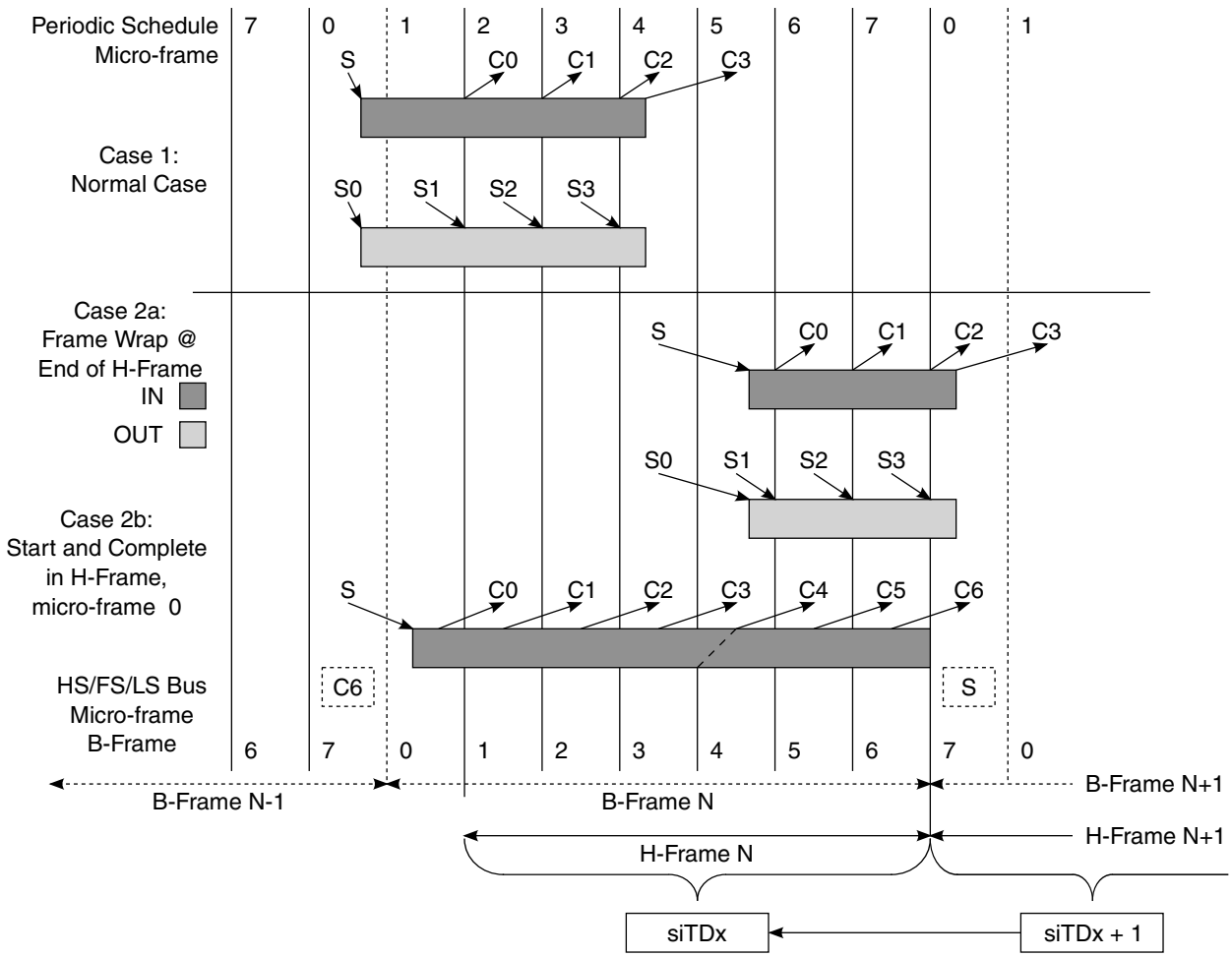


Figure 54-24. Split Transaction, Isochronous Scheduling Boundary Conditions

When the endpoint is an isochronous OUT, there are only start-splits, and no complete-splits. When the endpoint is an isochronous IN, there is at most one start-split and one to *N* complete-splits. The scheduling boundary cases are:

- *Case 1*: The entire split transaction is completely bounded by an *H-Frame*. For example: the start-splits and complete-splits are all scheduled to occur in the same *H-Frame*.

- *Case 2a*: This boundary case is where one or more (at most two) complete-splits of a split transaction IN are scheduled across an *H-Frame* boundary. This can only occur when the split transaction has the possibility of moving data in *B-Frame,* micro-frames 6 or 7 (*H-Frame* micro-frame 7 or 0). When an *H-Frame* boundary wrap condition occurs, the scheduling of the split transaction spans more than one location in the periodic list.(For example, it takes two siTDs in adjacent periodic frame list locations to fully describe the scheduling for the split transaction.)
- Although the scheduling of the split transaction may take two data structures, all of the complete-splits for each full-speed IN isochronous transaction must use only one data pointer. For this reason, siTDs contain a back pointer, the use of which is described below.
- Software must never schedule full-speed isochronous OUTs across an *H-Frame* boundary.
- *Case 2b*: This case can only occur for a very large isochronous IN. It is the only allowed scenario where a start-split and complete-split for the same endpoint can occur in the same micro-frame. Software must enforce this rule by scheduling the large transaction first. Large is defined to be anything larger than 579 byte maximum packet size.

A subset of the same mechanisms employed by full- and low-speed interrupt queue heads are employed in siTDs to schedule and track the portions of isochronous split transactions. The following fields are initialized by system software to instruct the host controller when to execute portions of the split transaction protocol.

- *SplitXState*. This is a single bit residing in the *Status* field of an siTD (see Figure 54-25). This bit is used to track the current state of the split transaction. The rules for managing this bit are described in Section Split Transaction Execution State Machine for Interrupt.
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 54-24, case one, the *S-mask* would have a value of 00000001b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do Start Split, and the current micro-frame as indicated by USB_n_FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask.* This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 54-24, case one, the *C-mask* would have a value of 00111100b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do

Complete Split, and the current micro-frame as indicated by *USB_n_FRINDEX*[2:0] is 2, 3, 4, or 5, then execute a complete-split transaction.

- *Back Pointer*. This field in a siTD is used to complete an IN split-transaction using the previous *H-Frame*'s siTD. This is only used when the scheduling of the complete-splits span an *H-Frame* boundary.

There exists a one-to-one relationship between a high-speed isochronous split transaction (including all start- and complete-splits) and one full-speed isochronous transaction. An siTD contains (amongst other things) buffer state and split transaction scheduling information. An siTD's buffer state always maps to one full-speed isochronous data payload. This means that for any full-speed transaction payload, a single siTD's data buffer must be used. This rule applies to both IN an OUTs. An siTD's scheduling information usually also maps to one high-speed isochronous split transaction. The exception to this rule is the *H-Frame* boundary wrap cases mentioned above.

The siTD data structure describes at most, one frame's worth of high-speed transactions and that description is strictly bounded within a frame boundary. The figure below illustrates some examples. On the top are examples of the full-speed transaction footprints for the boundary scheduling cases described above. In the middle are time-frame references for both the *B-Frames* (HS/FS/LS Bus) and the *H-Frames*. On the bottom is illustrated the relationship between the scope of an siTD description and the time references. Each *H-Frame* corresponds to a single location in the periodic frame list. The implication is that each siTD is reachable from a single periodic frame list location at a time.

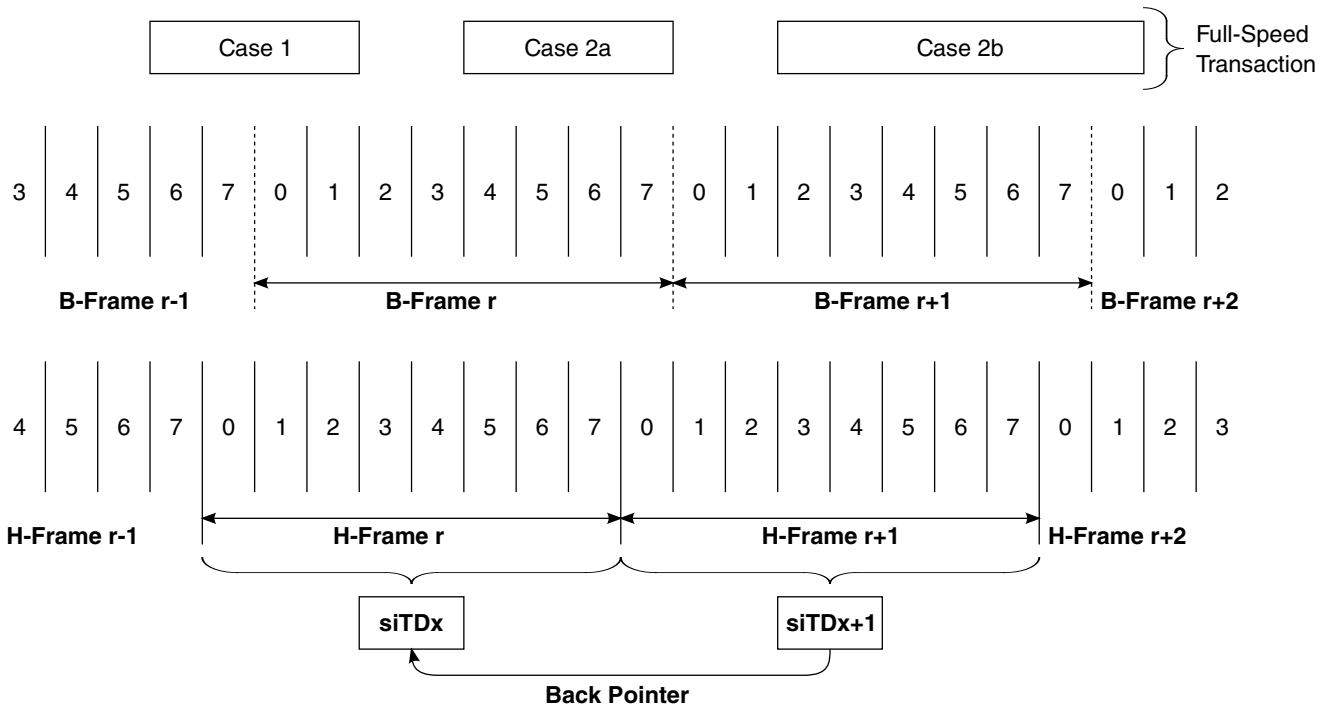


Figure 54-25. siTD Scheduling Boundary Examples

Each case is described below:

- *Case 1*: One siTD is sufficient to describe and complete the isochronous split transaction because the whole isochronous split transaction is tightly contained within a single *H-Frame*.
- *Case 2a, 2b*: Although both INs and OUTs can have these footprints, OUTs always take only one siTD to schedule. However, INs (for these boundary cases) require two siTDs to complete the scheduling of the isochronous split transaction $siTD_X$ is used to always issue the start-split and the first *N* complete-splits. The full-speed transaction (for these cases) can deliver data on the full-speed bus segment during micro-frame 7 of $H\text{-}Frame_{Y+1}$, or micro-frame 0 of $H\text{-}Frame_{Y+2}$. The complete splits are scheduled using $siTD_{X+2}$ (not shown). The complete-splits to extract this data must use the buffer pointer from $siTD_{X+1}$. The only way for the host controller to reach $siTD_{X+1}$ from $H\text{-}Frame_{Y+2}$is to use $siTD_{X+2}$'s back pointer. The host controller rules for when to use the back pointer are described is Section Periodic Isochronous - Do Complete Split .

Software must apply the following rules when calculating the schedule and linking the schedule data structures into the periodic schedule:

- Software must ensure that an isochronous split-transaction is started so that it will complete before the end of the *B-Frame*.
- Software must ensure that for a single full-speed isochronous endpoint, there is never a start-split and complete-split in *H-Frame, micro-frame 1*. This is mandated as a rule so that case 2a and case 2b can be discriminated. According to the core USB specification, the long isochronous transaction illustrated in Case 2b, could be scheduled so that the start-split was in micro-frame 1 of *H-Frame* N and the last complete-split would need to occur in micro-frame 1 of H-Frame N+1. However, it is impossible to discriminate between cases 2a and case 2b, which has significant impact on the complexity of the host controller.

54.5.3.12.3.2 Tracking Split Transaction Progress for Isochronous Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where device to host data is lost. Isochronous endpoints do not employ the concept of a halt on error, however the host is required to identify and report per-packet errors observed in the data stream. This includes schedule traversal problems (skipped micro-frames), timeouts and corrupted data received.

In similar kind to interrupt split-transactions, the portions of the split transaction protocol must execute in the micro-frames they are scheduled. The queue head data structure used to manage full- and low-speed interrupt has several mechanisms for tracking when portions of a transaction have occurred. Isochronous transfers use siTDs, for their transfers, and the data structures are only reachable via the schedule in the exact micro-frame in which they are required (so all the mechanism employed for tracking in queue heads is not required for siTDs). Software has the option of reusing siTD several times in the complete periodic schedule. However, it must ensure that the results of split transaction N are consumed and the siTD reinitialized (activated) before the host controller gets back to the siTD (in a future micro-frame).

Split-transaction isochronous OUTs utilize a low-level protocol to indicate which portions of the split transaction data have arrived. Control over the low-level protocol is exposed in an siTD via the fields *Transaction Position* (*TP*) and *Transaction Count* (*T-count*). If the entire data payload for the OUT split transaction is larger than 188 bytes, there will be more than one start-split transaction, each of which require proper annotation. If host hold-offs occur, then the sequence of annotations received from the host will not be complete, which is detected and handled by the transaction translator. See Section Periodic Isochronous - Do Start Split for a description on how these fields are used during a sequence of start-split transactions.

The fields *siTD.T-Count* and *siTD.TP* are used by the host controller to drive and sequence the transaction position annotations. It is the responsibility of system software to properly initialize these fields in each siTD. Once the budget for a split-transaction

isochronous endpoint is established, *S-mask*, *T-Count*, and *TP* initialization values for all the siTD associated with the endpoint are constant. They remain constant until the budget for the endpoint is recalculated by software and the periodic schedule adjusted.

For IN-endpoints, the transaction translator simply annotates the response data packets with enough information to allow the host controller to identify the last data. As with split transaction Interrupt, it is the host controller's responsibility to detect when it has missed an opportunity to execute a complete-split. The following field in the siTD is used to track and detect errors in the execution of a split transaction for an IN isochronous endpoint.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets a bit for each complete-split executed. The bit position is determined by the micro-frame (USB_n_FRINDEX[2:0]) number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a complete-split transaction. If the previous complete-splits have not been executed, then it means one (or more) have been skipped and data has potentially been lost. System software is required to initialize this field to zero before setting an siTD's *Active* bit to a one.

If a transaction translator returns with the final data before all of the complete-splits have been executed, the state of the transfer is advanced so that the remaining complete-splits are not executed. Refer to Section Asynchronous - Do Complete Split for a description on how the state of the transfer is advanced. It is important to note that an IN siTD is retired based solely on the responses from the Transaction Translator to the complete-split transactions. This means, for example, that it is possible for a transaction translator to respond to a complete-split with an MDATA PID. The number of bytes in the MDATA's data payload could cause the siTD field *Total Bytes to Transfer* to decrement to zero. This response can occur, before all of the scheduled complete-splits have been executed. In other interface, data structures (for example, high-speed data streams through queue heads), the transition of *Total Bytes to Transfer* to zero signals the end of the transfer and results in setting of the *Active* bit to zero. However, in this case, the result has not been delivered by the Transaction Translator and the host must continue with the next complete-split transaction to extract the residual transaction state. This scenario occurs because of the pipeline rules for a Transaction Translator (see Chapter 11 of the Universal Serial Bus Revision 2.0). In summary the periodic pipeline rules require that on a micro-frame boundary, the Transaction Translator will hold the final two bytes received (if it has not seen an End Of Packet (EOP)) in the full-speed bus pipe stage and give the

remaining bytes to the high-speed pipeline stage. At the micro-frame boundary, the Transaction Translator could have received the entire packet (including both CRC bytes) but not received the packet EOP. In the next micro-frame, the Transaction Translator will respond with an MDATA and send all of the data bytes (with the two CRC bytes being held in the full-speed pipeline stage). This could cause the siTD to decrement its *Total Bytes to Transfer* field to zero, indicating it has received all expected data. The host must still execute one more (scheduled) complete-split transaction in order to extract the results of the full-speed transaction from the Transaction Translator (for example, the Transaction Translator may have detected a CRC failure, and this result must be forwarded to the host).

If the host experiences hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous OUT, then the protocol to the transaction translator will not be consistent and the transaction translator will detect and react to the problem. Likewise, for host hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous IN, the *C-prog-mask* is used by the host controller to detect errors. However, if the host experiences a hold-off that causes it to skip all of an siTD, or an siTD expires during a host hold off (for example, a hold-off occurs and the siTD is no longer reachable by the host controller in order for it to report the hold-off event), then system software must detect that the siTDs have not been processed by the host controller (that is, state not advanced) and report the appropriate error to the client driver.

54.5.3.12.3.3 Split Transaction Execution State Machine for Isochronous

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame.*

If the *Active* bit in the *Status* byte is a zero, the host controller will ignore the siTD and continue traversing the periodic schedule. Otherwise the host controller will process the siTD as specified below. A split transaction state machine is used to manage the split-transaction protocol sequence. The host controller uses the fields defined in Section Tracking Split Transaction Progress for Interrupt Transfers , plus the variable *cMicroFrameBit* defined in Section Split Transaction Execution State Machine for Interrupt to track the progress of an isochronous split transaction. The figure below illustrates the state machine for managing an siTD through an isochronous split transaction. Bold, dotted circles denote the state of the *Active* bit in the *Status* field of a siTD. The Bold, dotted arcs denote the transitions between these states. Solid circles denote the states of the split transaction state machine and the solid arcs denote the transitions between these states. Dotted arcs and boxes reference actions that take place either as a result of a transition or from being in a state.

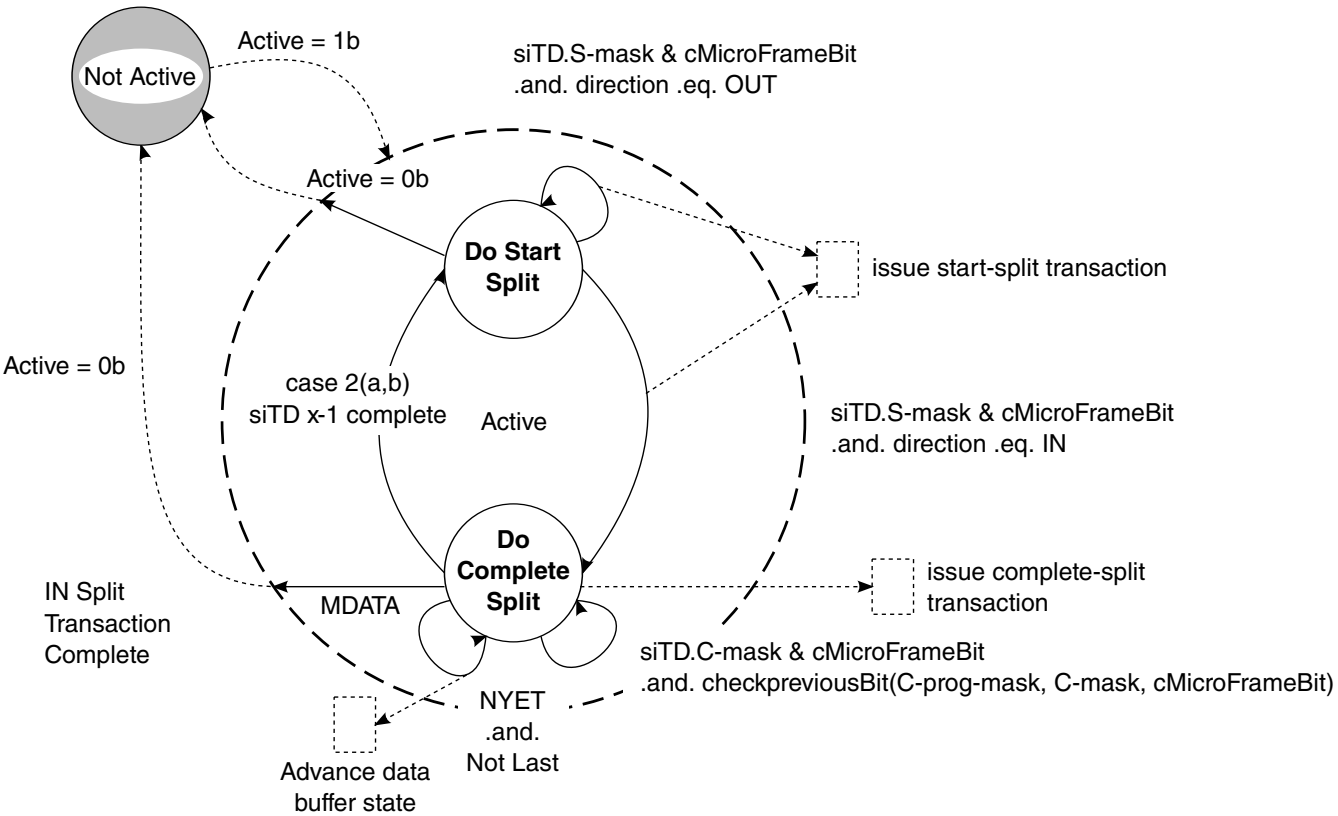


Figure 54-26. Split Transaction State Machine for Isochronous

54.5.3.12.3.4 Periodic Isochronous - Do Start Split

Isochronous split transaction OUTs use only this state.

An siTD for a split-transaction isochronous IN is either initialized to this state, or the siTD transitions to this state from Do Complete Split when a case 2a (IN) or 2b scheduling boundary isochronous split-transaction completes.

Each time the host controller reaches an active siTD in this state, it checks the *siTD.S-mask* against *cMicroFrameBit*. If there is a one in the appropriate position, the siTD will execute a start-split transaction. By definition, the host controller cannot *reach* an siTD at the wrong time. If the *I/O* field indicates an IN, then the start-split transaction includes only the extended token plus the full-speed token. Software must initialize the *siTD.Total Bytes To Transfer* field to the number of bytes expected. This is usually the maximum packet size for the full-speed endpoint. The host controller exits this state when the start-split transaction is complete.

The remainder of this section is specific to an isochronous OUT endpoint (that is, the *I/O* field indicates an OUT). When the host controller executes a start-split transaction for an isochronous OUT it includes a data payload in the start-split transaction. The memory

buffer address for the data payload is constructed by concatenating *siTD.Current Offset* with the page pointer indicated by the page selector field (*siTD.P*). A zero in this field selects Page 0 and a 1 selects Page 1. During the start-split for an OUT, if the data transfer crosses a page boundary during the transaction, the host controller must detect the page cross, update the *siTD.P*-bit from a zero to a one, and begin using the *siTD.Page 1* with *siTD.Current Offset* as the memory address pointer. The field *siTD.TP* is used to annotate each start-split transaction with the indication of which part of the split-transaction data the current payload represents (ALL, BEGIN, MID, END). In all cases the host controller simply uses the value in *siTD.TP* to mark the start-split with the correct transaction position code.

T-Count is always initialized to the number of start-splits for the current frame. *TP* is always initialized to the first required transaction position identifier. The scheduling boundary case (see Figure 54-25) is used to determine the initial value of *TP*. The initial cases are summarized in the following table.

Table 54-50. Initial Conditions for OUT siTD's TP and T-count Fields

Case	T-count	TP	Description
1, 2a	=1	ALL	When the OUT data payload is less than (or equal to) 188 bytes, only one start-split is required to move the data. The one start-split must be marked with an ALL.
1, 2a	!=1	BEGIN	When the OUT data payload is greater than 188 bytes more than one start-split must be used to move the data. The initial start-split must be marked with a BEGIN.

After each start-split transaction is complete, the host controller updates *T-Count* and *TP* appropriately so that the next start-split is correctly annotated.

The table below illustrates all of the *TP* and *T-count* transitions, which must be accomplished by the host controller.

.

Table 54-51. Transaction Position (TP)/Transaction Count (T-Count) Transition Table

TP	T-count next	TP next	Description
ALL	0	N/A	Transition from ALL, to done.
BEGIN	1	END	Transition from BEGIN to END. Occurs when *T-count* starts at 2.
BEGIN	!=1	MID	Transition from BEGIN to MID. Occurs when *T-count* starts at greater than 2.
MID	!=1	MID	*TP* stays at MID while *T-count* is not equal to 1 (that is, greater than 1). This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 3.
MID	1	END	Transition from MID to END. This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 2.

The start-split transactions do not receive a handshake from the transaction translator, so the host controller always advances the transfer state in the siTD after the bus transaction is complete. To advance the transfer state the following operations take place:

- The *siTD.Total Bytes To Transfer* and the *siTD.Current Offset* fields are adjusted to reflect the number of bytes transferred.
- The *siTD.P* (page selector) bit is updated appropriately.
- The *siTD.TP* and *siTD.T-count* fields are updated appropriately as defined in Table 54-51.

These fields are then written back to the memory based siTD. The *S-mask* is fixed for the life of the current budget. As mentioned above, *TP* and *T-count* are set specifically in each siTD to reflect the data to be sent from this siTD. Therefore, regardless of the value of *S-mask*, the actual number of start-split transactions depends on *T-count* (or equivalently, *Total Bytes to Transfer*). The host controller must set the *Active* bit to a zero when it detects that all of the schedule data has been sent to the bus. The preferred method is to detect when *T-Count* decrements to zero as a result of a start-split bus transaction. Equivalently, the host controller can detect when *Total Bytes to Transfer* decrements to zero. Either implementation must ensure that if the initial condition is *Total Bytes to Transfer* equal to zero and *T-count* is equal to a one, then the host controller will issue a single start-split, with a zero-length data payload. Software must ensure that *TP*, *T-count* and *Total Bytes to Transfer* are set to deliver the appropriate number of bus transactions from each siTD. An inconsistent combination will yield undefined behavior.

If the host experiences hold-offs that cause the host controller to skip start-split transactions for an OUT transfer, the state of the transfer will not progress appropriately. The transaction translator will observe protocol violations in the arrival of the start-splits for the OUT endpoint (that is, the transaction position annotation will be incorrect as received by the transaction translator).

Example scenarios are described in Section Split Transaction for Isochronous - Processing Examples.

A host controller implementation can optionally track the progress of an OUT split transaction by setting appropriate bits in the *siTD.C-prog-mask* as it executes each scheduled start-split. The *checkPreviousBit*() algorithm defined in Periodic Isochronous - Do Complete Split can be used prior to executing each start-split to determine whether start-splits were skipped. The host controller can use this mechanism to detect missed micro-frames. It can then set the siTD's *Active* bit to zero and stop execution of this siTD. This saves on both memory and high-speed bus bandwidth.

54.5.3.12.3.5 Periodic Isochronous - Do Complete Split

This state is only used by a split-transaction isochronous IN endpoint.

This state is entered unconditionally from the Do Start State after a start-split transaction is executed for an IN endpoint. Each time the host controller visits an siTD in this state, it conducts a number of tests to determine whether it should execute a complete-split transaction. The individual tests are listed below. The sequence they are applied depends on which micro-frame the host controller is currently executing which means that the tests might not be applied until after the siTD referenced from the back pointer has been fetched.

- Test A. *cMicroFrameBit* is bit-wise anded with *siTD.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame. This test is always applied to a newly fetched siTD that is in this state.
- Test B. The *siTD.C-prog-mask* bit vector is checked to determine whether the previous complete splits have been executed. An example algorithm is below (this is slightly different than the algorithm used in Section Periodic Isochronous - Do Complete Split). The sequence in which this test is applied depends on the current value of USB_n_FRINDEX[2:0]. If USB_n_FRINDEX[2:0] is 0 or 1, it is not applied until the back pointer has been used. Otherwise it is applied immediately.

```
Algorithm Boolean CheckPreviousBit(siTD.C-prog-mask, siTD.C-mask, cMicroFrameBit)
Begin
          Boolean rvalue = TRUE;
          previousBit = cMicroFrameBit rotate-right(1)
          -- Bit-wise anding previousBit with C-mask indicates whether there was an intent
          -- to send a complete split in the previous micro-frame. So, if the
          -- 'previous bit' is set in C-mask, check C-prog-mask to make sure it
          -- happened.
          if previousBit bitAND siTD.C-mask then
                    if not (previousBit bitAND siTD.C-prog-mask) then
                              rvalue = FALSE
                    End if
          End if
          Return rvalue
End Algorithm
```

If Test A is true and USB_n_FRINDEX[2:0] is zero or one, then this is a case 2a or 2b scheduling boundary (see Figure 54-24). See Section Periodic Isochronous - Do Complete Split for details in handling this condition.

If Test A and Test B evaluate to true, then the host controller will execute a complete-split transaction using the transfer state of the current siTD. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. The transfer state is advanced based on the completion status of the complete-split transaction. To advance the transfer state of an IN siTD, the host controller must:

- Decrement the number of bytes received from *siTD.Total Bytes To Transfer,*
- Adjust *siTD.Current Offset* by the number of bytes received,

- Adjust *siTD.P* (page selector) field if the transfer caused the host controller to use the next page pointer, and
- Set any appropriate bits in the *siTD.Status* field, depending on the results of the transaction.

Note that if the host controller encounters a condition where *siTD.Total Bytes To Transfer* is zero, and it receives more data, the host controller must not write the additional data to memory. The *siTD.Status.Active* bit must be set to zero and the *siTD.Status.Babble Detected* bit must be set to a one. The fields *siTD.Total Bytes To Transfer*, *siTD.Current Offset*, and *siTD.P* (page selector) are not required to be updated as a result of this transaction attempt.

The host controller must accept (assuming good data packet CRC and sufficient room in the buffer as indicated by the value of *siTD.Total Bytes To Transfer*) MDATA and DATA0/1 data payloads up to and including 192 bytes. A host controller implementation may optionally set *siTD.Status Active* to a zero and *siTD.Status.Babble Detected* to a one when it receives and MDATA or DATA0/1 with a data payload of more than 192 bytes. The following responses have the noted effects:

- ERR. The full-speed transaction completed with a time-out or bad CRC and this is a reflection of that error to the host. The host controller sets the *ERR* bit in the *siTD.Status* field and sets the *Active* bit to a zero.
- Transaction Error (XactErr). The complete-split transaction encounters a Timeout, CRC16 failure, etc. The *siTD.Status* field *XactErr* field is set to a one and the complete-split transaction must be retried immediately. The host controller must use an internal error counter to count the number of retries as a counter field is not provided in the siTD data structure. The host controller will not retry more than two times. If the host controller exhausts the retries or the end of the micro-frame occurs, the *Active* bit is set to zero.
- DATAx (0 or 1). This response signals that the final data for the split transaction has arrived. The transfer state of the siTD is advanced and the *Active* bit is set to a zero. If the *Bytes To Transfer* field has not decremented to zero (including the reception of the data payload in the DATAx response), then less data than was expected, or allowed for was actually received. This *short packet* event does not set the USBINT status bit in the USBSTS register to a one. The host controller will not detect this condition.
- NYET (and Last). On each NYET response, the host controller also checks to determine whether this is the last complete-split for this split transaction. Last was defined in Section Periodic Isochronous - Do Complete Split . If it is the last complete-split (with a NYET response), then the transfer state of the siTD is not advanced (never received any data) and the *Active* bit is set to a zero. No bits are set in the *Status* field because this is essentially a skipped transaction. The transaction translator must have responded to all the scheduled clompete-splits with NYETs,

meaning that the start-split issued by the host controller was not received. This result should be interpreted by system software as if the transaction was completely skipped. The test for whether this is the last complete split can be performed by XORing C-mask with C-prog-mask. A zero result indicates that all complete-splits have been executed.

- MDATA (and Last). See above description for testing for Last. This can only occur when there is an error condition. Either there has been a babble condition on the full-speed link, which delayed the completion of the full-speed transaction, or software set up the *S-mask* and/or *C-mask*s incorrectly. The host controller must set *XactErr* bit to a one and the *Active* bit is set to a zero.
- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask*) and stay in this state.
- MDATA (and not Last). The transaction translator responds with an MDATA when it has partial data for the split transaction. For example, the full-speed transaction data payload spans from micro-frame X to X+1 and during micro-frame X, the transaction translator will respond with an MDATA and the data accumulated up to the end of micro-frame X. The host controller advances the transfer state to reflect the number of bytes received.

If Test A succeeds, but Test B fails, it means that one or more of the complete-splits have been skipped. The host controller sets the *Missed Micro-Frame* status bit and sets the *Active* bit to a zero.

54.5.3.12.3.6 Complete-Split for Scheduling Boundary Cases 2a, 2b

Boundary cases 2a and 2b (INs only) (see Figure 54-24) require that the host controller use the transaction state context of the previous siTD to finish the split transaction. The table below enumerates the transaction state fields.

Table 54-52. Summary siTD Split Transaction State

Buffer State	Status	Execution Progress
Total Bytes To Transfer P (page select) Current Offset TP (transaction position) T-count (transaction count)	All bits in the status field	C-prog-mask

NOTE

TP and *T-count* are used only for Host to Device (OUT) endpoints.

If software has budgeted the schedule of this data stream with a frame wrap case, then it must initialize the *siTD.Back Pointer* field to reference a valid siTD and will have the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer*

field set to a zero. Otherwise, software must set the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer* field to a one. The host controller's rules for interpreting when to use the *siTD.Back Pointer* field are listed below. These rules apply only when the siTD's *Active* bit is a one and the *SplitXState* is Do Complete Split.

- When cMicroFrameBit is a 1h and the siTDX.Back Pointer.T-bit is a zero, or
- If cMicroFrameBit is a 2h and siTDX.S-mask[0] is a zero

When either of these conditions apply, then the host controller must use the transaction state from $siTD_{X-1}$.

In order to access $siTD_{X-1}$, the host controller reads on-chip the siTD referenced from $siTD_X$.*Back Pointer*.

The host controller must save the entire state from $siTD_X$ while processing $siTD_{X-1}$. This is to accommodate for case 2b processing. The host controller must not recursively walk the list of *siTD.Back Pointers*.

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is Do Complete Split), then both Test A and Test B are applied as described above. If these criteria to execute a complete-split are met, the host controller executes the complete split and evaluates the results as described above. The transaction state (see Table 54-52) of $siTD_{X-1}$ is appropriately advanced based on the results and written back to memory. If the resultant state of $siTD_{X-1}$'s *Active* bit is a one, then the host controller returns to the context of $siTD_X$, and follows its next pointer to the next schedule item. No updates to $siTD_X$ are necessary.

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is Do Start Split), then the host controller must set *Active* bit to a zero and *Missed Micro-Frame* status bit to a one and the resultant status written back to memory.

If $siTD_{X-1}$*'sActive* bit is a zero, (because it was zero when the host controller first visited $siTD_{X-1}$ via $siTD_X$*'s* back pointer, it transitioned to zero as a result of a detected error, or the results of $siTD_{X-1}$*'s* complete-split transaction transitioned it to zero), then the host controller returns to the context of $siTD_X$ and transitions its *SplitXState* to Do Start Split. The host controller then determines whether the case 2b start split boundary condition exists (that is, if *cMicroframeBit* is a 1b and $siTD_X$.*S-mask*[0] is a 1b). If this criterion is met the host controller immediately executes a start-split transaction and appropriately advances the transaction state of $siTD_X$, then follows $siTD_X$.*Next Pointer* to the next schedule item. If the criterion is not met, the host controller simply follows $siTD_X$.*Next Pointer* to the next schedule item. Note that in the case of a 2b boundary case, the split-transaction of $siTD_{X-1}$ will have its *Active* bit set to zero when the host controller returns

to the context of $siTD_X$. Also, note that software should not initialize an siTD with *C-mask* bits 0 and 1 set to a one and an *S-mask* with bit zero set to a one. This scheduling combination is not supported and the behavior of the host controller is undefined.

54.5.3.12.3.7 Split Transaction for Isochronous - Processing Examples

There is an important difference between how the hardware/software manages the isochronous split transaction state machine and how it manages the asynchronous and interrupt split transaction state machines.

The asynchronous and interrupt split transaction state machines are encapsulated within a single queue head. The progress of the data stream depends on the progress of each split transaction. In some respects, the split-transaction state machine is sequenced via the Execute Transaction queue head traversal state machine (see Figure 54-17).

Isochronous is a pure time-oriented transaction/data stream. The interface data structures are optimized to efficiently describe transactions that need to occur at specific times. The isochronous split-transaction state machine must be managed across these time-oriented data structures. This means that system software must correctly describe the scheduling of split-transactions across more than one data structure.

Then the host controller must make the appropriate state transitions at the appropriate times, in the correct data structures.

For example, the table below illustrates a couple of frames worth of scheduling required to schedule a case 2a full-speed isochronous data stream.

Table 54-53. Example Case 2a - Software Scheduling siTDs for an IN Endpoint

siTDX		Micro-Frames								Initial
#	Masks	0	1	2	3	4	5	6	7	SplitXState
X	S-Mask	-	-	-	-	1	-	-	-	Do Start Split
	C-Mask	1	1	-	-	-	-	1	1	
X+1	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+2	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+3	S-Mask	Repeats previous pattern								Do Complete Split
	C-Mask									

This example shows the first three siTDs for the transaction stream. Because this is the case-2a frame-wrap case, *S-masks* of all siTDs for this endpoint have a value of 10h (a one bit in micro-frame 4) and *C-mask* value of C3h (one-bits in micro-frames 0,1, 6 and 7). Additionally, sofware ensures that the *Back Pointer* field of each siTD references the appropriate siTD data structure (and the *Back PointerT-bits* are set to zero).

The initial *SplitXState* of the first siTD is Do Start Split. The host controller will visit the first siTD eight times during frame X. The C-mask bits in micro-frames 0 and 1 are ignored because the state is Do Start Split. During micro-frame 4, the host controller determines that it can run a start-split (and does) and changes *SplitXState* to Do Complete Split. During micro-frames 6 and 7, the host controller executes complete-splits. Notice the siTD for frame X+1 has its *SplitXState* initialized to Do Complete Split. As the host controller continues to traverse the schedule during *H-Frame* X+1, it will visit the second siTD eight times. During micro-frames 0 and 1 it will detect that it must execute complete-splits.

During *H-Frame* X+1, micro-frame 0, the host controller detects that $siTD_{X+1}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+1}$ and fetches $siTD_X$. It executes the complete split transaction using the transaction state of $siTD_X$. If the $siTD_X$ split transaction is complete, siTD's *Active* bit is set to zero and results written back to $siTD_X$. The host controller retains the fact that $siTD_X$ is retired and transitions the *SplitXState* in the $siTD_{X+1}$ to Do Start Split. At this point, the host controller is prepared to execute the start-split for $siTD_{X+1}$ when it reaches micro-frame 4. If the split-transaction completes early (transaction-complete is defined in Section Periodic Isochronous - Do Complete Split), that is, before all the scheduled complete-splits have been executed, the host controller will transition $siTD_X$.*SplitXState* to Do Start Split early and naturally skip the remaining scheduled complete-split transactions. For this example, $siTD_{X+1}$ does not receive a DATA0 response until *H-Frame* X+2, micro-frame 1.

During *H-Frame X+2,* micro-frame 0, the host controller detects that $siTD_{X+2}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. As described above, it executes another split transaction, receives an MDATA response, updates the transfer state, but does not modify the *Active* bit. The host controller returns to the context of $siTD_{X+2}$, and traverses its next pointer without any state change updates to $siTD_{X+2}$. S

During *H-Frame X+2*, micro-frame 1, the host controller detects $siTD_{X+2}$'s *S-mask[0]* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. It executes another complete-split transaction, receives a DATA0 response, updates the transfer state and sets the *Active* bit to a zero. It returns to the state of $siTD_{X+2}$ and changes its *SplitXState* to Do Start Split. At this point, the host controller is prepared to execute start-splits for $siTD_{X+2}$ when it reaches micro-frame 4.

54.5.3.13 Host Controller Pause

When the host controller's *HCHalted* bit in the USBSTS register is a zero, the host controller is sending SOF (Start OF Frame) packets down all enabled ports.

When the schedules are enabled, the EHCI host controller will access the schedules in main memory each micro-frame. This constant pinging of main memory is known to create Arm platform power management problems for mobile systems. Specifically, mobile systems aggressively manage the state of the Arm platform, based on recent history usage. In the more aggressive power saving modes, the Arm platform can disable its caches. Current PC architectures assume that bus-master accesses to main memory must be cache-coherent. So, when bus masters are busy touching memory, the Arm platform power management software can detect this activity over time and inhibit the transition of the Arm platform into its lowest power savings mode. USB controllers are bus-masters and the frequency at which they access their memory-based schedules keeps the Arm platform power management software from placing the Arm platform into its lowest power savings state.

USB Host controllers don't access main memory when they are suspended. However, there are a variety of reasons why placing the USB controllers into suspend won't work, but they are beyond the scope of this document. The base requirement is that the USB controller needs to be kept out of main memory, while at the same time, the USB bus is kept from going into suspend.

EHCI controllers provide a large-grained mechanism that can be manipulated by system software to change the memory access pattern of the host controller. System software can manipulate the schedule enable bits in the USBCMD register to turn on/off the scheduling traversal. A software heuristic can be applied to implement an on/off duty cycle that allows the USB to make reasonable progress and allow the Arm platform power management to get the Arm platform into its lowest power state. This method is not intended to be applied at all times to throttle USB, but should only be applied in very specific configurations and usage loads. For example, when only a keyboard or mouse is attached to the USB, the heuristic could detect times when the USB is attempting to move data only very infrequently and can adjust the duty cycle to allow the Arm platform to reach its low power state for longer periods of time. Similarly, it could detect increases in the USB load and adjust the duty cycle appropriately, even to the point where the schedules are never disabled. The assumption here is that the USB is moving data and the Arm platform will be required to process the data streams.

It is suggested that in order to provide a complete solution for the system, the companion host controllers should also provide a similar method to allow system software to inhibit the companion host controller from accessing its shared memory based data structures (schedule lists or otherwise).

54.5.3.14 Port Test Modes -Host Operational Model

EHCI host controllers must implement the port test modes Test J_State, Test K_State, Test_Packet, Test Force_Enable, and Test SE0_NAK as described in the USB Specification Revision 2.0.

The system is only allowed to test ports that are owned by the EHCI controller (for example, *CF-bit* is a one and *PortOwner* bit is a zero). System software is allowed to have at most one port in test mode at a time. Placing more than one port in test mode will yield undefined results. The required, per port test sequence is (assuming the *CF-bit* in the USB_n_CONFIGFLAG register is a one):

- Disable the periodic and asynchronous schedules by setting the *Asynchronous Schedule Enable* and *Periodic Schedule Enable* bits in the USBCMD register to a zero.
- Place all enabled root ports into the suspended state by setting the *Suspend* bit in each appropriate USB_n_PORTSC register to a one.
- Set the *Run/Stop* bit in the USBCMD register to a zero and wait for the *HCHalted* bit in the USBSTS register, to transition to a one. Note that an EHCI host controller implementation may optionally allow port testing with the *Run/Stop* bit set to a one. However, all host controllers must support port testing with *Run/Stop* set to a zero and *HCHalted* set to a one.
- Set the *Port Test Control* field in the port under test PORTSC register to the value corresponding to the desired test mode. If the selected test is Test_Force_Enable, then the *Run/Stop* bit in the USBCMD register must then be transitioned back to one, in order to enable transmission of SOFs out of the port under test.
- When the test is complete, system software must ensure the host controller is halted (*HCHalted* bit is a one) then it terminates and exits test mode by setting *HCReset* to a one.

54.5.3.15 Interrupts-Host Operational Model

The EHCI Host Controller hardware provides interrupt capability based on a number of sources.

There are several general groups of interrupt sources:

- Interrupts as a result of executing transactions from the schedule (success and error conditions),
- Host controller events (Port change events, etc.), and
- Host Controller error events

All transaction-based sources are maskable through the Host Controller's Interrupt Enable register (USBINTR, see Section). Additionally, individual transfer descriptors can be marked to generate an interrupt on completion. This section describes each interrupt source and the processing that occurs in response to the interrupt.

During normal operation, interrupts may be immediate or deferred until the next interrupt threshold occurs. The interrupt threshold is a tunable parameter via the *Interrupt Threshold Control* field in the USBCMD register. The value of this register controls when the host controller will generate an interrupt on behalf of normal transaction execution. When a transaction completes during an interrupt interval period, the interrupt signaling the completion of the transfer will not occur until the interrupt threshold occurs. For example, the default value is eight micro-frames. This means that the host controller will not generate interrupts any more frequently than once every eight micro-frames.

Section Host System Error details effects of a host system error.

If an interrupt has been scheduled to be generated for the current interrupt threshold interval, the interrupt is not signaled until after the status for the last complete transaction in the interval has been written back to host memory. This may sometimes result in the interrupt not being signaled until the next interrupt threshold.

Initial interrupt processing is the same, regardless of the reason for the interrupt. When an interrupt is signaled by the hardware, Arm platform control is transferred to host controller's USB interrupt handler. The precise mechanism to accomplish the transfer is OS specific. For this discussion it is just assumed that control is received. When the interrupt handler receives control, its first action is to reads the USBSTS (USB Status Register). It then acknowledges the interrupt by clearing all of the interrupt status bits by writing ones to these bit positions. The handler then determines whether the interrupt is due to schedule processing or some other event. After acknowledging the interrupt, the handler (via an OS-specific mechanism), schedules a deferred procedure call (DPC) which will execute later. The DPC routine processes the results of the schedule execution. The precise mechanisms used are beyond the scope of this document.

Note: the host controller is not required to de-assert a currently active interrupt condition when software sets the interrupt enables (in the USBINR register, see Section) to a zero. The only reliable method software should use for acknowledging an interrupt is by transitioning the appropriate status bits in the USBSTS register (Section) from a one to a zero.

54.5.3.15.1 Transfer/Transaction Based Interrupts

These interrupt sources are associated with transfer and transaction progress. They are all dependent on the next interrupt threshold.

54.5.3.15.1.1 Transaction Error

A transaction error is any error that caused the host controller to think that the transfer did not complete successfully.

The table below lists the events/responses that the host can observe as a result of a transaction. The effects of the error counter and interrupt status are summarized in the following paragraphs. Most of these errors set the *XactErr* status bit in the appropriate interface data structure.

There is a small set of protocol errors that relate only when executing a queue head and fit under the umbrella of a WRONG PID error that are significant to explicitly identify. When these errors occur, the *XactErr* status bit in the queue head is set and the *CErr* field is decremented. When the *PIDCode* indicates a SETUP, the following responses are protocol errors and result in *XactErr* bit being set to a one and the *CErr* field being decremented.

- *EPS* field indicates a high-speed device and it returns a Nak handshake to a SETUP.
- *EPS* field indicates a high-speed device and it returns a Nyet handshake to a SETUP.
- *EPS* field indicates a low- or full-speed device and the complete-split receives a Nak handshake.

Table 54-54. Summary of Transaction Errors

Event / Result	Queue Head/qTD/iTD/siTD Side-effects		USB Status Register (USBSTS)
	Cerr	Status Field	USBERRINT
CRC	-1	XactErr set to a one.	1[1]
Timeout	-1	XactErr set to a one.	1[1]
Bad PID[2]	-1	XactErr set to a one.	1[1]
Babble	N/A	Section Serial Bus Babble	1
Buffer Error	N/A	Section Data Buffer Error	

1. If occurs in a queue head, then *USBERRINT* is asserted only when *CErr* counts down from a one to a zero. In addition the queue is halted, see Halting a Queue Head.
2. The host controller received a response from the device, but it could not recognize the PID as a valid PID.

54.5.3.15.1.2 Serial Bus Babble

When a device transmits more data on the USB than the host controller is expecting for this transaction, it is defined to be babbling. In general, this is called a *Packet Babble*.

When a device sends more data than the *Maximum Length* number of bytes, the host controller sets the *Babble Detected* bit to a one and halts the endpoint if it is using a queue head (see Halting a Queue Head). *Maximum Length* is defined as the minimum of *Total Bytes to Transfer* and *Maximum Packet Size*. The *CErr* field is not decremented for a packet babble condition (only applies to queue heads). A babble condition also exists if IN transaction is in progress at High-speed EOF2 point. This is called a frame babble. A frame babble condition is recorded into the appropriate schedule data structure. In addition, the host controller must disable the port to which the frame babble is detected.

The *USBERRINT* bit in the USB_n_USBSTS register is set to a one and if the *USB Error Interrupt Enable* bit in the USB_n_USBINTR register is a one, then a hardware interrupt is signaled to the system at the next interrupt threshold. The host controller must never start an OUT transaction that will babble across a micro-frame EOF.

NOTE

When a host controller detects a data PID mismatch, it must either: disable the packet babble checking for the duration of the bus transaction or do packet babble checking based solely on *Maximum Packet Size*. The USB core specification defines the requirements on a data receiver when it receives a data PID mismatch (for example, expects a DATA0 and gets a DATA1 or visa-versa). In summary, it must ignore the received data and respond with an ACK handshake, in order to advance the transmitter's data sequence.

The EHCI interface allows System software to provide buffers for a Control, Bulk or Interrupt IN endpoint that are not an even multiple of the maximum packet size specified by the device. Whenever a device misses an ACK for an IN endpoint, the host and device are out of synchronization with respect to the progress of the data transfer. The host controller may have advanced the transfer to a buffer that is less than maximum packet size. The device will re-send its maximum packet size data packet, with the original data PID, in response to the next IN token. In order to properly manage the bus protocol, the host controller must disable the packet babble check when it observes the data PID mismatch.

54.5.3.15.1.3 Data Buffer Error

This event indicates that an overrun of incoming data or a underrun of outgoing data has occurred for this transaction.

This would generally be caused by the host controller not being able to access required data buffers in memory within necessary latency requirements. These conditions are not considered transaction errors, and do not effect the error count in the queue head. When these errors do occur, the host controller records the fact the error occurred by setting the *Data Buffer Error* bit in the queue head, iTD or siTD.

If the data buffer error occurs on a non-isochronous IN, the host controller will not issue a handshake to the endpoint. This will force the endpoint to resend the same data (and data toggle) in response to the next IN to the endpoint.

If the data buffer error occurs on an OUT, the host controller must corrupt the end of the packet so that it cannot be interpreted by the device as a good data packet. Simply truncating the packet is not considered acceptable. An acceptable implementation option is to 1's complement the CRC bytes and send them. There are other options suggested in the Transaction Translator section of the USB Specification Revision 2.0.

54.5.3.15.1.4 USB Interrupt (Interrupt on Completion (IOC))

Transfer Descriptors (iTDs, siTDs, and queue heads (qTDs)) contain a bit that can be set to cause an interrupt on their completion. The completion of the transfer associated with that schedule item causes the USB Interrupt (USBINT) bit in the USB_n_USBSTS register to be set to a one.

In addition, if a short packet is encountered on an IN transaction associated with a queue head, then this event also causes USBINT to be set to a one. If the USB Interrupt Enable bit in the USB_n_USBINTR register is set to a one, a hardware interrupt is signaled to the system at the next interrupt threshold. If the completion is because of errors, the *USBERRINT* bit in the USB_n_USBSTS register is also set to a one.

54.5.3.15.1.5 Short Packet

Reception of a data packet that is less than the endpoint's Max Packet size during Control, Bulk or Interrupt transfers signals the completion of the transfer. Whenever a short packet completion occurs during a queue head execution, the *USBINT* bit in the USB_n_USBSTS register is set to a one.

If the *USB Interrupt Enable* bit is set in the USB_n_USBINTR register, a hardware interrupt is signaled to the system at the next interrupt threshold.

54.5.3.15.2 Host Controller Event Interrupts

These interrupt sources are independent of the interrupt threshold (with the one exception being the Interrupt on Async Advance, see Section Interrupt on Async Advance).

54.5.3.15.2.1 Port Change Events

Port registers contain status and status change bits. When the status change bits are set to a one, the host controller sets the *Port Change Detect* bit in the USBSTS register to a one.

If the *Port Change Interrupt Enable* bit in the USB_n_USBINTR register is a one, then the host controller will issue a hardware interrupt. The port status change bits include:

- Connect Status Change
- Port Enable/Disable Change
- Over-current Change
- Force Port Resume

54.5.3.15.2.2 Frame List Rollover

This event indicates that the host controller has wrapped the frame list. The current programmed size of the frame list effects how often this interrupt occurs.

If the frame list size is 1024, then the interrupt will occur every 1024 milliseconds, if it is 512, then it will occur every 512 milliseconds, etc. When a frame list rollover is detected, the host controller sets the *Frame List Rollover* bit in the USB.USBSTS register to a one. If the *Frame List Rollover Enable* bit in the USB.USBINTR register is set to a one, the host controller issues a hardware interrupt. This interrupt is not delayed to the next interrupt threshold.

54.5.3.15.2.3 Interrupt on Async Advance

This event is used for deterministic removal of queue heads from the asynchronous schedule. Whenever the host controller advances the on-chip context of the asynchronous schedule, it evaluates the value of the *Interrupt on Async Advance Doorbell* bit in the USB.USBCMD register.

If it is a one, it sets the *Interrupt on Async Advance* bit in the USB.USBSTS register to a one. If the *Interrupton Async Advance Enable* bit in the USB.USBINTR register is a one, the host controller issues a hardware interrupt at the next interrupt threshold. A detailed explanation of this feature is described in Section Removing Queue Heads from Asynchronous Schedule .

54.5.3.15.2.4 Host System Error

The host controller is a bus master and any interaction between the host controller and the system may experience errors.

The type of host error may be catastrophic to the host controller (such as a Master Abort) making it impossible for the host controller to continue in a coherent fashion.In the presence of non-catastrophic host errors, such as parity errors, the host controller could potentially continue operation. The recommended behavior for these types of errors is to escalate it to a catastrophic error and halt the host controller. Host-based error must result in the following actions:

- The *Run/Stop* bit in the USB.USBCMD register is set to a zero.
- The following bits in the USB.USBSTS register are set:
  - *Host System Error* bit is to a one.
  - *HCHalted* bit is set to a one.
- If the *Host System Error Enable* bit in the USB.USBINTR register is a one, then the host controller will issue a hardware interrupt. This interrupt is not delayed to the next interrupt threshold. The following table summarizes the required actions taken on the various host errors.

Table 54-55. Summary Behavior of EHCI Host Controller on Host System Errors

Cycle Type	Master Abort	Target Abort	Data Phase Parity
Frame list pointer fetch (read)	Fatal	Fatal	Fatal [o]
siTD fetch (read)	Fatal	Fatal	Fatal [o]
siTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
iTD fetch (read)	Fatal	Fatal	Fatal [o]
iTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
qTD fetch (read)	Fatal	Fatal	Fatal [o]
qHD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
Data write	Fatal [o]	Fatal [o]	Fatal [o]
Data read	Fatal	Fatal	Fatal [o]

Potentially, a host controller implementation could continue operation without a halt. However, the recommended behavior is to halt the host controller.

NOTE

After a *Host System Error*, Software must reset the host controller through *HCReset* in the USB.USBCMD register before re-initializing and restarting the host controller.

54.5.4 EHCI Deviation

For the purposes a dual-role Host/Device controller with support for On-The-Go applications, it is necessary to deviate from the EHCI specification. Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November

2000, Intel Corporation. http://www.intel.com. Device operation & On-The-Go operation is not specified in the EHCI and thus the implementation supported in this core is proprietary.

The host mode operation of the core is near EHCI compatible with few minor differences documented in this section.

The particulars of the deviations occur in the areas summarized here:

- Embedded Transaction Translator - Allows direct attachment of FS and LS devices in host mode without the need for a companion controller.
- Device operation - In host mode the device operational registers are generally disabled and thus device mode is mostly transparent when in host mode. However, there are a couple exceptions documented in the following sections.
- Embedded design interface - This core does not have a PCI Interface and therefore the PCI configuration registers described in the EHCI specification are not applicable.
- On-The-Go Operation - This design includes an On-The-Go controller for Port #1.

54.5.4.1 Embedded Transaction Translator Function

The OTG controller supports directly connected full and low speed devices without requiring a companion controller by including the capabilities of a USB 2.0 high speed hub transaction translator.

Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a high speed hub has been implemented within the DMA and Protocol engine blocks. The embedded transaction translator function is an extension to EHCI interface, but makes use of the standard data structures and operational models that exist in the EHCI specification to support full and low speed devices.

54.5.4.1.1 Capability Registers

The following additions have been added to the capability registers to support the embedded Transaction Translator Function:

- N_TT added to USB.HCSPARAMS - Host Control Structural Parameters
- N_PTT added to USB.HCSPARAMS - Host Control Structural Parameters

54.5.4.1.2 Operational Registers

The following additions have been added to the operational registers to support the embedded TT:

- Addition of two-bit Port Speed (PSPD) to the register.

54.5.4.1.3 Discovery-EHCI Deviation

In a standard EHCI controller design, the EHCI host controller driver detects a Full speed (FS) or Low speed (LS) device by noting if the port enable bit is set after the port reset operation.

The port enable will only be set in a standard EHCI controller implementation after the port reset operation and when the host and device negotiate a High-Speed connection (that is, Chirp completes successfully).

Because this controller has an embedded Transaction Translator, the port enable will always be set after the port reset operation regardless of the result of the host device chirp result and the resulting port speed will be indicated by the PSPD field in USB.PORTSCx.

Therefore, the standard EHCI host controller driver requires an alteration to handle directly connected Full and Low speed devices or hubs.

The change is a fundamental one in that is summarized in the following table.

Table 54-56. Summary of EHCI

Standard EHCI	EHCI with embedded Transaction Translator
After port enable bit is set following a connection and reset sequence, the device/hub is assumed to be HS.	After port enable bit is set following a connection and reset sequence, the device/hub speed is noted from USB.PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub thus, all port-level control is performed through the Hub Class to the nearest Hub.	FS and LS device can be either downstream from a HS hub or directly attached. When the FS/LS device is downstream from a HS hub, then port-level control is done using the Hub Class through the nearest Hub. When a FS/LS device is directly attached, then port-level control is accomplished using USB.PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub with HubAddr=X. [where HubAddr > 0 and HubAddr is the address of the Hub where the bus transitions from HS to FS/LS (ie. Split target hub)]	FS and LS device can be either downstream from a HS hub with HubAddr = X [HubAddr > 0] or directly attached [where HubAddr = 0 and HubAddr is the address of the Root Hub where the bus transitions from HS to FS/LS (ie. Split target hub is the root hub)]

54.5.4.1.4 Data Structures

The same data structures used for FS/LS transactions though a HS hub are also used for transactions through the Root Hub with sm embedded Transaction Translator.

Here it is demonstrated how the Hub Address and Endpoint Speed fields should be set for directly attached FS/LS devices and hubs:

1. QH (for direct attach FS/LS) - Async. (Bulk/Control Endpoints) Periodic (Interrupt)
   - Hub Address = 0
   - Transactions to direct attached device/hub.
     - QH.EPS = Port Speed
   - Transactions to a device downstream from direct attached FS hub.
     - QH.EPS = Downstream Device Speed

NOTE

When QH.EPS = 01 (LS) and PORTSCx.PSPD = 00 (FS), a LS-pre-pid will be sent before the transmitting LS traffic.

Maximum Packet Size must be less than or equal 64 or undefined behaviour may result.

2. siTD (for direct attach FS) - Periodic (ISO Endpoint)
   - All FS ISO transactions:
     - Hub Address = 0
     - siTD.EPS = 00 (full speed)
       - Maximum Packet Size must less than or equal to 1023 or undefined behaviour may result.

54.5.4.1.5 Operational Model

The operational models are well defined for the behavior of the Transaction Translator (see USB 2.0 specification. Universal Serial Bus Specification, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. http://www.usb.org) and for the EHCI controller moving packets between system memory and a USB-HS hub. Because the embedded Transaction Translator exists within the host controller there is no physical bus between EHCI host controller driver and the USB FS/LS bus. These sections will briefly discuss the operational model for how the EHCI and Transaction Translator operational models are combined without the physical bus between. The following sections assume the reader is familiar with both the EHCI and USB 2.0 Transaction Translator operational models.

54.5.4.1.5.1 Micro- frame Pipeline

The EHCI operational model uses the concept of H-frames and B-frames to describe the pipeline between the Host (H) and the Bus (B). The embedded Transaction Translator shall use the same pipeline algorithms specified in the USB 2.0 specification for a Hub-based Transaction Translator.

All periodic transfers always begin at B-frame 0 (after SOF) and continue until the stored periodic transfers are complete. As an example of the micro-frame pipeline implemented in the embedded Transaction Translator, all periodic transfers that are tagged in EHCI to execute in H-frame 0 will be ready to execute on the bus in B-frame 0.

It is important to note that when programming the S-mask and C-masks in the EHCI data structures to schedule periodic transfers for the embedded Transaction Translator, the EHCI host controller driver must follow the same rules specified in EHCI for programming the S-mask and C-mask for downstream Hub-based Transaction Translators.

Once periodic transfers are exhausted, any stored asynchronous transfer will be moved. Asynchronous transfers are opportunistic in that they shall execute whenever possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer can not babble through the SOF (start of B-frame 0.)

54.5.4.1.5.2 Split State Machines

The start and complete split operational model differs from EHCI slightly because there is no bus medium between the EHCI controller and the embedded Transaction Translator.

Where a start or complete-split operation would occur by requesting the split to the HS hub, the start/complete split operation is simple an internal operation to the embedded Transaction Translator. The following table summarizes the conditions where handshakes are emulated from internal state instead of actual handshakes to HS split bus traffic.

Table 54-57. Summary of the Conditons of Handshakes[1]

Condition	Emulate TT Response
Start-Split: All asynchronous buffers full.	NAK
Start-Split: All periodic buffers full.	ERR
Start-Split: Success for start of Async. Transaction.	ACK
Start-Split: Start Periodic Transaction.	No Handshake (Ok)
Complete-Split: Failed to find transaction in queue.	Bus Time Out
Complete-Split: Transaction in Queue is Busy.	NYET
Complete-Split: Transaction in Queue is Complete.	[Actual Handshake from LS/FS device]

1. The un-shaded cells represent Start-Splits and the shaded cells represent Complete-Splits

54.5.4.1.5.3 Asynchronous Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.
- Transaction tracking for 2 data pipes.

54.5.4.1.5.3.1 USB 2.0 - 11.17.3

- Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.

54.5.4.1.5.3.2 USB 2.0 - 11.17.4

- Transaction tracking for 2 data pipes.

54.5.4.1.5.4 Periodic Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- Abort of pending start-splits
  - EOF (and not started in micro-frames 6)
  - Idle for more than 4 micro-frames
- Abort of pending complete-splits
  - EOF
  - Idle for more than 4 micro-frames
- Transaction tracking for up to 16 data pipes.
- Complete-split transaction searching.

NOTE

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

54.5.4.1.5.4.1 USB 2.0 - 11.18.6.[1-2]

- Abort of pending start-splits
  - EOF (and not started in micro-frames 6)
  - Idle for more than 4 micro-frames
- Abort of pending complete-splits
  - EOF
  - Idle for more than 4 micro-frames

54.5.4.1.5.4.2 USB 2.0 - 11.18.[7-8]

- Transaction tracking for up to 16 data pipes.
- Complete-split transaction searching.

NOTE

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

54.5.4.1.5.5 Multiple Transaction Translators

The maximum number of embedded Transaction Translators that is currently supported is one as indicated by the N_TT field in the register.

54.5.4.2 Device Operation

The co-existence of a device operational controller within the host controller has little effect on EHCI compatibility for host operation except as noted in this section.

54.5.4.2.1 USB_USBMODE Register

Given that the dual-role controller is initialized in neither host nor device mode, the USB Device Mode (USB_*n*USBMODE) register must be programmed for host operation before the EHCI host controller driver can begin EHCI host operations.

54.5.4.2.2 Non-Zero Fields the Register File

Some of the reserved fields and reserved addresses in the capability registers and operational register have use in device mode, the following must be adhered to:

- Write operations to all EHCI reserved fields (some of which are device fields) with the operation registers should always be written to zero. This is an EHCI requirement of the device controller driver that must be adhered to.
- Read operations by the host controller must properly mask EHCI reserved fields (some of which are device fields) because fields that are used exclusive for device are undefined in host mode .

54.5.4.2.3 SOF Interrupt

This SOF Interrupt used for device mode is shared as a free running 125us interrupt for host mode.

EHCI does not specify this interrupt but it has been added for convenience and as a potential software time base. See USB Status Register (USB_*n*USBSTS) and Interrupt Enable Register (USB_*n*USBINTR) registers.

54.5.4.3 Embedded Design Interface

This is an Embedded USB Host Controller as defined by the EHCI specification and thus does not implement the PCI configuration registers.

54.5.4.3.1 Frame Adjust Register

Given that the optional PCI configuration registers are not included in this implementation, there is no corresponding bit level timing adjustments like is provided by the Frame Adjust register in the PCI configuration registers. Starts of micro-frames are timed precisely to 125 us using the transceiver clock as a reference clock. That is, a 60 Mhz transceiver clock for 8-bit physical interfaces & full-speed serial interfaces or 30 Mhz transceiver clock for 16-bit physical interfaces.

54.5.4.4 Miscellaneous variations from EHCI

54.5.4.4.1 Programmable Physical Interface Behaviour

This design supports multiple Physical interfaces which can operate in differing modes when the core is configured with software programmable Physical Interface Modes. Software programmability allows the selection of the Physical interface part during the board design phase instead of during the chip design phase. The control bits for selecting the Physical Interface operating mode have been added to the Port Status & Control (USB_*n*PORTSC1) register providing a capability that is not defined by EHCI.

54.5.4.4.2 Discovery

54.5.4.4.2.1 Port Reset

The port connect methods specified by EHCI require setting the port reset bit in the Port Status & Control (USB_*n*PORTSC1) register for a duration of 10ms. Due to the complexity required to support the attachment of devices that are not high speed there are counter already present in the design that can count the 10ms reset pulse to alleviate the requirement of the software to measure this duration. Therefore, the basic connection is then summarized as the following:

- [Port Change Interrupt] Port connect change occurs to notify the host controller driver that a device has attached.
- Software shall write a '1' to reset the device.

- Software shall write a '0' to reset the device after 10 ms.
  - This step, which is necessary in a standard EHCI design, may be omitted with this implementation. Should the EHCI host controller driver attempt to write a '0' to the reset bit while a reset is in progress, the write will simple be ignored and the reset will continue until completion.
- [Port Change Interrupt] Port enable change occurs to notify the host controller that the device in now operational and at this point the port speed has been determined.

54.5.4.4.2.2 Port Speed Detection

After the port change interrupt indicates that a port is enabled, the EHCI stack should determine the port speed. Unlike the EHCI implementation, which will re-assign the port owner for any device that does not connect at High-Speed, this host controller supports direct attach of non High-Speed devices.

Therefore, the following differences are important regarding port speed detection:

- Port Owner is read-only and always reads 0.
- A 2-bit Port Speed indicator has been added to PORTSC to provide the current operating speed of the port to the host controller driver.
- A 1-bit High Speed indicator has been added to PORTSC to signify that the port is in High-Speed vs. Full/Low Speed - *This information is redundant with the 2-bit Port Speed indicator above.*

54.5.4.4.3 Port Test Mode

Port Test Control mode behaves fully as described in EHCI. An alternate host controller driver procedure is not necessary or supported.

54.5.5 Device Data Structures

This section defines the interface data structures used to communicate control, status, and data between Device Controller Driver (DCD) Software and the Device Controller.

The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of device Queue Heads and Transfer Descriptors.

NOTE

Software must ensure that no interface data structure reachable by the Device Controller spans a 4K-page boundary.

The data structures defined in the chapter are (from the device controller's perspective) a mix of read-only and read/ writable fields. The device controller must preserve the read-only fields on all data structure writes.

The figure below shows the organization of the EndPoint Queue Head.

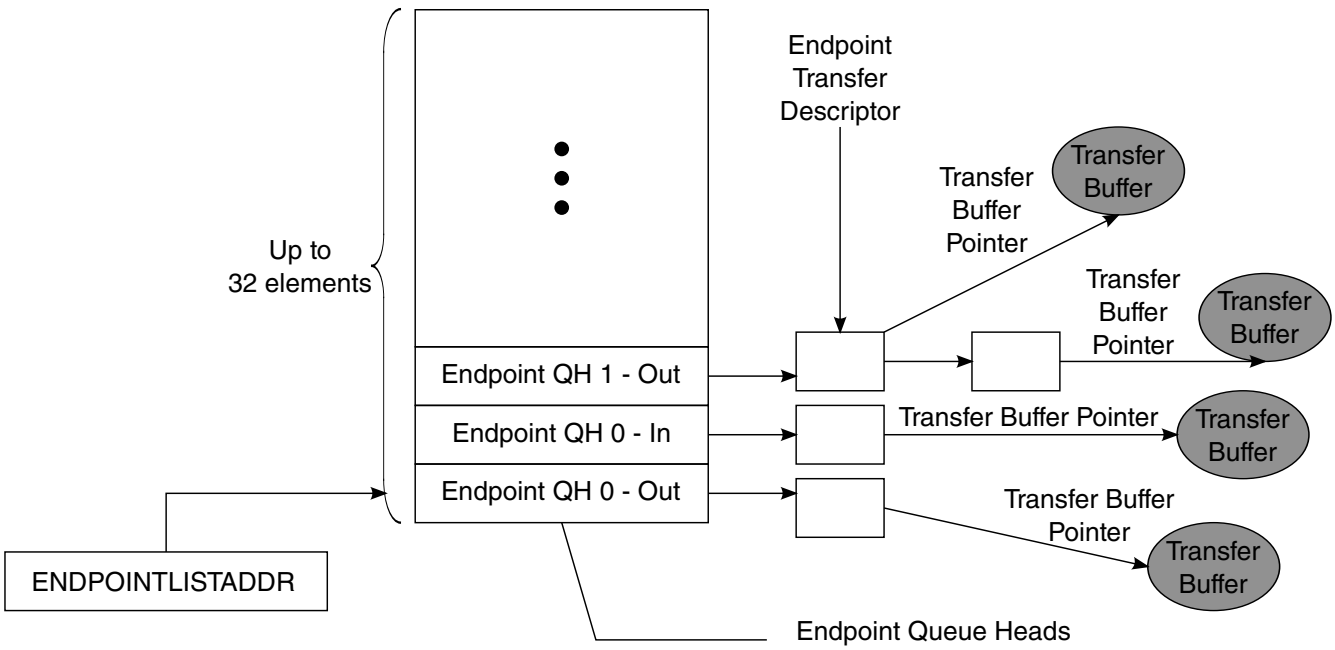


Figure 54-27. EndPoint Queue Head Organization

Endpoint queue heads are arranged in an array in a continuous area of memory pointed to by the USB.ENDPOINTLISTADDR pointer. The even -numbered device queue heads in the list support receive endpoints (OUT/SETUP) and the odd-numbered queue heads in the list are used for transmit endpoints (IN/INTERRUPT). The device controller will index into this array based upon the endpoint number received from the USB bus. All information necessary to respond to transactions for all primed transfers is contained in this list so the Device Controller can readily respond to incoming requests without having to traverse a linked list.

NOTE

The Endpoint Queue Head List must be aligned to a 2k boundary.

54.5.5.1 Endpoint Queue Head (dQH)

The device Endpoint Queue Head (dQH) is where all transfers for a given endpoint are managed. The dQH is a 48-byte data structure, but must be aligned on 64-byte boundaries.

During priming of an endpoint, the dTD (device transfer descriptor) is copied into the overlay area of the dQH, which starts at the nextTD pointer DWord and continues through the end of the buffer pointers DWords. After a transfer is complete, the dTD status DWord is updated in the dTD pointed to by the currentTD pointer. While a packet is in progress, the overlay area of the dQH is used as a staging area for the dTD so that the Device Controller can access needed information with little minimal latency.

Table 54-58. Endpoint Queue Head (dQH)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mult		zlt	0		Maximum Packet Length											ios	0														
Current dTD Pointer																											0				
Next dTD Pointer																											0				T[1]
0	Total Bytes															ioc	0			MultO		0		Status							
Buffer Pointer (Page 0)																				Current Offset											
Buffer Pointer (Page 1)																				Reserved											
Buffer Pointer (Page 2)																				Reserved											
Buffer Pointer (Page 3)																				Reserved											
Buffer Pointer (Page 4)[1]																				Reserved											
Reserved																															
Set-up Buffer Bytes 3…0																															
Set-up Buffer Bytes 7…4																															

1. Transfer overlay starts at T and continues through Buffer Pointer (Page 4).

Host Controller Read/Write

Host Controller Read Only

54.5.5.1.1 Endpoint Capabilities/Characteristics

This DWord specifies static information about the endpoint, in other words, this information does not change over the lifetime of the endpoint. Device Controller software should not attempt to modify this information while the corresponding endpoint is enabled.

Table 54-59 describes the endpoint capabilities.

Table 54-59. Endpoint Capabilities/Characteristics

Bit	Description
31-30	Mult. This field is used to indicate the number of packets executed per transaction description as given by the following: 00 - Execute N Transactions as demonstrated by the USB variable length packet protocol where N is computed using the Maximum Packet Length (dQH) and the Total Bytes field (dTD) 01 Execute 1 Transaction. 10 Execute 2 Transactions. 11 Execute 3 Transactions. **NOTE:** Non-ISO endpoints must set Mult="00". ISO endpoints must set Mult="01", "10", or "11" as needed.
29	Zero Length Termination Select. This bit is used to indicate when a zero length packet is used to terminate transfers where to total transfer length is a multiple . This bit is not relevant for Isochronous 0 - Enable zero length packet to terminate transfers equal to a multiple of the Maximum Packet Length. (default). 1 - Disable the zero length packet on transfers that are equal in length to a multiple Maximum Packet Length.
28-27	Reserved. These bit reserved for future use and should be set to zero.
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). The maximum value this field may contain is 0x400 (1024).
15	Interrupt On Setup (IOS). This bit is used on control type endpoints to indicate if USBINT is set in response to a setup being received.
14-0	Reserved. Bits reserved for future use and should be set to zero.

54.5.5.1.2 Transfer Overlay-Endpoint Queue Head

The seven DWords in the overlay area represent a transaction working space for the device controller.

The general operational model is that the device controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it will not read the associated endpoint.

After an endpoint is readied, the dTD will be copied into this queue head overlay area by the device controller. Until a transfer is expired, software must not write the queue head overlay area or the associated transfer descriptor. When the transfer is complete, the device controller will write the results back to the original transfer descriptor and advance the queue.

See dTD for a description of the overlay fields.

54.5.5.1.3 Current dTD Pointer

The current dTD pointer is used by the device controller to locate the transfer in progress. This word is for Device Controller (hardware) use only and should not be modified by DCD software.

The following table describes the dTD Pointer.

Table 54-60. Next dTD Pointer

Bit	Description
31-5	Current dTD. This field is a pointer to the dTD that is represented in the transfer overlay area. This field will be modified by the Device Controller to next dTD pointer during endpoint priming or queue advance.
4-0	Reserved. Bit reserved for future use and should be set to zero.

54.5.5.1.4 Set-up Buffer

The set-up buffer is dedicated storage for the 8-byte data that follows a set-up PID.

NOTE

Each endpoint has a TX and an RX dQH associated with it, and only the RX queue head is used for receiving setup data packets.

The following table describes the Multiple Mode Control.

Table 54-61. Multiple Mode Control (HCCPARAMS)

DWord	Bits	Description
1	31-0	Setup Buffer 0. This buffer contains bytes 3 to 0 of an incoming setup buffer packet and is written by the device controller to be read by software.
2	31-0	Setup Buffer 1. This buffer contains bytes 7 to 4 of an incoming setup buffer packet and is written by the device controller to be read by software.

54.5.5.2 Endpoint Transfer Descriptor (dTD)

The dTD describes to the device controller the location and quantity of data to be sent/ received for a given transfer.

The DCD should not attempt to modify any field in an active dTD except the Next Like Pointer, which should only be modified as described in section Managing Transfers with Transfer Descriptors.

Table below shows the Endpoint Transfer Descriptor (dTD).

Table 54-62. Endpoint Transfer Descriptor (dTD)

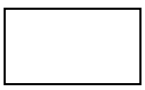
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Next Link Pointer																											0				T
0	Total Bytes															ioc	0			MultO		0		Status							
Buffer Pointer (Page 0)																				Current Offset											

Table continues on the next page...

Table 54-62. Endpoint Transfer Descriptor (dTD) (continued)

Buffer Pointer (Page 1)	0	Frame Number
Buffer Pointer (Page 2)	Reserved	
Buffer Pointer (Page 3)	Reserved	
Buffer Pointer (Page 4)	Reserved	

Host Controller Read/Write



Host Controller Read Only

The following table describes the dTD Pointer.

Table 54-63. Next dTD Pointer

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next dTD to be processed. The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved. Bits reserved for future use and should be set to zero.
0	Terminate (T). 1=pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Device Controller that there are no more valid entries in the queue.

The following table describes the dTD Token.

Table 54-64. dTD Token

Bit	Description
31	Reserved. Bit reserved for future use and should be set to zero.
30-16	Total Bytes. This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction and only on the successful completion of the transaction. The maximum value software may store in the field is 5*4K (5000H). This is the maximum number of bytes 5 page pointers can access. Although it is possible to create a transfer up to 20K this assumes the 1st offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16K (4000H). If the value of the field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the device controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for IN transfers that Total Bytes To Transfer be an even multiple of *Maximum Packet Length*. If software builds such a transfer descriptor for an IN transfer, the last transaction will always be less that *Maximum Packet Length*.
15	Interrupt On Complete (IOC). This bit is used to indicate if USBINT is to be set in response to device controller being finished with this dTD.
14-12	Reserved. Bits reserved for future use and should be set to zero.
11-10	Multiplier Override (MultO). This field can be used for transmit ISO's (ie. ISO-IN) to override the multiplier in the QH. This field must be zero for all packet types that are not transmit-ISO. Example:

Table continues on the next page...

Table 54-64. dTD Token (continued)

	if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 0 [default] Three packets are sent: {Data2(8); Data1(7); Data0(0)} if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 2 Two packets are sent: {Data1(8); Data0(7)} For maximal efficiency, software should compute MultO = greatest integer of (Total Bytes / Max. Packet Size) except for the case when Total Bytes = 0; then MultO should be 1. Note: Non-ISO and Non-TX endpoints must set MultO = "00".
9-8	Reserved. Bits reserved for future use and should be set to zero.
7-0	Status. This field is used by the Device Controller to communicate individual command execution states back to the Device Controller software. This field contains the status of the last transaction performed on this qTD. The bit encodings are: Bit Status Field Description 7 Active. 6 Halted. 5 Data Buffer Error. 3 Transaction Error. 4, 2, 0 Reserved.

The table below describes the dTD Buffer Page Pointer List.

Table 54-65. dTD Buffer Page Pointer List

Bit	Description
31-12	Buffer Pointer. Selects the page offset in memory for the packet buffer. Non virtual memory systems will typically set the buffer pointers to a series of incrementing integers.
0,11-0	Current Offset. Offset into the 4kb buffer where the packet is to begin.
1,10-0	Frame Number. Written by the device controller to indicate the frame number in which a packet finishes. This is typically be used to correlate relative completion times of packets on an ISO endpoint.

54.5.6 Device Operational Model

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus.

Using a set of linked list transfer descriptors, pointed to by a queue head, the device controller will perform the data transfers. The following sections explain the use of the device controller from the device controller driver (DCD) point-of-view and further describe how specific USB bus events relate to status changes in the device controller programmer's interface.

54.5.6.1 Device Controller Initialization

After hardware reset, the device is disabled until the Run/Stop bit is set to a '1'. In the disabled state, the pull-up on the USB D+ is not active which prevents an attach event from occurring. At a minimum, it is necessary to have the queue heads setup for endpoint zero before the device attach occurs.

Shortly after the device is enabled, a USB reset will occur followed by setup packet arriving at endpoint 0. A Queue head must be prepared so that the device controller can store the incoming setup packet.

In order to initialize a device, the software should perform the following steps:

- Set Controller Mode in the USB.USBMODE register to device mode.

NOTE

Transitioning from host mode to device mode requires a device controller reset before modifying USB.USBMODE.

- Allocate and Initialize device queue heads in system memory.
  - Minimum: Initialize device queue heads 0 Tx & 0 Rx.

NOTE

All device queue heads for control endpoints must be initialized before the endpoint is enabled. Non-Control device queue heads before the endpoint can be used.

  - For information on device queue heads, refer to section Device Data Structures.
- Configure USB.ENDPOINTLISTADDR Pointer.
  - For additional information on USB.ENDPOINTLISTADDR, refer to the register table.
- Enable the microprocessor interrupt associated with the USB core.
  - Recommended: enable all device interrupts including: USBINT, USBERRINT, Port Change Detect, USB Reset Received, DCSuspend.
  - For a list of available interrupts refer to the Interrupt Enable Register (USB_*n*USBINTR) and the USB Status Register (USB_*n*USBSTS) register tables.
- Set Run/Stop bit to Run Mode.
  - After the Run bit is set and the device is connected to a host, a Bus Reset will be issued by host downstream port. The DCD must monitor the reset event and adjust the software state as described in the Bus Reset section of the Port State and Control section below.

NOTE

Endpoint 0 is designed as a control endpoint only and does not need to be configured using ENDPTCTRL0 register.

It is also not necessary to prime Endpoint 0 initially because the first packet received will always be a setup packet. The contents of the first setup packet will require a response in accordance with USB device framework (Chapter 9) command set.

54.5.6.2 Port State and Control

From a chip or system reset, the device controller enters the *powered* state. A transition from the *powered* state to the *attach* state occurs when the Run/Stop bit is set to a '1'.

After receiving a reset on the bus, the port will enter the *defaultFS or defaultHS* state in accordance with the reset protocol described in Appendix C.2 of the USB Specification Rev. 2.0.

The following state diagram depicts the state of a USB 2.0 device.

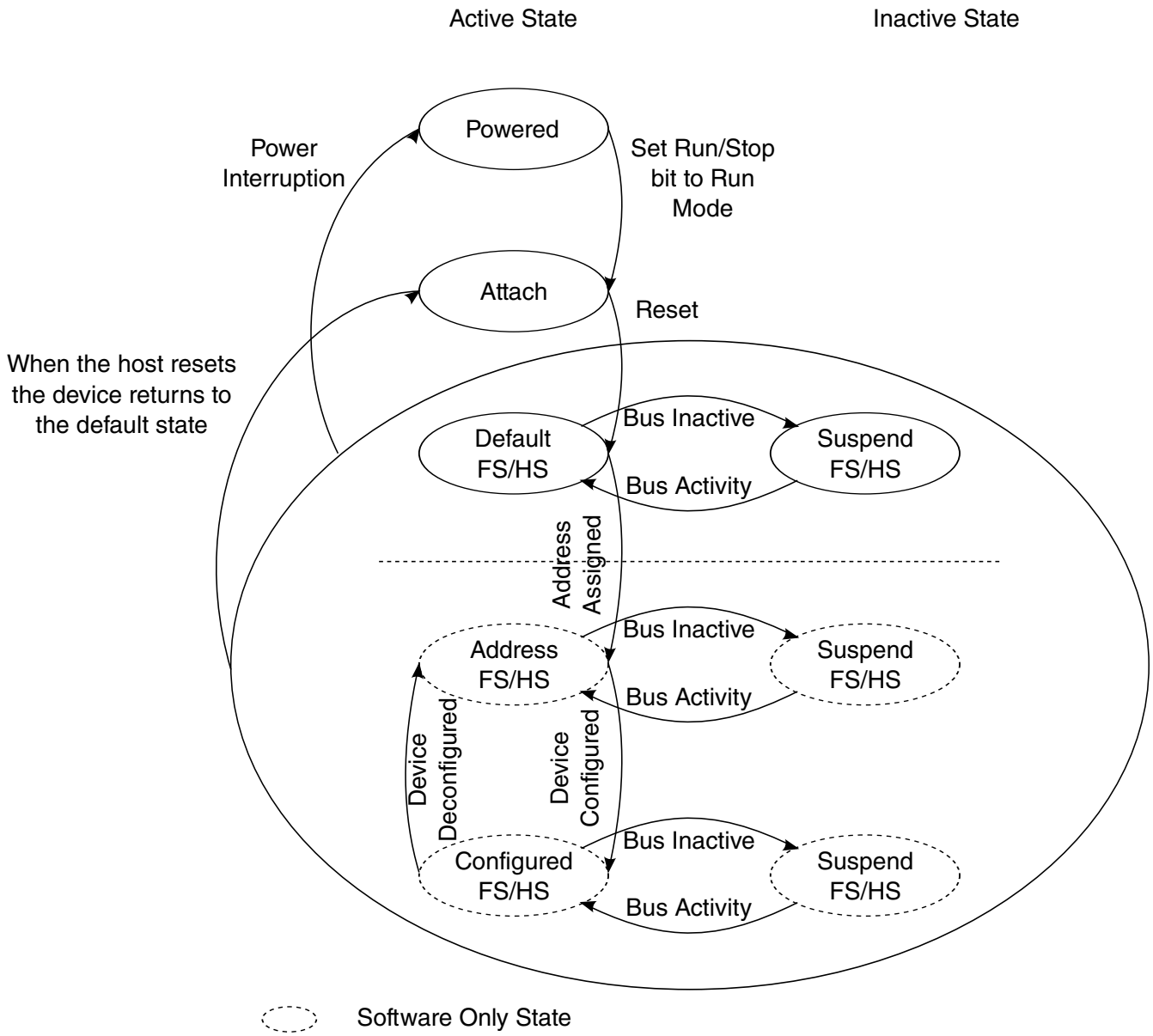


Figure 54-28. Device State Diagram

States *powered, attach, defaultFS/HS, suspendFS/HS* are implemented in the device controller and are communicated to the DCD using the following status bits:

The following table describes the Device Controller State Information Bits.

Table 54-66. Device Controller State Information Bits

Bit	Register
DCSuspend	
USB Reset Received	
Port Change Detect	
High-Speed Port	

It is the responsibility of the DCD to maintain a state variable to differentiate between the *DefaultFS/HS* state and the *Address/Configured* states. Change of state from *Default* to *Address* and the *Configured* states is part of the enumeration process described in the device framework section of the USB 2.0 Specification.

As a result of entering the *Address* state, the device address register (DEVICEADDR) must be programmed by the DCD.

Entry into the *Configured* indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the USB_UOG_ENDPTCTRLx registers and initializing the associated queue heads.

54.5.6.2.1 Bus Reset

A bus reset is used by the host to initialize downstream devices.

When a bus reset is detected, the device controller will renegotiate its attachment speed, reset the device address to 0, and notify the DCD by interrupt (assuming the USB Reset Interrupt Enable is set). After a reset is received, all endpoints (except endpoint 0) are disabled and any primed transactions will be cancelled by the device controller. The concept of priming will be clarified below, but the DCD must perform the following tasks when a reset is received:

Clear all setup token semaphores by reading the Endpoint Status (USB_*n*ENDPTSTAT) register and writing the same value back to the Endpoint Status (USB_*n*ENDPTSTAT) register.

Clear all the endpoint complete status bits by reading the Endpoint Complete (USB_*n*ENDPTCOMPLETE) register and writing the same value back to the Endpoint Complete (USB_*n*ENDPTCOMPLETE) register.

Cancel all primed status by waiting until all bits in the Endpoint Prime (USB_*n*ENDPTPRIME) are 0 and then writing 0xFFFFFFFF to Endpoint Flush (USB_*n*ENDPTFLUSH).

Read the reset bit in the Port Status & Control (USB_*n*PORTSC1) register and make sure that it is still active. A USB reset will occur for a minimum of 3 ms and the DCD must reach this point in the reset cleanup before end of the reset occurs, otherwise a hardware reset of the device controller is recommended (rare.)

- A hardware reset can be performed by writing a one to the device controller reset bit in the USBCMD reset. Note: a hardware reset will cause the device to detach from the bus by clearing the Run/Stop bit. Thus, the DCD must completely re-initialize the device controller after a hardware reset.

Free all allocated dTDs because they will no longer be executed by the device controller. If this is the first time the DCD is processing a USB reset event, then it is likely that no dTDs have been allocated.

At this time, the DCD may release control back to the OS because no further changes to the device controller are permitted until a Port Change Detect is indicated.

After a Port Change Detect, the device has reached the default state and the DCD can read the Port Status & Control (USB_*n*PORTSC1) to determine if the device is operating in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration according to the USB Chapter 9 - Device Framework.

NOTE

The device DCD may use the FS/HS mode information to determine the bandwidth mode of the device

In some applications, it may not be possible to enable one or more pipes while in FS mode. *Beyond the data rate issue, there is no difference in DCD operation between FS and HS modes.*

54.5.6.2.2 Suspend/Resume

The detials of suspend and resume are explained in these sections.

54.5.6.2.2.1 Suspend

Suspend Description

In order to conserve power, USB devices automatically enter the suspended state when the device has observed no bus traffic for a specified period. When suspended, the USB device maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend at any time they are powered, regardless of if they have been assigned a non-default address, are configured, or neither. Bus activity may cease due to the host entering a suspend mode of its own. In addition, a USB device shall also enter the suspended state when the hub port it is attached to is disabled.

A USB device exits suspend mode when there is bus activity. A USB device may also request the host to exit suspend mode or selective suspend by using electrical signaling to indicate remote wakeup. The ability of a device to signal remote wakeup is optional. If the USB device is capable of remote wakeup signaling, the device must support the ability of the host to enable and disable this capability. When the device is reset, remote wakeup signaling must be disabled.

Suspend Operational Model

The device controller moves into the suspend state when suspend signaling is detected or activity is missing on the upstream port for more than a specific period. After the device controller enters the suspend state, the DCD is notified by an interrupt (assuming *DC Suspend Interrupt* is enabled). When the *DCSuspend* bit in the Port Status & Control (USB_*n*PORTSC1) is set to a '1', the device controller is suspended.

DCD response when the device controller is suspended is application specific and may involve switching to low power operation.

Information on the bus power limits in suspend state can be found in USB 2.0 specification.

NOTE

Review system level clocking issues defined in section (Ref: Signals-Clocking) for the clocking requirements of a suspended device controller.

54.5.6.2.2.2 Resume

If the device controller is suspended, its operation is resumed when any non-idle signaling is received on its upstream facing port. In addition, the device can signal the system to resume operation by forcing resume signaling to the upstream port.

Resume signaling is sent upstream by writing a '1' to the Resume bit in the in the Port Status & Control (USB_*n*PORTSC1) while the device is in suspend state. Sending resume signal to an upstream port should cause the host to issue resume signaling and bring the suspended bus segment (one more devices) back to the active condition.

NOTE

Before resume signaling can be used, the host must enable it by using the Set Feature command defined in device framework (chapter 9) of the USB 2.0 Specification.

54.5.6.3 Managing Endpoints

The USB 2.0 specification defines an endpoint, also called a device endpoint or an address endpoint as a uniquely addressable portion of a USB device that can source or sink data in a communications channel between the host and the device.

The endpoint address is specified by the combination of the endpoint number and the endpoint direction.

The channel between the host and an endpoint at a specific device represents a data pipe. Endpoint 0 for a device is always a *control* type data channel used for device discovery and enumeration. Other types of endpoints support by USB include *bulk, interrupt*, and *isochronous*. Each endpoint type has specific behavior related to packet response and error handling. More detail on endpoint operation can be found in the USB 2.0 specification.

The USB OTG device controller hardware supports up to 8 endpoint numbers.

Each endpoint direction is essentially independent and can be configured with differing behavior in each direction. For example, the DCD can configure endpoint 1-IN to be a bulk endpoint and endpoint 1-OUT to be an isochronous endpoint. This helps to conserve the total number of endpoints required for device operation. The only exception is that control endpoints must use both directions on a single endpoint number to function as a control endpoint. Endpoint 0 is, for example, is always a control endpoint and uses the pair of directions.

Each endpoint direction requires a *queue head* allocated in memory. To support the 8 endpoint numbers, 16 *queue heads* are required. The operation of an endpoint and use of *queue heads* are described later in this document.

54.5.6.3.1 Endpoint Initialization

After hardware reset, all endpoints except endpoint zero are uninitialized and disabled. The DCD must configure and enable each endpoint by writing to configuration bit in the USB_UOG_ENDPTCTRLx register.

Each 32-bit USB_UOG_ENDPTCTRLx is split into an upper and lower half. The lower half of USB_UOG_ENDPTCTRLx is used to configure the receive or OUT endpoint and the upper half is likewise used to configure the corresponding transmit or IN endpoint. Control endpoints must be configured the same in both the upper and lower half of the USB_UOG_ENDPTCTRLx register otherwise the behavior is undefined. The following table shows how to construct a configuration word for endpoint initialization. The following table shows the fields and values for the Device Controller Endpoint initialization.

Table 54-67. Device Controller Endpoint Initialization

Field	Value
Data Toggle Reset	1
Data Toggle Inhibit	0
Endpoint Type	00 Control 01 Isochronous 10 Bulk

Table continues on the next page...

Table 54-67. Device Controller Endpoint Initialization (continued)

	11 Interrupt
Endpoint Stall	0

54.5.6.3.2 Stalling

There are two occasions where the device controller may need to return to the host a STALL.

The first occasion is the functional stall, which is a condition set by the DCD as described in the USB 2.0 device framework. A functional stall is only used on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the USB_UOG_ENDPTCTRLx register associated with the given endpoint and the given direction. In a functional stall condition, the device controller will continue to return STALL responses to all transactions occurring on the respective endpoint and direction until the endpoint stall bit is cleared by the DCD.

A protocol stall, unlike a function stall, is used on control endpoints is automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, the DCD should enable the stall bits (both directions) as a pair. A single write to the USB_UOG_ENDPTCTRLx register can ensure that both stall bits are set at the same instant.

NOTE

Any write to the USB_UOG_ENDPTCTRLx register during operational mode must preserve the endpoint type field (that is, perform a read-modify-write).

The following table shows the response matrix for the Device Controller Stall.

Table 54-68. Device Controller Stall Response Matrix

USB Packet	Endpoint Stall Bit.	Effect on STALL bit.	USB Response
SETUP packet received by a non-control endpoint.	N/A	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'1'	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'0'	None.	ACK/ NAK/ NYET
SETUP packet received by a control endpoint.	N/A	Cleared	ACK
IN/OUT/PING packet received by a control endpoint	'1'	None	STALL

Table continues on the next page...

Table 54-68. Device Controller Stall Response Matrix (continued)

IN/OUT/PING packet received by a control endpoint.	'0'	None.	ACK/ NAK/ NYET

54.5.6.3.3 Data Toggle

Data toggle is a mechanism to maintain data coherency between host and device for any given data pipe.

For more information on data toggle, refer to the USB 2.0 specification.

54.5.6.3.3.1 Data Toggle Reset

The DCD may reset the data toggle state bit and cause the data toggle sequence to reset in the device controller by writing a '1' to the data toggle reset bit in the USB_UOG_ENDPTCTRLx register.

This should only be necessary when configuring/initializing an endpoint or returning from a STALL condition.

54.5.6.3.3.2 Data Toggle Inhibit

NOTE

This feature is for test purposes only and should never be used during normal device controller operation.

Setting the *data toggle Inhibit bit* active ('1') causes the device controller to ignore the data toggle pattern that is normally sent and accept all incoming data packets regardless of the data toggle state.

In normal operation, the device controller checks the DATA0/DATA1 bit against the data toggle to determine if the packet is valid. If Data PID does not match the data toggle state bit maintained by the device controller for that endpoint, the Data toggle is considered not valid. If the data toggle is not valid, the device controller assumes the packet was already received and discards the packet (not reporting it to the DCD). To prevent the host controller from re-sending the same packet, the device controller will respond to the error packet by acknowledging it with either an ACK or NYET response.

54.5.6.3.3.3 Priming Transmit Endpoints

Priming a transmit endpoint will cause the device controller to fetch the device transfer descriptor (dTD) for the transaction pointed to by the device queue head (dQH).

After the dTD is fetched, it will be stored in the dQH until the device controller completes the transfer described by the dTD. Storing the dTD in the dQH allows the device controller to fetch the operating context needed to handle a request from the host without the need to follow the linked list, starting at the dQH when the host request is received.

After the device has loaded the dTD, the leading data in the packet is stored in a FIFO in the device controller. This FIFO is split into virtual channels so that the leading data can be stored for any endpoint up to the maximum number of endpoints configured at device synthesis time.

After a priming request is complete, an endpoint state of primed is indicated in the USB_UOG_ENDPTSTATUS register. For a primed transmit endpoint, the device controller can respond to an IN request from the host and meet the stringent bus turnaround time of High Speed USB.

Because only the leading data is stored in the device controller FIFO, it is necessary for the device controller to begin filling in behind leading data after the transaction starts. The FIFO must be sized to account for the maximum latency that can be incurred by the system memory bus. More information about FIFO sizing is presented in section .

54.5.6.3.3.4 Priming Receive Endpoints

Priming receive endpoints is identical to priming of transmit endpoints from the point of view of the DCD. At the device controller the major difference in the operational model is that there is no data movement of the leading packet data simply because the data is to be received from the host.

Note as part of the architecture, the FIFO for the receive endpoints is not partitioned into multiple channels like the transmit FIFO. Thus, the size of the RX FIFO does not scale with the number of endpoints.

54.5.6.4 Operational Model For Packet Transfers

All transactions on the USB bus are initiated by the host and in turn, the device must respond to any request from the host within the turnaround time stated in the USB 2.0 Specification.

At USB 1.1 Full or Low Speed rates, this turnaround time was significant and the USB 1.1 device controllers were architected so that the device controller could access main memory or interrupt a host protocol processor in order to respond to the USB 1.1

transaction. The architecture of the USB 2.0 device controller must be different because same methods will not meet USB 2.0 High-speed turnaround time requirements by simply increasing clock rate.

A USB host will send requests to the device controller in an order that can not be precisely predicted as a single pipeline, so it is not possible to prepare a single packet for the device controller to execute. However, the order of packet requests is predictable when the endpoint number and direction is considered. For example, if endpoint 3 (transmit direction) is configured as a bulk pipe, then we can expect the host will send IN requests to that endpoint. This device controller is architected in such a way that it can prepare packets for each endpoint/direction in anticipation of the host request. The process of preparing the device controller to send or receive data in response to host initiated transaction on the bus is referred to as "priming" the endpoint. This term will be used throughout the following documentation to describe the device controller operation so the DCD can be architected properly use priming. Further, note that the term "flushing" is used to describe the action of clearing a packet that was queued for execution.

54.5.6.4.1 Interrupt/Bulk Endpoint Operational Model

The behaviors of the device controller for interrupt and bulk endpoints are identical.

All valid IN and OUT transactions to bulk pipes will handshake with a NAK unless the endpoint had been primed. Once the endpoint has been primed, data delivery will commence.

A dTD will be retired by the device controller when the packets described in the transfer descriptor have been completed. Each dTD describes N packets to be transferred according to the USB Variable Length transfer protocol. The formula and table on the following page describe how the device controller computes the number and length of the packets to be sent/received by the USB vary according to the total number of bytes and maximum packet length.

With Zero Length Termination (ZLT) = 0

N = INT(Number Of Bytes/Max. Packet Length) + 1

With Zero Length Termination (ZLT) = 1

N = MAXINT(Number Of Bytes/Max. Packet Length)

Table 54-69. Variable Length Transfer Protocol Example (ZLT = 0)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	

Table continues on the next page...

Table 54-69. Variable Length Transfer Protocol Example (ZLT = 0) (continued)

512	256	3	256	256	0
512	512	2	512	0	

Table 54-70. Variable Length Transfer Protocol Example (ZLT = 1)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	
512	256	2	256	256	
512	512	1	512		

NOTE

The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.

TX-dTD is complete when:

- All packets described dTD were successfully transmitted. *** Total bytes in dTD will equal zero when this occurs.

RX-dTD is complete when:

- All packets described in dTD were successfully received. *** Total bytes in dTD will equal zero when this occurs.
- A short packet (number of bytes < maximum packet length) was received. *** This is a successful transfer completion; DCD must check Total Bytes in dTD to determine the number of bytes that are remaining. From the total bytes remaining in the dTD, the DCD can compute the actual bytes received.
- A long packet was received (number of bytes > maximum packet size) OR (total bytes received > total bytes specified). *** This is an error condition. The device controller will discard the remaining packet, and set the Buffer Error bit in the dTD. In addition, the endpoint will be flushed and the USBERR interrupt will become active.

On the successful completion of the packet(s) described by the dTD, the active bit in the dTD will be cleared and the next pointer will be followed when the Terminate bit is clear. When the Terminate bit is set, the device controller will flush the endpoint/direction and cease operations for that endpoint/direction.

On the unsuccessful completion of a packet (see long packet above), the dQH will be left pointing to the dTD that was in error. In order to recover from this error condition, the DCD must properly reinitialize the dQH by clearing the active bit and update the nextTD pointer before attempting to re-prime the endpoint.

NOTE

All packet level errors such as a missing handshake or CRC error will be retried automatically by the device controller.

There is no required interaction with the DCD for handling such errors.

54.5.6.4.1.1 Interrupt/Bulk Endpoint Bus Response Matrix

The table below shows the response matrix for Interrput/Bulk Endpoint Bus.

Table 54-71. Interrupt/Bulk Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	N/A	N/A
In	STALL	NAK	Transmit	BS Error	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK
Ping	STALL	NAK	ACK	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

NOTE

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

54.5.6.4.2 Control Endpoint Operation Model

This section details the setup phase, data phase, status phase, and the control endpoint bus response matrix.

54.5.6.4.2.1 Setup Phase

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase. The device controller will always accept the setup phase unless the setup lockout is engaged.

The setup lockout will engage so that future setup packets are ignored. Lockout of setup packets ensures that while software is reading the setup packet stored in the queue head, that data is not written as it is being read potentially causing an invalid setup packet.

The setup lockout mechanism can be disabled and a new tripwire type semaphore will ensure that the setup packet payload is extracted from the queue head without being corrupted be an incoming setup packet. This is the preferred behavior because ignoring repeated setup packets due to long software interrupt latency would be a compliance issue.

- Disable Setup Lockout by writing 1 to Setup Lockout Mode (SLOM) in USB Device Mode (USB_*n*USBMODE). (once at initialization). Setup lockout is not necessary when using the tripwire as described below.

NOTE

Leaving the Setup Lockout Mode As 0 will result in pre-2.3 hardware behavior.

- After receiving an interrupt and inspecting Endpoint Setup Status (USB_*n*ENDPTSETUPSTAT) to determine that a setup packet was received on a particular pipe:
  a. Write 1 to clear corresponding bit Endpoint Setup Status (USB_*n*ENDPTSETUPSTAT).
  b. Write 1 to Setup Tripwire (SUTW) in USB Command Register (USB_*n*USBCMD) register.
  c. Duplicate contents of dQH.SetupBuffer into local software byte array.
  d. Read Setup TripWire (SUTW) in USB Command Register (USB_*n*USBCMD) register. (if set - continue; if cleared - goto 2)
  e. Write 0 to clear Setup Tripwire (SUTW) in USB Command Register (USB_*n*USBCMD) register.
  f. Process setup packet using local software byte array copy and execute status/ handshake phases.

NOTE

After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed & deallocated before linking a new status and/or handshake dTD for the most recent setup packet.

54.5.6.4.2.2 Data Phase

Following the setup phase, the DCD must create a device transfer descriptor for the data phase and prime the transfer.

After priming the packet, the DCD must verify a new setup packet has not been received by reading the USB.ENDPTSETUPSTAT register immediately verifying that the prime had completed. A prime will complete when the associated bit in the Endpoint Prime (USB_*n*ENDPTPRIME) register is zero and the associated bit in the Endpoint Status

(USB_*n*ENDPTSTAT) register is a one. If a prime fails, ie. The Endpoint Prime (USB_*n*ENDPTPRIME) bit goes to zero and the Endpoint Status (USB_*n*ENDPTSTAT) bit is not set, then the prime has failed. This can only be due to improper setup of the dQH, dTD or a setup arriving during the prime operation. If a new setup packet is indicated after the ENDPTPRIME bit is cleared, then the transfer descriptor can be freed and the DCD must reinterpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller will automatically clear the prime status (Endpoint Status (USB_*n*ENDPTSTAT)) to enforce data coherency with the setup packet.

NOTE

- The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

54.5.6.4.2.3 Status Phase

Similar to the data phase, the DCD must create a transfer descriptor (with byte length equal zero) and prime the endpoint for the status phase.

The DCD must also perform the same checks of the USB.ENDPTSETUPSTAT as described above in the data phase.

NOTE

- The MULT field in the dQH must be set to 00 for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

54.5.6.4.2.4 Control Endpoint Bus Response Matrix

Shown in the following table is the device controller response to packets on a control endpoint according to the device controller state.

The table below shows the response matrix for the Control Endpoint Bus.

Table 54-72. Control Endpoint Bus Response Matrix

Token Type	Endpoint State					Setup Lockout
	Stall	Not Primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	N/A	SYSERR	
In	STALL	NAK	Transmit	BS Error	N/A	N/A

Table continues on the next page...

Table 54-72. Control Endpoint Bus Response Matrix (continued)

Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK	N/A
Ping	STALL	NAK	ACK	N/A	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

54.5.6.4.3 Isochronous Endpoint Operational Model

Isochronous endpoints are used for real-time scheduled delivery of data and their operational model is significantly different than the host throttled Bulk, Interrupt, and Control data pipes.

Real time delivery by the device controller will be accomplished by the following:

- Exactly MULT Packets per (micro) Frame are transmitted/received. Note: MULT is a two-bit field in the device Queue Head. The variable length packet protocol is not used on isochronous endpoints.
- NAK responses are not used. Instead, zero length packets are sent in response to an IN request to an unprimed endpoints. For unprimed RX endpoints, the response to an OUT transaction is to ignore the packet within the device controller.
- Prime requests always schedule the transfer described in the dTD for the next (micro) frame. If the ISO-dTD is still active after that frame, then the ISO-dTD will be held ready until executed or canceled by the DCD.

An EHCI compatible host controller uses the periodic frame list to schedule data exchanges to Isochronous endpoints. The operational model for device mode does not use such a data structure. Instead, the same dTD used for Control/Bulk/Interrupt endpoints is also used for isochronous endpoints. The difference is in the handling of the dTD.

The first difference between bulk and ISO-endpoints is that priming an ISO-endpoint is a delayed operation such that an endpoint will become primed only after a SOF is received. After the DCD writes the prime bit, the prime bit will be cleared as usual to indicate to software that the device controller completed a priming the dTD for transfer. Internal to the design, the device controller hardware masks that prime start until the next frame boundary. This behavior is hidden from the DCD but occurs so that the device controller can match the dTD to a specific (micro)frame.

Another difference with isochronous endpoints is that the transaction must wholly complete in a (micro)frame. Once an ISO transaction is started in a (micro)frame it will retire the corresponding dTD when MULT transactions occur or the device controller finds a fulfillment condition.

The transaction error bit set in the status field indicates a fulfillment error condition. When a fulfillment error occurs, the frame after the transfer failed to complete wholly, the device controller will force retire the ISO-dTD and move to the next ISO-dTD.

It is important to note that fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction will stay primed indefinitely. This means it is up to software discard transmit ISO-dTDs that pile up from a failure of the host to move the data.

Finally, the last difference with ISO packets is in the data level error handling. When a CRC error occurs on a received packet, the packet is not retried similar to bulk and control endpoints. Instead, the CRC is noted by setting the *Transaction Error* bit and the data is stored as usual for the application software to sort out.

- TX Packet Retired
  - MULT counter reaches zero.
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT

NOTE

For TX-ISO, MULT Counter can be loaded with a lesser value in the dTD Multiplier Override field in hardware versions 2.3 and later. If the Multiplier Override is zero, the MULT Counter is initialized to the Multiplier in the QH.

- RX Packet Retired:
  - MULT counter reaches zero.
  - Non-MDATA Data PID is received**
    - ** Exit criteria only valid in hardware version 2.3 or later. Previous to hardware version 2.3, any PID sequence that did not match the MULT field exactly would be flagged as a transaction error due to PID mismatch or fulfillment error.
  - Overflow Error:
    - Packet received is > maximum packet length. [*Buffer Error* bit is set]
    - Packet received exceeds total bytes allocated in dTD. [*Buffer Error* bit is set]
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT
  - CRC Error [*Transaction Error* bit is set]

NOTE

For ISO, when a dTD is retired, the next dTD is primed for the next frame. For continuous (micro)frame to (micro)frame operation the DCD should ensure that the dTD linked-list is out ahead of the device controller by at least two (micro)frames.

54.5.6.4.3.1 Isochronous Pipe Synchronization

When it is necessary to synchronize an isochronous data pipe to the host, the (micro) frame number (USB_UOG_FRINDEX register) can be used as a marker.

To cause a packet transfer to occur at a specific (micro) frame number [N], the DCD should interrupt on SOF during frame N-1. When the USB_UOG_FRINDEX=N-1, the DCD must write the prime bit. The device controller will prime the isochronous endpoint in (micro) frame N-1 so that the device controller will execute delivery during (micro) frame N.

NOTE

Priming an endpoint towards the end of (micro) frame N-1 will not guarantee delivery in (micro) frame N. The delivery may actually occur in (micro) frame N+1 if device controller does not have enough time to complete the prime before the SOF for packet N is received.

54.5.6.4.3.2 Isochronous Endpoint Bus Response Matrix

The following table shows the response matrix for the Isochronous Endpoint Bus.

Table 54-73. Isochronous Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	N/A	N/A
In	NULL Packet	NULL Packet	Transmit	BS Error	N/A
Out	Ignore	Ignore	Receive	N/A	Drop Packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

1. BS Error = Force Bit Stuff Error

   NULL Packet = Zero Length Packet

54.5.6.5 Managing Queue Heads

The following figure shows the End Point Queue Head.

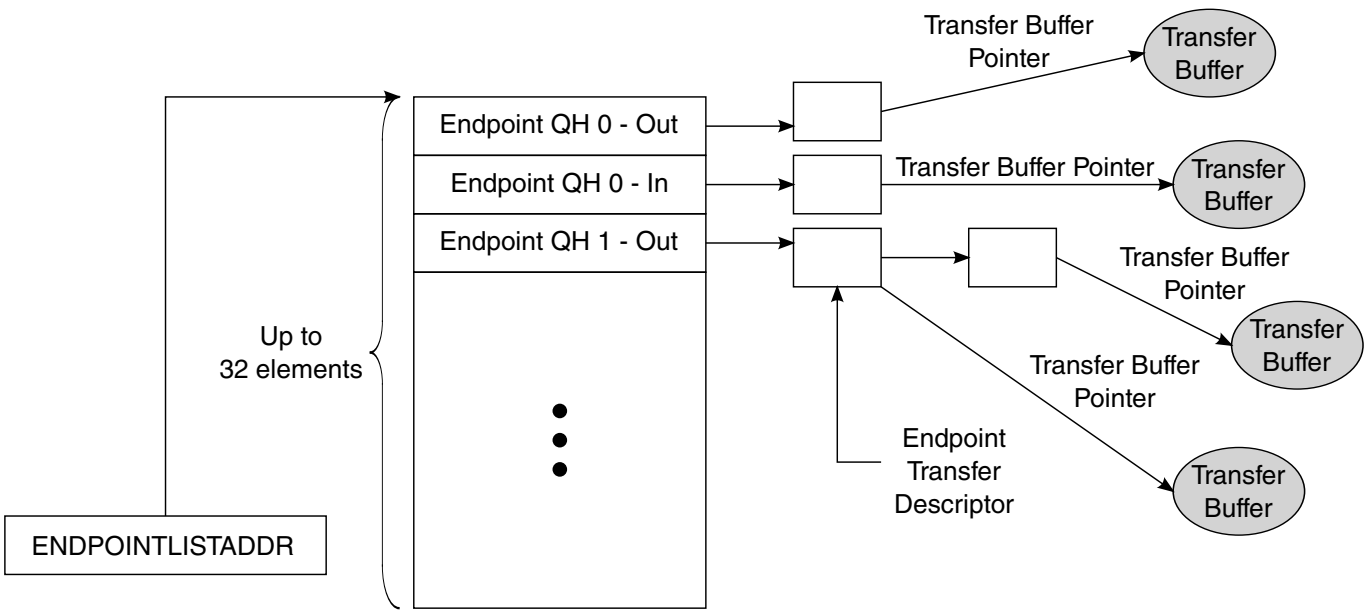


Figure 54-29. End Point Queue Head Diagram

The device queue head (dQH) points to the linked list of transfer tasks, each depicted by the device Transfer Descriptor (dTD). An area of memory pointed to by USB.ENDPOINTLISTADDR contains a group of all dQH's in a sequential list as shown in Figure 54-29. The even elements in the list of dQH's are used for receive endpoints (OUT/SETUP) and the odd elements are used for transmit endpoints (IN/INTERRUPT). Device transfer descriptors are linked head to tail starting at the queue head and ending at a terminate bit. Once the dTD has been retired, it will no longer be part of the linked list from the queue head. Therefore, software is required to track all transfer descriptors because pointers will no longer exist within the queue head once the dTD is retired (see section Software Link Pointers).

In addition to the current and next pointers and the dTD overlay examined in section Operational Model For Packet Transfers, the dQH also contains the following parameters for the associated endpoint: Multipler, Maximum Packet Length, Interrupt On Setup. The complete initialization of the dQH including these fields is demonstrated in the next section.

54.5.6.5.1 Queue Head Initialization

One device queue head must be initialized for each active endpoint.

To initialize a device queue head:

- Write the wMaxPacketSize field as required by the USB Chapter 9 or application specific protocol.
- Write the multiplier field to 0 for control, bulk, and interrupt endpoints. For ISO endpoints, set the multiplier to 1,2, or 3 as required bandwidth and in conjunction with the USB Chapter 9 protocol.

NOTE

In FS mode, the multiplier field can only be 1 for ISO endpoints.

- Write the next dTD Terminate bit field to 1.
- Write the Active bit in the status field to 0.
- Write the Halt bit in the status field to 0.

NOTE

The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTD's.

54.5.6.5.2 Operational Model For Setup Transfers

As discussed in section Control Endpoint Operation Model, setup transfer requires special treatment by the DCD. A setup transfer does not use a dTD but instead stores the incoming data from a setup packet in an 8-byte buffer within the dQH.

Upon receiving notification of the setup packet, the DCD should handle the setup transfer as demonstrated here:

1. Copy setup buffer contents from dQH - RX to software buffer.
2. Acknowledge setup backup by writing a "1" to the corresponding bit in ENDPTSETUPSTAT.

NOTE

- The acknowledge must occur before continuing to process the setup packet.
- After the acknowledge has occurred, the DCD must not attempt to access the setup buffer in the dQH - RX. Only the local software copy should be examined.

3. Check for pending data or status dTD's from previous control transfers and flush if any exist as discussed in section Flushing/De-priming an Endpoint.
4. Decode setup packet and prepare data phase [optional] and status phase transfer as required by the USB Chapter 9 or application specific protocol.

NOTE

It is possible for the device controller to receive setup packets before previous control transfers complete. Existing control packets in progress must be flushed and the new control packet completed.

54.5.6.6 Managing Transfers with Transfer Descriptors

54.5.6.6.1 Software Link Pointers

It is necessary for the DCD software to maintain head and tail pointers to the for the linked list of dTDs for each respective queue head.

This is necessary because the dQH only maintains pointers to the current working dTD and the next dTD to be executed. The operations described in next section for managing dTD will assume the DCD can use reference the head and tail of the dTD linked list. The following figure shows the Software Link Pointers.

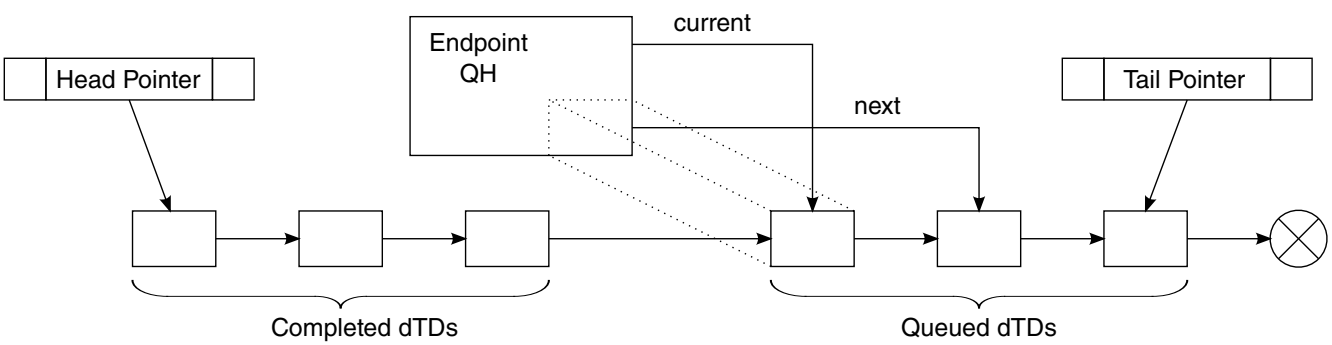


Figure 54-30. Software Link Pointers

NOTE

To conserve memory, the reserved fields at the end of the dQH can be used to store the Head & Tail pointers, but it still remains the responsibility of the DCD to maintain the pointers.

54.5.6.6.2 Building a Transfer Descriptor

Before a transfer can be executed from the linked list, a dTD must be built to describe the transfer.

Use the following procedure for building dTDs.

Allocate 8-DWord dTD block of memory aligned to 8-DWord boundaries. Example: bit address 4:0 would be equal to "00000"

Write the following fields:

1. Initialize first 7 DWords to 0.
2. Set the terminate bit to 1.
3. Fill in total bytes with transfer size.
4. Set the interrupt on complete if desired.
5. Initialize the status field with the active bit set to 1 and all remaining status bits set to 0.
6. Fill in buffer pointer page 0 and the current offset to point to the start of the data buffer.
7. Initialize buffer pointer page 1 through page 4 to be one greater than each of the previous buffer pointer.

54.5.6.6.3 Executing A Transfer Descriptor

To safely add a dTD, the DCD must be follow this procedure which will handle the event where the device controller reaches the end of the dTD list at the same time a new dTD is being added to the end of the list.

Determine whether the link list is empty: Check DCD driver to see if pipe is empty (internal representation of linked-list should indicate if any packets are outstanding).

- Case 1: Link list is empty
  a. Write dQH next pointer AND dQH terminate bit to 0 as a single DWord operation.
  b. Clear active & halt bit in dQH (in case set from a previous error).
  c. Prime endpoint by writing 1 to correct bit position in .
- Case 2: Link list is not empty
  a. Add dTD to end of linked list.
  b. Read correct prime bit in - if 1 DONE.
  c. Set ATDTW bit in USBCMD register to 1.
  d. Read correct status bit in . (store in tmp. variable for later)
  e. Read ATDTW bit in USBCMD register.
     - If 0 goto 3.
     - If 1 continue to 6.
  f. Write ATDTW bit in USBCMD register to 0.
  g. If status bit read in (3) is 1 DONE.
  h. If status bit read in (3) is 0 then Goto Case 1: Step 1.

54.5.6.6.4 Transfer Completion

After a dTD has been initialized and the associated endpoint primed the device controller will execute the transfer upon the host-initiated request. The DCD will be notified with a USB interrupt if the Interrupt On Complete bit was set or alternately, the DCD can poll the endpoint complete register to find when the dTD had been executed. After a dTD has been executed, DCD can check the status bits to determine success or failure.

NOTE

Multiple dTD can be completed in a single endpoint complete notification. After clearing the notification, DCD must search the dTD linked list and retire all dTDs that have finished (Active bit cleared).

By reading the status fields of the completed dTDs, the DCD can determine if the transfers completed successfully. Success is determined with the following combination of status bits:

- Active = 0
- Halted = 0
- Transaction Error = 0
- Data Buffer Error = 0

Should any combination other than the one shown above exist, the DCD must take proper action. Transfer failure mechanisms are indicated in the Device Error Matrix.

In addition to checking the status bit the DCD must read the Transfer Bytes field to determine the actual bytes transferred. When a transfer is complete, the Total Bytes transferred is by decremented by the actual bytes transferred. For Transmit packets, a packet is only complete after the actual bytes reaches zero, but for receive packets, the host may send fewer bytes in the transfer according the USB variable length packet protocol.

54.5.6.6.5 Flushing/De-priming an Endpoint

It is necessary for the DCD to flush to de-prime one more endpoints on a USB device reset or during a broken control transfer.

There may also be application specific requirements to stop transfers in progress. The following procedure can be used by the DCD to stop a transfer in progress:

1. Write a '1' to the corresponding bit(s) in .
2. Wait until all bits in are '0'.

- Software note: this operation may take a large amount of time depending on the USB bus activity. It is not desirable to have this wait loop within an interrupt service routine.

3. Read to ensure that for all endpoints commanded to be flushed, that the corresponding bits are now '0'. If the corresponding bits are '1' after step #2 has finished, then the flush failed as described in the following:
   - Explanation: In very rare cases, a packet is in progress to the particular endpoint when commanded flush using . A safeguard is in place to refuse the flush to ensure that the packet in progress completes successfully. The DCD may need to repeatedly flush any endpoints that fail to flush be repeating steps 1-3 until each endpoint is successfully flushed.

54.5.6.6.6 Device Error Matrix

The following table summarizes packet errors that are not automatically handled by the Device Controller.

Table 54-74. Device Error Matrix

Error	Direction	Packet Type	Data Buffer Error Bit	Transaction Error Bit
Overflow **	RX	Any	1	0
ISO Packet Error	RX	ISO	0	1
ISO Fulfillment Error	Both	ISO	0	1

Notice that the device controller handles all errors on Bulk/Control/Interrupt Endpoints except for a data buffer overflow. However, for ISO endpoints, errors packets are not retried and errors are tagged as indicated. The table below describes the errors.

Table 54-75. Error Descriptions

Error	Description
Overflow	Number of bytes received exceeded max. packet size or total buffer length. ** This error will also set the Halt bit in the dQH and if there are dTDs remaining in the linked list for the endpoint, then those will not be executed.
ISO Packet Error	CRC Error on received ISO packet. Contents not guaranteed to be correct.
ISO Fulfillment Error	Host failed to complete the number of packets defined in the dQH mult field within the given (micro)frame. For scheduled data delivery the DCD may need to readjust the data queue because a fulfillment error will cause Device Controller to cease data transfers on the pipe for one (micro)frame. During the "dead" (micro)frame, the Device Controller reports error on the pipe and primes for the following frame.

54.5.6.7 Servicing Interrupts

The interrupt service routine must consider that there are high-frequency, low-frequency operations, and error operations and order accordingly.

54.5.6.7.1 High-Frequency Interrupts

High frequency interrupts in particular should be handled in the order below. The most important of these is listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

The table below describes the High frequency interrupt events.

Table 54-76. High Frequency Interrupt Events

Execution Order	Interrupt	Action
1a	USB Interrupt - USB.ENDPTSETUPSTATUS	Copy contents of setup buffer and acknowledge setup packet (as indicated in Figure 54-29 shows the End Point Queue Head). Process setup packet according to USB 2.0 Chapter 9 or application specific protocol.
1b	USB Interrupt[1] - USB.ENDPTCOMPLETE	Handle completion of dTD as indicated in Figure 54-29 shows the End Point Queue Head.
2	SOF Interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

1. It is likely that multiple interrupts to stack up on any call to the Interrupt Service Routine AND during the Interrupt Service Routine.

54.5.6.7.2 Low-Frequency Interrupts

The low frequency interrupts can be handled in any order because they do not occur often in comparison to the high-frequency interrupts.

The table below shows the Low frequency interrupt events.

Table 54-77. Low Frequency Interrupt Events

Interrupt	Action
Port Change	Change software state information.
Sleep Enable (Suspend)	Change software state information. Low power handling as necessary.
Reset Received	Change software state information. Abort pending transfers.

54.5.6.7.3 Error Interrupts

Error interrupts will be least frequent and should be placed last in the interrupt service routine.

The following table shows the error interrput events.

Table 54-78. Error Interrupt Events

Interrupt	Action
USB Error Interrupt	This error is redundant because it combines USB Interrupt and an error status in the dTD. The DCD will more aptly handle packet-level errors by checking dTD status field upon receipt of USB Interrupt (w/ USB.ENDPTCOMPLETE).
System Error	Unrecoverable error. Immediate Reset of core; free transfers buffers in progress and restart the DCD.

54.6 USB Non-Core Memory Map/Register Definition

There are two kinds of registers in the USB module: USB core registers and USB non-core registers.

USB core registers are used to control USB core functions, and more independent of USB features. Each USB controller core has its own core registers.

USB non-core registera are additional to USB core registers, and more dependent on USB features. i.MX series products vary in non-core registers.

This section describes only the USB non-core registers. For detailed descriptions of USB core registers, please refer to Register Interface .

NOTE

- For reserved bits, please preserve the value when writing (read its reset value, then write this value back)
- "USB_UOG1_", "USB_UOG2_" prefix in register name indicates it is a core register for OTG1/OTG2 controller core respectively.
- USBNC_USB_" prefix in register name indicates it is a USB non-core register.

USBNC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4033_0200	USB OTG Control 1 Register (USBNC_OTG1_CTRL1)	32	R/W	3000_1000h	54.6.1/2461
4033_0204	USB OTG Control 2 Register (USBNC_OTG1_CTRL2)	32	R/W	See section	54.6.2/2464
4034_0200	USB OTG Control 1 Register (USBNC_OTG2_CTRL1)	32	R/W	3000_1000h	54.6.1/2461

Table continues on the next page...

USBNC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4034_0204	USB OTG Control 2 Register (USBNC_OTG2_CTRL2)	32	R/W	See section	54.6.2/2464
4034_0210	USB Host HSIC Control Register (USBNC_OTG2_HSIC_CTRL)	32	R/W	0000_4084h	54.6.3/2466

54.6.1 USB OTG Control 1 Register (USBNC_*n*_CTRL1)

The USB Control 1 register controls the integration specific features of the USB OTG1 and OTG2 module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Address: 4033_0000h base + 200h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WIR	Reserved	WKUP_DPDM_EN	Reserved								ULPI_PHY_CLK_EN	Reserved		WKUP_VBUS_EN	WKUP_ID_EN
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WKUP_SW	WKUP_SW_EN	Reserved			WIE	PWR_POL	OVER_CUR_POL	OVER_CUR_DIS	Reserved						
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

USBNC_*n*_CTRL1 field descriptions

Field	Description
31 WIR	Wake-up Interrupt Request This bit indicates that a wake-up interrupt request is received on the OTG1 port. This bit is cleared by disabling the wake-up interrupt (clearing bit "OWIE"). 1 Wake-up Interrupt Request received 0 No wake-up interrupt request received
30 Reserved	This field is reserved.
29 WKUP_DPDM_EN	Wake-up on DPDM change enable 1 (Default) DPDM changes wake-up to be enabled, it is for device only. 0 DPDM changes wake-up to be disabled only when VBUS is 0.
28–21 Reserved	This field is reserved.
20 ULPI_PHY_CLK_EN	ULPI PHY clock enable This bit is for ULPI interface only. 1 Enable 0 Disable
19–18 -	This field is reserved. Reserved
17 WKUP_VBUS_EN	Wake-up on VBUS change enable 1 Enable 0 Disable
16 WKUP_ID_EN	Wake-up on ID change enable 1 Enable 0 Disable
15 WKUP_SW	Software Wake-up 1 Force wake-up 0 Inactive
14 WKUP_SW_EN	Software Wake-up Enable 1 Enable 0 Disable
13–11 -	This field is reserved. Reserved
10 WIE	Wake-up Interrupt Enable This bit enables or disables the OTG1 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
9 PWR_POL	Power Polarity

Table continues on the next page...

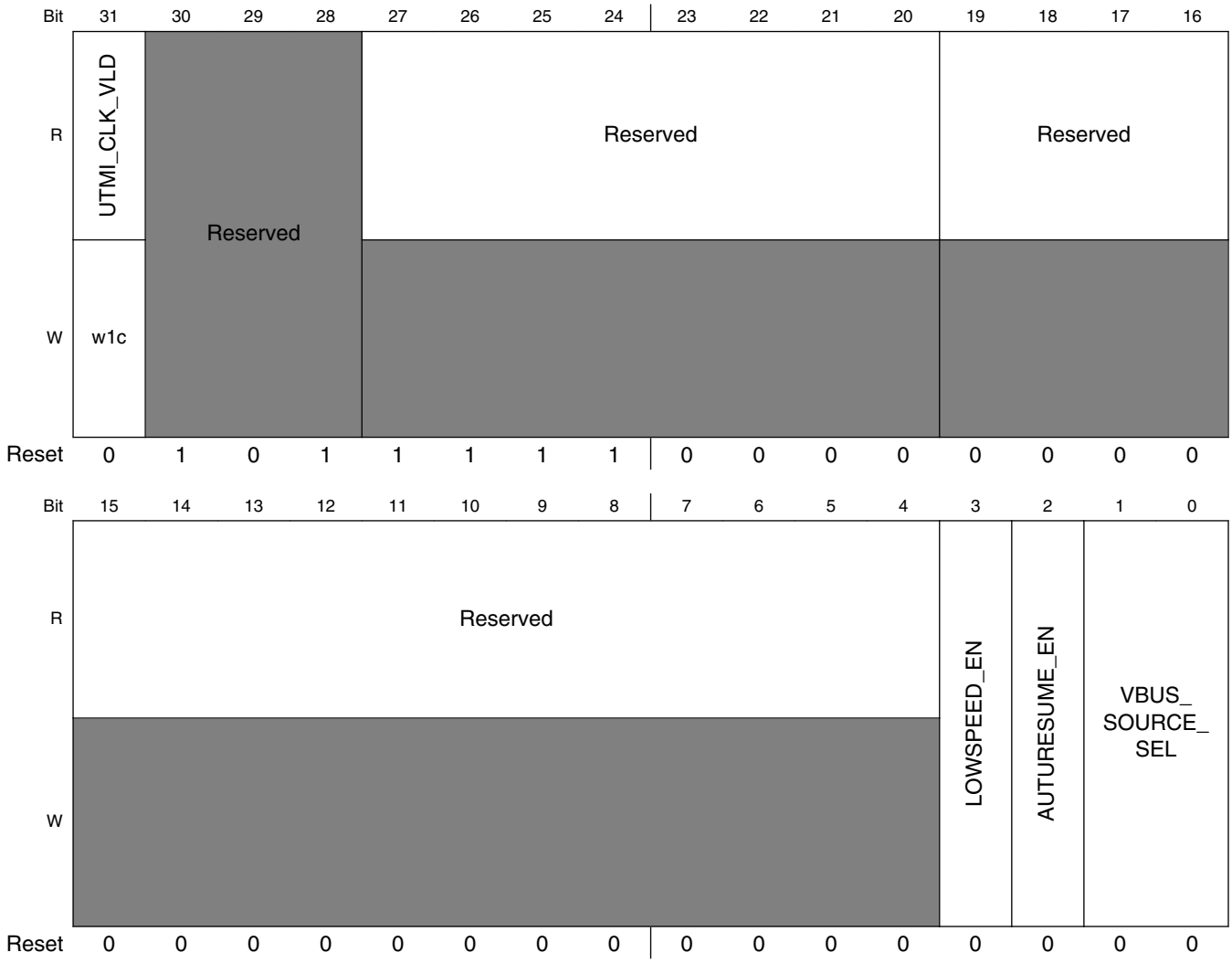
USBNC_*n*_CTRL1 field descriptions (continued)

Field	Description
	This bit should be set according to PMIC Power Pin polarity. 1 PMIC Power Pin is High active. 0 PMIC Power Pin is Low active.
8 OVER_CUR_POL	Polarity of Overcurrent The polarity of OTG1/OTG2 port overcurrent event 1 Low active (low on this signal represents an overcurrent condition) 0 High active (high on this signal represents an overcurrent condition)
7 OVER_CUR_DIS	Disable Overcurrent Detection 1 Disables overcurrent detection 0 Enables overcurrent detection
-	This field is reserved. Reserved

54.6.2 USB OTG Control 2 Register (USBNC_*n*_CTRL2)

The USB Control 2 register controls the integration specific features of the USB module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wakeup functionality.

Address: 4033_0000h base + 204h offset + (65536d × i), where i=0d to 1d



USBNC_*n*_CTRL2 field descriptions

Field	Description
31 UTMI_CLK_VLD	Indicate whether the UTMI clock to the USB PHY is valid. Write 1 to clear 0 Default

Table continues on the next page...

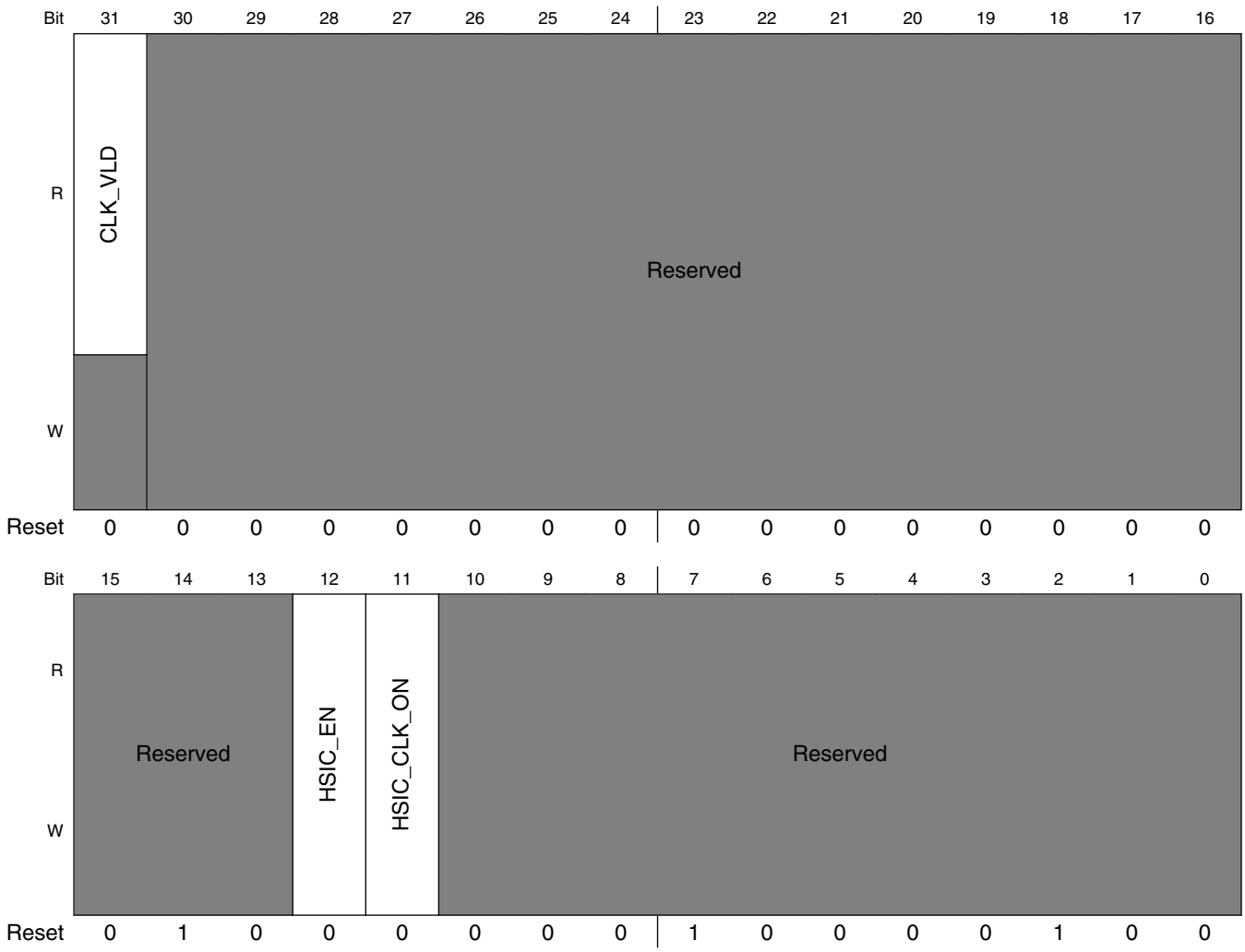
USBNC_*n*_CTRL2 field descriptions (continued)

Field	Description
30–28 Reserved	This field is reserved.
27–20 Reserved	This field is reserved. 0B00011110 Default
19–4 Reserved	This field is reserved.
3 LOWSPEED_EN	Set if AUTURESUME_EN is set and works on low speed. 0 Default
2 AUTURESUME_EN	Auto Resume Enable It is for UTMI PHY host mode only (UH core has no this feature since HSIC PHY does not support auto resume). To set USB, it will send resume with 32 KHz clock when it detects remote wakeup from device. This feature is useful if device drive a very short remote wakeup (minimul value is 1 ms for USB2 spec) and system 24 MHz is off during suspend mode. 0 Default
VBUS_ SOURCE_SEL	VBUS source select when detect VBUS wakeup event, it is for UTMI PHY only (UH core has no such feature). 2'b00 vbus_valid others sess_valid

54.6.3 USB Host HSIC Control Register (USBNC_*n*_HSIC_CTRL)

The USB Host HSIC control register controls Host high speed IC configuration. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control.

Address: 4033_0000h base + 1_0210h offset + (65536d × i), where i=0d to 0d



USBNC_*n*_HSIC_CTRL field descriptions

Field	Description
31 CLK_VLD	Indicating whether Host HSIC clock is valid. 1 Valid 2 Invalid

Table continues on the next page...

USBNC_*n*_HSIC_CTRL field descriptions (continued)

Field	Description
30–13 -	This field is reserved. Reserved
12 HSIC_EN	Host HSIC enable 1 Enabled 0 Disabled
11 HSIC_CLK_ON	Force Host HSIC module 480M clock on, even when in Host is in suspend mode. 1 Active 0 Inactive
-	This field is reserved. Reserved

54.7 USB Core Memory Map/Register Definition

USB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4033_0000	Identification register (USB_UOG1_ID)	32	R	E401_FA05h	54.7.1/2471
4033_0004	Hardware General (USB_UOG1_HWGENERAL)	32	R	0000_0015h	54.7.2/2472
4033_0008	Host Hardware Parameters (USB_UOG1_HWHOST)	32	R	1002_0001h	54.7.3/2474
4033_000C	Device Hardware Parameters (USB_UOG1_HWDEVICE)	32	R	0000_0011h	54.7.4/2474
4033_0010	TX Buffer Hardware Parameters (USB_UOG1_HWTXBUF)	32	R	8008_0B08h	54.7.5/2475
4033_0014	RX Buffer Hardware Parameters (USB_UOG1_HWRXBUF)	32	R	0000_0808h	54.7.6/2476
4033_0080	General Purpose Timer #0 Load (USB_UOG1_GPTIMER0LD)	32	R/W	0000_0000h	54.7.7/2476
4033_0084	General Purpose Timer #0 Controller (USB_UOG1_GPTIMER0CTRL)	32	R/W	0000_0000h	54.7.8/2477
4033_0088	General Purpose Timer #1 Load (USB_UOG1_GPTIMER1LD)	32	R/W	0000_0000h	54.7.9/2478
4033_008C	General Purpose Timer #1 Controller (USB_UOG1_GPTIMER1CTRL)	32	R/W	0000_0000h	54.7.10/ 2479
4033_0090	System Bus Config (USB_UOG1_SBUSCFG)	32	R/W	0000_0002h	54.7.11/ 2480
4033_0100	Capability Registers Length (USB_UOG1_CAPLENGTH)	8	R	40h	54.7.12/ 2481
4033_0102	Host Controller Interface Version (USB_UOG1_HCIVERSION)	16	R	0100h	54.7.13/ 2481
4033_0104	Host Controller Structural Parameters (USB_UOG1_HCSPARAMS)	32	R	0001_0011h	54.7.14/ 2482

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4033_0108	Host Controller Capability Parameters (USB_UOG1_HCCPARAMS)	32	R	0000_0006h	54.7.15/ 2484
4033_0120	Device Controller Interface Version (USB_UOG1_DCIVERSION)	16	R	0001h	54.7.16/ 2486
4033_0124	Device Controller Capability Parameters (USB_UOG1_DCCPARAMS)	32	R	0000_0188h	54.7.17/ 2486
4033_0140	USB Command Register (USB_UOG1_USBCMD)	32	R/W	0008_0000h	54.7.18/ 2488
4033_0144	USB Status Register (USB_UOG1_USBSTS)	32	R/W	0000_0000h	54.7.19/ 2492
4033_0148	Interrupt Enable Register (USB_UOG1_USBINTR)	32	R/W	0000_0000h	54.7.20/ 2496
4033_014C	USB Frame Index (USB_UOG1_FRINDEX)	32	R/W	0000_0000h	54.7.21/ 2498
4033_0154	Frame List Base Address (USB_UOG1_PERIODICLISTBASE)	32	R/W	0000_0000h	54.7.22/ 2499
4033_0154	Device Address (USB_UOG1_DEVICEADDR)	32	R/W	0000_0000h	54.7.23/ 2499
4033_0158	Next Asynch. Address (USB_UOG1_ASYNCLISTADDR)	32	R/W	0000_0000h	54.7.24/ 2500
4033_0158	Endpoint List Address (USB_UOG1_ENDPTLISTADDR)	32	R/W	0000_0000h	54.7.25/ 2501
4033_0160	Programmable Burst Size (USB_UOG1_BURSTSIZE)	32	R/W	0000_0000h	54.7.26/ 2501
4033_0164	TX FIFO Fill Tuning (USB_UOG1_TXFILLTUNING)	32	R/W	0000_0808h	54.7.27/ 2502
4033_0178	Endpoint NAK (USB_UOG1_ENDPTNAK)	32	R/W	0000_0000h	54.7.28/ 2504
4033_017C	Endpoint NAK Enable (USB_UOG1_ENDPTNAKEN)	32	R/W	0000_0000h	54.7.29/ 2504
4033_0180	Configure Flag Register (USB_UOG1_CONFIGFLAG)	32	R/W	0000_0001h	54.7.30/ 2505
4033_0184	Port Status & Control (USB_UOG1_PORTSC1)	32	R/W	1000_0000h	54.7.31/ 2505
4033_01A4	On-The-Go Status & control (USB_UOG1_OTGSC)	32	R/W	0000_0120h	54.7.32/ 2512
4033_01A8	USB Device Mode (USB_UOG1_USBMODE)	32	R/W	0000_0000h	54.7.33/ 2516
4033_01AC	Endpoint Setup Status (USB_UOG1_ENDPTSETUPSTAT)	32	R/W	0000_0000h	54.7.34/ 2517
4033_01B0	Endpoint Prime (USB_UOG1_ENDPTPRIME)	32	R/W	0000_0000h	54.7.35/ 2518
4033_01B4	Endpoint Flush (USB_UOG1_ENDPTFLUSH)	32	R/W	0000_0000h	54.7.36/ 2519

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4033_01B8	Endpoint Status (USB_UOG1_ENDPTSTAT)	32	R	0000_0000h	54.7.37/ 2519
4033_01BC	Endpoint Complete (USB_UOG1_ENDPTCOMPLETE)	32	R/W	0000_0000h	54.7.38/ 2520
4033_01C0	Endpoint Control0 (USB_UOG1_ENDPTCTRL0)	32	R/W	0080_0080h	54.7.39/ 2521
4033_01C4	Endpoint Control 1 (USB_UOG1_ENDPTCTRL1)	32	R/W	0000_0000h	54.7.40/ 2523
4033_01C8	Endpoint Control 2 (USB_UOG1_ENDPTCTRL2)	32	R/W	0000_0000h	54.7.41/ 2526
4033_01CC	Endpoint Control 3 (USB_UOG1_ENDPTCTRL3)	32	R/W	0000_0000h	54.7.42/ 2528
4033_01D0	Endpoint Control 4 (USB_UOG1_ENDPTCTRL4)	32	R/W	0000_0000h	54.7.43/ 2531
4033_01D4	Endpoint Control 5 (USB_UOG1_ENDPTCTRL5)	32	R/W	0000_0000h	54.7.44/ 2534
4033_01D8	Endpoint Control 6 (USB_UOG1_ENDPTCTRL6)	32	R/W	0000_0000h	54.7.45/ 2537
4033_01DC	Endpoint Control 7 (USB_UOG1_ENDPTCTRL7)	32	R/W	0000_0000h	54.7.46/ 2540
4034_0000	Identification register (USB_UOG2_ID)	32	R	E401_FA05h	54.7.1/2471
4034_0004	Hardware General (USB_UOG2_HWGENERAL)	32	R	0000_0015h	54.7.2/2472
4034_0008	Host Hardware Parameters (USB_UOG2_HWHOST)	32	R	1002_0001h	54.7.3/2474
4034_000C	Device Hardware Parameters (USB_UOG2_HWDEVICE)	32	R	0000_0011h	54.7.4/2474
4034_0010	TX Buffer Hardware Parameters (USB_UOG2_HWTXBUF)	32	R	8008_0B08h	54.7.5/2475
4034_0014	RX Buffer Hardware Parameters (USB_UOG2_HWRXBUF)	32	R	0000_0808h	54.7.6/2476
4034_0080	General Purpose Timer #0 Load (USB_UOG2_GPTIMER0LD)	32	R/W	0000_0000h	54.7.7/2476
4034_0084	General Purpose Timer #0 Controller (USB_UOG2_GPTIMER0CTRL)	32	R/W	0000_0000h	54.7.8/2477
4034_0088	General Purpose Timer #1 Load (USB_UOG2_GPTIMER1LD)	32	R/W	0000_0000h	54.7.9/2478
4034_008C	General Purpose Timer #1 Controller (USB_UOG2_GPTIMER1CTRL)	32	R/W	0000_0000h	54.7.10/ 2479
4034_0090	System Bus Config (USB_UOG2_SBUSCFG)	32	R/W	0000_0002h	54.7.11/ 2480
4034_0100	Capability Registers Length (USB_UOG2_CAPLENGTH)	8	R	40h	54.7.12/ 2481
4034_0102	Host Controller Interface Version (USB_UOG2_HCIVERSION)	16	R	0100h	54.7.13/ 2481
4034_0104	Host Controller Structural Parameters (USB_UOG2_HCSPARAMS)	32	R	0001_0011h	54.7.14/ 2482

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4034_0108	Host Controller Capability Parameters (USB_UOG2_HCCPARAMS)	32	R	0000_0006h	54.7.15/ 2484
4034_0120	Device Controller Interface Version (USB_UOG2_DCIVERSION)	16	R	0001h	54.7.16/ 2486
4034_0124	Device Controller Capability Parameters (USB_UOG2_DCCPARAMS)	32	R	0000_0188h	54.7.17/ 2486
4034_0140	USB Command Register (USB_UOG2_USBCMD)	32	R/W	0008_0000h	54.7.18/ 2488
4034_0144	USB Status Register (USB_UOG2_USBSTS)	32	R/W	0000_0000h	54.7.19/ 2492
4034_0148	Interrupt Enable Register (USB_UOG2_USBINTR)	32	R/W	0000_0000h	54.7.20/ 2496
4034_014C	USB Frame Index (USB_UOG2_FRINDEX)	32	R/W	0000_0000h	54.7.21/ 2498
4034_0154	Frame List Base Address (USB_UOG2_PERIODICLISTBASE)	32	R/W	0000_0000h	54.7.22/ 2499
4034_0154	Device Address (USB_UOG2_DEVICEADDR)	32	R/W	0000_0000h	54.7.23/ 2499
4034_0158	Next Asynch. Address (USB_UOG2_ASYNCLISTADDR)	32	R/W	0000_0000h	54.7.24/ 2500
4034_0158	Endpoint List Address (USB_UOG2_ENDPTLISTADDR)	32	R/W	0000_0000h	54.7.25/ 2501
4034_0160	Programmable Burst Size (USB_UOG2_BURSTSIZE)	32	R/W	0000_0000h	54.7.26/ 2501
4034_0164	TX FIFO Fill Tuning (USB_UOG2_TXFILLTUNING)	32	R/W	0000_0808h	54.7.27/ 2502
4034_0178	Endpoint NAK (USB_UOG2_ENDPTNAK)	32	R/W	0000_0000h	54.7.28/ 2504
4034_017C	Endpoint NAK Enable (USB_UOG2_ENDPTNAKEN)	32	R/W	0000_0000h	54.7.29/ 2504
4034_0180	Configure Flag Register (USB_UOG2_CONFIGFLAG)	32	R/W	0000_0001h	54.7.30/ 2505
4034_0184	Port Status & Control (USB_UOG2_PORTSC1)	32	R/W	1000_0000h	54.7.31/ 2505
4034_01A4	On-The-Go Status & control (USB_UOG2_OTGSC)	32	R/W	0000_0120h	54.7.32/ 2512
4034_01A8	USB Device Mode (USB_UOG2_USBMODE)	32	R/W	0000_0000h	54.7.33/ 2516
4034_01AC	Endpoint Setup Status (USB_UOG2_ENDPTSETUPSTAT)	32	R/W	0000_0000h	54.7.34/ 2517
4034_01B0	Endpoint Prime (USB_UOG2_ENDPTPRIME)	32	R/W	0000_0000h	54.7.35/ 2518
4034_01B4	Endpoint Flush (USB_UOG2_ENDPTFLUSH)	32	R/W	0000_0000h	54.7.36/ 2519

Table continues on the next page...

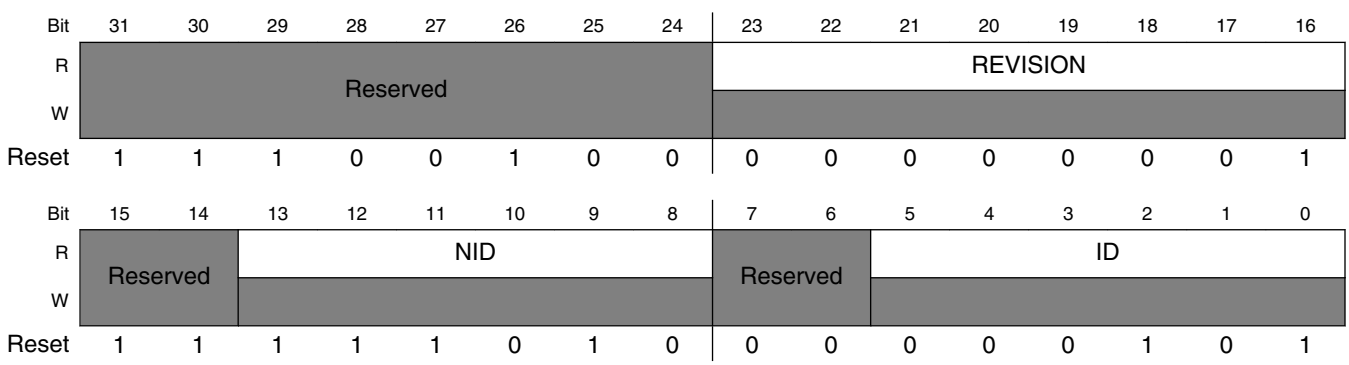
USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4034_01B8	Endpoint Status (USB_UOG2_ENDPTSTAT)	32	R	0000_0000h	54.7.37/ 2519
4034_01BC	Endpoint Complete (USB_UOG2_ENDPTCOMPLETE)	32	R/W	0000_0000h	54.7.38/ 2520
4034_01C0	Endpoint Control0 (USB_UOG2_ENDPTCTRL0)	32	R/W	0080_0080h	54.7.39/ 2521
4034_01C4	Endpoint Control 1 (USB_UOG2_ENDPTCTRL1)	32	R/W	0000_0000h	54.7.40/ 2523
4034_01C8	Endpoint Control 2 (USB_UOG2_ENDPTCTRL2)	32	R/W	0000_0000h	54.7.41/ 2526
4034_01CC	Endpoint Control 3 (USB_UOG2_ENDPTCTRL3)	32	R/W	0000_0000h	54.7.42/ 2528
4034_01D0	Endpoint Control 4 (USB_UOG2_ENDPTCTRL4)	32	R/W	0000_0000h	54.7.43/ 2531
4034_01D4	Endpoint Control 5 (USB_UOG2_ENDPTCTRL5)	32	R/W	0000_0000h	54.7.44/ 2534
4034_01D8	Endpoint Control 6 (USB_UOG2_ENDPTCTRL6)	32	R/W	0000_0000h	54.7.45/ 2537
4034_01DC	Endpoint Control 7 (USB_UOG2_ENDPTCTRL7)	32	R/W	0000_0000h	54.7.46/ 2540

54.7.1 Identification register (USB_*n*ID)

The ID register identifies the USB 2.0 High-Speed core and its revision.

Address: 4033_0000h base + 0h offset + (65536d × i), where i=0d to 1d



USB_*n*ID field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 REVISION	Revision number of the controller core.
15–14 -	This field is reserved. Reserved
13–8 NID	Complement version of ID
7–6 -	This field is reserved. Reserved
ID	Configuration number. This number is set to 0x05 and indicates that the peripheral is USB 2.0 High-Speed core.

54.7.2 Hardware General (USB_*n*HWGENERAL)

General hardware parameters as defined in System Level Issues and Core Configuration.

NOTE

The reset value could vary from instance to instance. Please see the detail in bit field description and ignore reset value in summary table in this case!

Address: 4033_0000h base + 4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					SM		PHYM			PHYW		Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1

USB_*n*HWGENERAL field descriptions

Field	Description
31–11 -	This field is reserved. Reserved
10–9 SM	Serial interface mode capability SM bit reset value is '00b' 00 No Serial Engine, always use parallel signalling.

Table continues on the next page...

USB_*n*HWGENERAL field descriptions (continued)

Field	Description
	01 Serial Engine present, always use serial signalling for FS/LS. 10 Software programmable - Reset to use parallel signalling for FS/LS 11 Software programmable - Reset to use serial signalling for FS/LS
8–6 PHYM	Transciever type PHYM bit reset value: '0000b' for OTG controller core, '0100b' for Host-only controller core. 000 UTMI/UMTI+ 001 ULPI DDR 010 ULPI 011 Serial Only 100 Software programmable - reset to UTMI/UTMI+ 101 Software programmable - reset to ULPI DDR 110 Software programmable - reset to ULPI 111 Software programmable - reset to Serial 1000 IC-USB 1001 Software programmable - reset to IC-USB 1010 HSIC 1011 Software programmable - reset to HSIC
5–4 PHYW	Data width of the transciever connected to the controller core. PHYW bit reset value is PHYW bit reset value is '01b'. 00 8 bit wide data bus Software non-programmable 01 16 bit wide data bus Software non-programmable 10 Reset to 8 bit wide data bus Software programmable 11 Reset to 16 bit wide data bus Software programmable
-	This field is reserved. Reserved

54.7.3 Host Hardware Parameters (USB_*n*HWHOST)

Address: 4033_0000h base + 8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												NPORT			HC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

USB_*n*HWHOST field descriptions

Field	Description
31–4 -	This field is reserved. Reserved
3–1 NPORT	The Nmber of downstream ports supported by the host controller is (NPORT+1). **NOTE:** When these bits value is '000', it indicates a single-port host controller.
0 HC	Host Capable. Indicating whether host operation mode is supported or not. 1 Supported 0 Not supported

54.7.4 Device Hardware Parameters (USB_*n*HWDEVICE)

NOTE

This register is only available in OTG core.

Address: 4033_0000h base + Ch offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										DEVEP					DC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

USB_*n*HWDEVICE field descriptions

Field	Description
31–6 -	This field is reserved. Reserved
5–1 DEVEP	Device Endpoint Number
0 DC	Device Capable. Indicating whether device operation mode is supported or not. 1 Supported 0 Not supported

54.7.5 TX Buffer Hardware Parameters (USB_*n*HWTXBUF)

Address: 4033_0000h base + 10h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								TXCHANADD								Reserved								TXBURST							
W																																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0

USB_*n*HWTXBUF field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 TXCHANADD	TX FIFO Buffer size is: (2^TXCHANADD) * 4 Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. For the OTG controller operating in device mode, this is the FIFO buffer size per endpoint. As the OTG controller has 8 TX endpoint, there are 8 of these buffers. For the OTG controller operating in host mode, or for Host-only controller, the entire buffer memory is used as a single TX buffer. Therefore, there is only 1 of this buffer
15–8 -	This field is reserved. Reserved
TXBURST	Default burst size for memory to TX buffer transfer. This is reset value of TXPBURST bits in USB core regsiter USB_n_BURSTSIZE. Please see Programmable Burst Size (USB_*n*BURSTSIZE) .

54.7.6 RX Buffer Hardware Parameters (USB_*n*HWRXBUF)

Address: 4033_0000h base + 14h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																RXADD								RXBURST							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

USB_*n*HWRXBUF field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15–8 RXADD	Buffer total size for all receive endpoints is (2^RXADD). RX Buffer size is: (2^RXADD) * 4 Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. There is a single Receive FIFO buffer in the USB controller. The buffer is shared for all endpoints for the OTG controller in device mode.
RXBURST	Default burst size for memory to RX buffer transfer. This is reset value of RXPBURST bits in USB core regsiter USB_n_BURSTSIZE. Please see Programmable Burst Size (USB_*n*BURSTSIZE) .

54.7.7 General Purpose Timer #0 Load (USB_*n*GPTIMER0LD)

This register controls load value of the count timer in register n_GPTIMER0CTRL. Please see General Purpose Timer #0 Controller (USB_*n*GPTIMER0CTRL) .

Address: 4033_0000h base + 80h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								GPTLD																							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*GPTIMER0LD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
GPTLD	General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration.

Table continues on the next page...

USB_*n*GPTIMER0LD field descriptions (continued)

Field	Description
	Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. **NOTE:** Max value is 0xFFFFFF or 16.777215 seconds.

54.7.8 General Purpose Timer #0 Controller (USB_*n*GPTIMER0CTRL)

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI0 bit in n_USBSTS register (See USB Status Register (USB_*n*USBSTS)), interrupt enable bit is TIE0 bit in n_USBINTR register. (See Interrupt Enable Register (USB_*n*USBINTR) .)

Address: 4033_0000h base + 84h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPTRUN	GPTRST	Reserved					GPTMODE	GPTCNT							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPTCNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*GPTIMER0CTRL field descriptions

Field	Description
31 GPTRUN	General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit. 0 Stop counting 1 Run
30 GPTRST	General Purpose Timer Reset 0 No action 1 Load counter value from GPTLD bits in n_GPTIMER0LD

Table continues on the next page...

USB_*n*GPTIMER0CTRL field descriptions (continued)

Field	Description
29–25 -	This field is reserved. Reserved
24 GPTMODE	General Purpose Timer Mode In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software; In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again. 0 One Shot Mode 1 Repeat Mode
GPTCNT	General Purpose Timer Counter. This field is the count value of the countdown timer.

54.7.9 General Purpose Timer #1 Load (USB_*n*GPTIMER1LD)

This register controls load value of the count timer in register n_GPTIMER1CTRL. Please see General Purpose Timer #1 Controller (USB_*n*GPTIMER1CTRL) .

Address: 4033_0000h base + 88h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								GPTLD																							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*GPTIMER1LD field descriptions

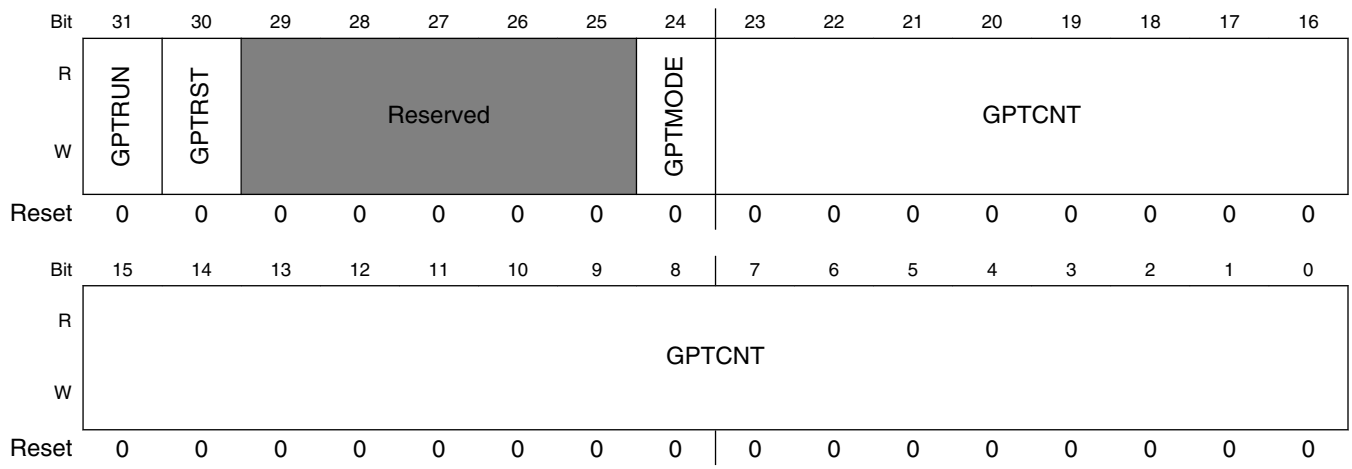
Field	Description
31–24 -	This field is reserved. Reserved
GPTLD	General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration. Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. **NOTE:** Max value is 0xFFFFFF or 16.777215 seconds.

54.7.10 General Purpose Timer #1 Controller (USB_*n*GPTIMER1CTRL)

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI1 bit in USB_n_USBSTS register (See USB Status Register (USB_*n*USBSTS)), interrupt enable bit is TIE1 bit in n_USBINTR register (See Interrupt Enable Register (USB_*n*USBINTR)).

Address: 4033_0000h base + 8Ch offset + (65536d × i), where i=0d to 1d



USB_*n*GPTIMER1CTRL field descriptions

Field	Description
31 GPTRUN	General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit. 0 Stop counting 1 Run
30 GPTRST	General Purpose Timer Reset 0 No action 1 Load counter value from GPTLD bits in USB_n_GPTIMER0LD
29–25 -	This field is reserved. Reserved
24 GPTMODE	General Purpose Timer Mode In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software. In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again.

Table continues on the next page...

USB_*n*GPTIMER1CTRL field descriptions (continued)

Field	Description
	0 One Shot Mode 1 Repeat Mode
GPTCNT	General Purpose Timer Counter. This field is the count value of the countdown timer.

54.7.11 System Bus Config (USB_*n*SBUSCFG)

Address: 4033_0000h base + 90h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																													AHBBRST		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

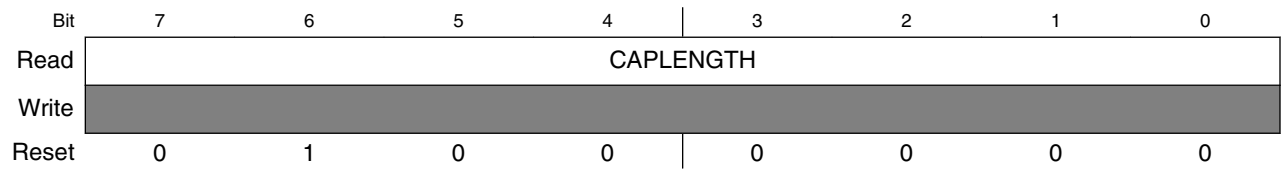
USB_*n*SBUSCFG field descriptions

Field	Description
31–3 -	This field is reserved. Reserved
AHBBRST	AHB master interface Burst configuration These bits control AHB master transfer type sequence (or priority). **NOTE:** This register overrides n_BURSTSIZE register when its value is not zero. 000 Incremental burst of unspecified length only 001 INCR4 burst, then single transfer 010 INCR8 burst, INCR4 burst, then single transfer 011 INCR16 burst, INCR8 burst, INCR4 burst, then single transfer 100 Reserved, don't use 101 INCR4 burst, then incremental burst of unspecified length 110 INCR8 burst, INCR4 burst, then incremental burst of unspecified length 111 INCR16 burst, INCR8 burst, INCR4 burst, then incremental burst of unspecified length

54.7.12 Capability Registers Length (USB_*n*CAPLENGTH)

The Capability Registers Length register contains the address offset to the Operational registers relative to the CAPLENGTH register.

Address: 4033_0000h base + 100h offset + (65536d × i), where i=0d to 1d



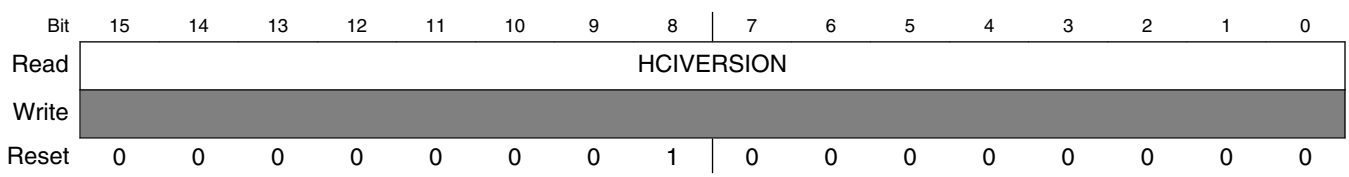
USB_*n*CAPLENGTH field descriptions

Field	Description
CAPLENGTH	These bits are used as an offset to add to register base to find the beginning of the Operational Register. Default value is '40h'.

54.7.13 Host Controller Interface Version (USB_*n*HCIVERSION)

This is a 2-byte register containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this register represents a major revision and the least significant byte is the minor revision.

Address: 4033_0000h base + 102h offset + (65536d × i), where i=0d to 1d



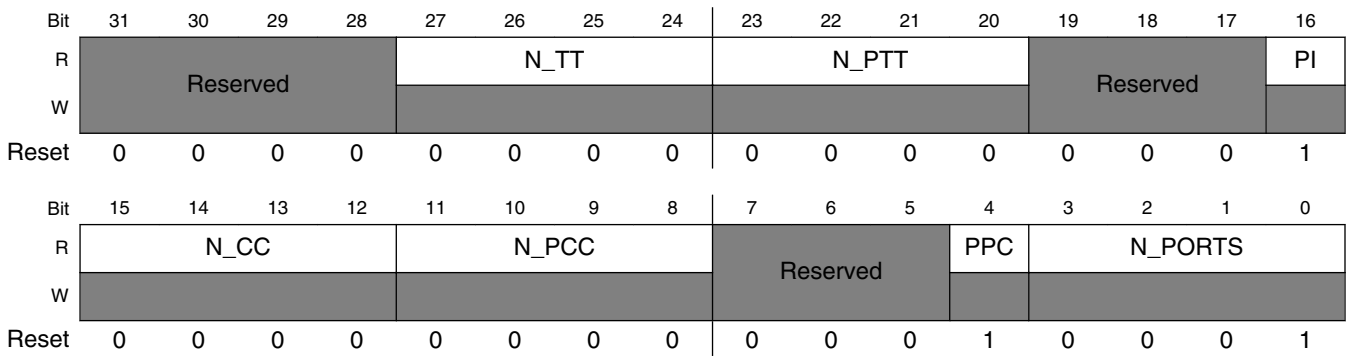
USB_*n*HCIVERSION field descriptions

Field	Description
HCIVERSION	Host Controller Interface Version Number Default value is '10h', which means EHCI rev1.0.

54.7.14 Host Controller Structural Parameters (USB_*n*HCSPARAMS)

The following figure shows the port steering logic capabilities of Host Control Structural Parameters (n_HCSPARAMS).

Address: 4033_0000h base + 104h offset + (65536d × i), where i=0d to 1d



USB_*n*HCSPARAMS field descriptions

Field	Description
31–28 -	This field is reserved. Reserved
27–24 N_TT	Number of Transaction Translators (N_TT). Default value '0000b' This field indicates the number of embedded transaction translators associated with the USB2.0 host controller. These bits would be set to '0001b' for Multi-Port Host, and '0000b' for Single-Port Host.
23–20 N_PTT	Number of Ports per Transaction Translator (N_PTT). Default value '0000b' This field indicates the number of ports assigned to each transaction translator within the USB2.0 host controller. These bits would be set to equal N_PORTS for Multi-Port Host, and '0000b' for Single-Port Host.
19–17 -	This field is reserved. Reserved
16 PI	Port Indicators (P INDICATOR) This bit indicates whether the ports support port indicator control. When set to one, the port status and control registers include a read/writeable field for controlling the state of the port indicator This bit is "1b" in all controller core.
15–12 N_CC	Number of Companion Controller (N_CC). This field indicates the number of companion controllers associated with this USB2.0 host controller. These bits are '0000b' in all controller core. 0 There is no internal Companion Controller and port-ownership hand-off is not supported. 1 There are internal companion controller(s) and port-ownership hand-offs is supported.
11–8 N_PCC	Number of Ports per Companion Controller

Table continues on the next page...

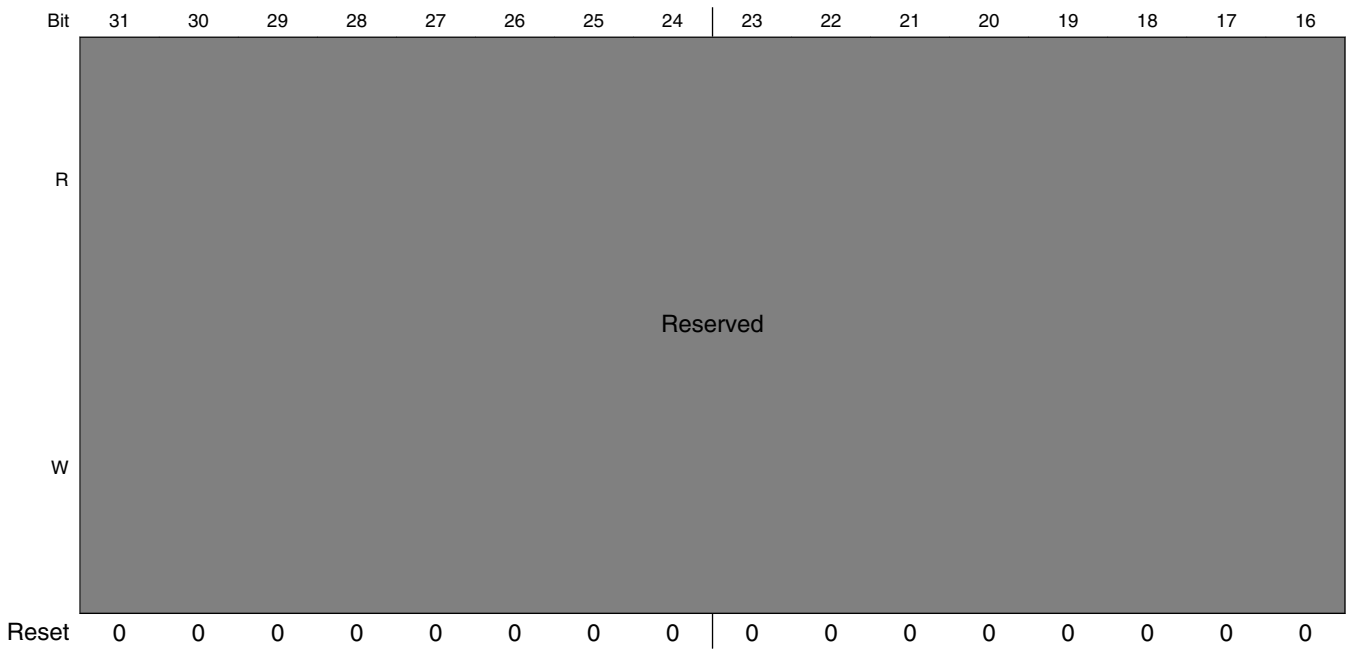
USB_*n*HCSPARAMS field descriptions (continued)

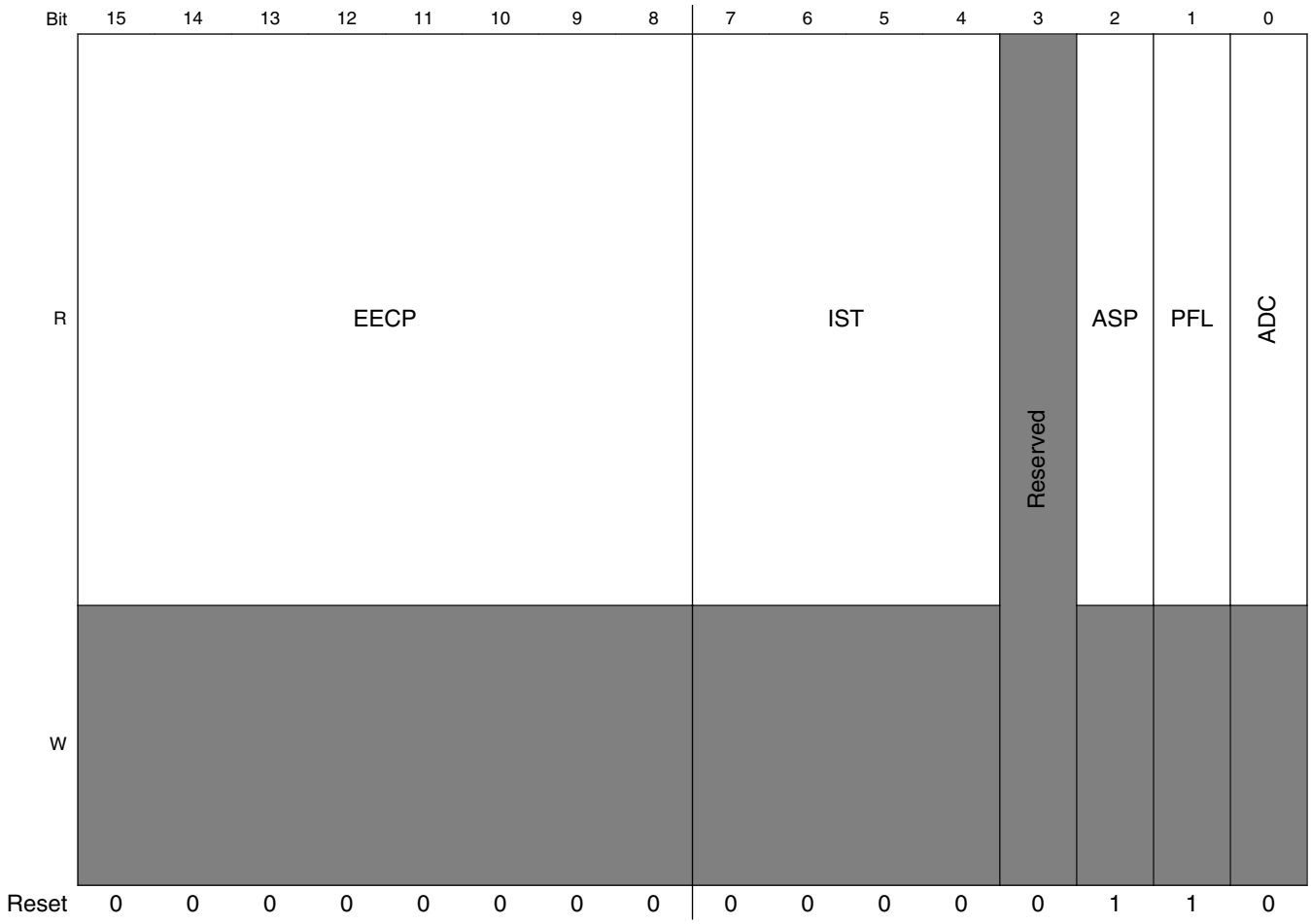
Field	Description
	This field indicates the number of ports supported per internal Companion Controller. It is used to indicate the port routing configuration to the system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC. These bits are '0000b' in all controller core.
7–5 -	This field is reserved. Reserved
4 PPC	Port Power Control This field indicates whether the host controller implementation includes port power control. A one indicates the ports have port power switches. A zero indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register
N_PORTS	Number of downstream ports. This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register. Valid values are in the range of 1h to Fh. A zero in this field is undefined. These bits are always set to '0001b' because all controller cores are Single-Port Host.

54.7.15 Host Controller Capability Parameters (USB_*n*HCCPARAMS)

This register identifies multiple mode control (time-base bit functionality), addressing capability.

Address: 4033_0000h base + 108h offset + (65536d × i), where i=0d to 1d





USB_*n*HCCPARAMS field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15–8 EECP	EHCI Extended Capabilities Pointer. This field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device. **NOTE:** These bits are set '00h' in all controller core.
7–4 IST	Isochronous Scheduling Threshold. This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is a one, then host software assumes the host controller may cache an isochronous data structure for an entire frame. These bits are set '00h' in all controller core.
3 -	This field is reserved. Reserved

Table continues on the next page...

USB_*n*HCCPARAMS field descriptions (continued)

Field	Description
2 ASP	Asynchronous Schedule Park Capability If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USBCMD register. **NOTE:** ASP bit reset value: '00b' for OTG controller core, '11b' for Host-only controller core.
1 PFL	Programmable Frame List Flag If this bit is set to zero, then the system software must use a frame list length of 1024 elements with this host controller. The USBCMD register Frame List Size field is a read-only register and must be set to zero. If set to a one, then the system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K-page boundary. This requirement ensures that the frame list is always physically contiguous. This bit is set '1b' in all controller core.
0 ADC	64-bit Addressing Capability This bit is set '0b' in all controller core, no 64-bit addressing capability is supported.

54.7.16 Device Controller Interface Version (USB_*n*DCIVERSION)

This register indicates the two-byte BCD encoding of the device controller interface version number.

Address: 4033_0000h base + 120h offset + (65536d × i), where i=0d to 1d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	DCIVERSION															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

USB_*n*DCIVERSION field descriptions

Field	Description
DCIVERSION	Device Controller Interface Version Number Default value is '01h', which means rev0.1.

54.7.17 Device Controller Capability Parameters (USB_*n*DCCPARAMS)

These fields describe the overall device capability of the controller.

NOTE

This register is only available in OTG controller core.

Address: 4033_0000h base + 124h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							HC	DC	Reserved		DEN				
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0

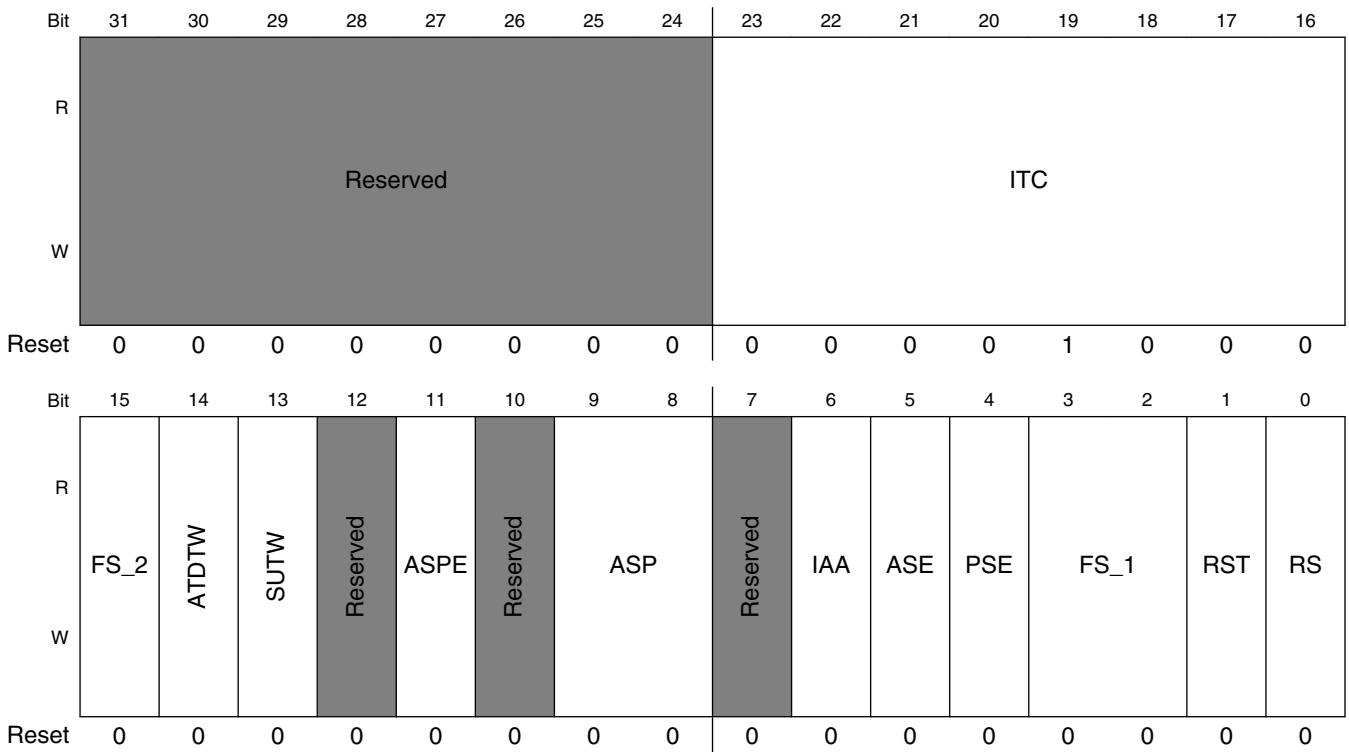
USB_*n*DCCPARAMS field descriptions

Field	Description
31–9 -	This field is reserved. Reserved
8 HC	Host Capable When this bit is 1, this controller is capable of operating as an EHCI compatible USB 2.0 host controller.
7 DC	Device Capable When this bit is 1, this controller is capable of operating as a USB 2.0 device.
6–5 -	This field is reserved. Reserved
DEN	Device Endpoint Number This field indicates the number of endpoints built into the device controller. If this controller is not device capable, then this field will be zero. Valid values are 0 - 15.

54.7.18 USB Command Register (USB_*n*USBCMD)

The Command Register indicates the command to be executed by the serial bus host/device controller. Writing to the register causes a command to be executed.

Address: 4033_0000h base + 140h offset + (65536d × i), where i=0d to 1d



USB_*n*USBCMD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 ITC	Interrupt Threshold Control -Read/Write. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. Value Maximum Interrupt Interval 0x00 Immediate (no threshold) 0x01 1 micro-frame 0x02 2 micro-frames 0x04 4 micro-frames 0x08 8 micro-frames 0x10 16 micro-frames

Table continues on the next page...

USB_*n*USBCMD field descriptions (continued)

Field	Description
	0x20 32 micro-frames 0x40 64 micro-frames
15 FS_2	Frame List Size - (Read/Write or Read Only). [host mode only] This field is Read/Write only if Programmable Frame List Flag in the HCCPARAMS registers is set to one. This field specifies the size of the frame list that controls which bits in the Frame Index Register should be used for the Frame List Current index. **NOTE:** This field is made up from USBCMD bits 15, 3 and 2. Value Meaning 000 1024 elements (4096 bytes) Default value 001 512 elements (2048 bytes) 010 256 elements (1024 bytes) 011 128 elements (512 bytes) 100 64 elements (256 bytes) 101 32 elements (128 bytes) 110 16 elements (64 bytes) 111 8 elements (32 bytes)
14 ATDTW	Add dTD TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure proper addition of a new dTD to an active (primed) endpoint's linked list. This bit is set and cleared by software. This bit would also be cleared by hardware when state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized.
13 SUTW	Setup TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by the DCD without being corrupted. If the setup lockout mode is off (SLOM bit in USB core register n_USBMODE, see USB Device Mode (USB_*n*USBMODE)) then there is a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This bit is set and cleared by software. This bit would also be cleared by hardware when a hazard detected.
12 -	This field is reserved. Reserved
11 ASPE	Asynchronous Schedule Park Mode Enable - Read/Write. If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is one, Park mode is enabled. When this bit is a zero, Park mode is disabled. **NOTE:** ASPE bit reset value: '0b' for OTG controller core, '1b' for Host-only controller core.
10 -	This field is reserved. Reserved
9–8 ASP	Asynchronous Schedule Park Mode Count - Read/Write. If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this field defaults to 3h and is R/W. Otherwise it defaults to zero and is Read-Only. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the

Table continues on the next page...

USB_*n*USBCMD field descriptions (continued)

Field	Description
	Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a zero to this bit when *Park Mode Enable* is a one as this will result in undefined behavior. This field is set to 3h in all controller core.
7 -	This field is reserved. Reserved
6 IAA	Interrupt on Async Advance Doorbell - Read/Write. This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Sync Advance Enable bit in the USBINTR register is one, then the host controller will assert an interrupt at the next interrupt threshold. The host controller sets this bit to zero after it has set the Interrupt on Sync Advance status bit in the USBSTS register to one. Software should not write a one to this bit when the asynchronous schedule is inactive. Doing so will yield undefined results. This bit is only used in host mode. Writing a one to this bit when device mode is selected will have undefined results.
5 ASE	Asynchronous Schedule Enable - Read/Write. Default 0b. This bit controls whether the host controller skips processing the Asynchronous Schedule. Only the host controller uses this bit. Values Meaning 0 Do not process the Asynchronous Schedule. 1 Use the ASYNCLISTADDR register to access the Asynchronous Schedule.
4 PSE	Periodic Schedule Enable- Read/Write. Default 0b. This bit controls whether the host controller skips processing the Periodic Schedule. Only the host controller uses this bit. Values Meaning 0 Do not process the Periodic Schedule 1 Use the PERIODICLISTBASE register to access the Periodic Schedule.
3–2 FS_1	See description at bit 15
1 RST	Controller Reset (RESET) - Read/Write. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Host operation mode: When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior. Device operation mode:

Table continues on the next page...

USB_*n*USBCMD field descriptions (continued)

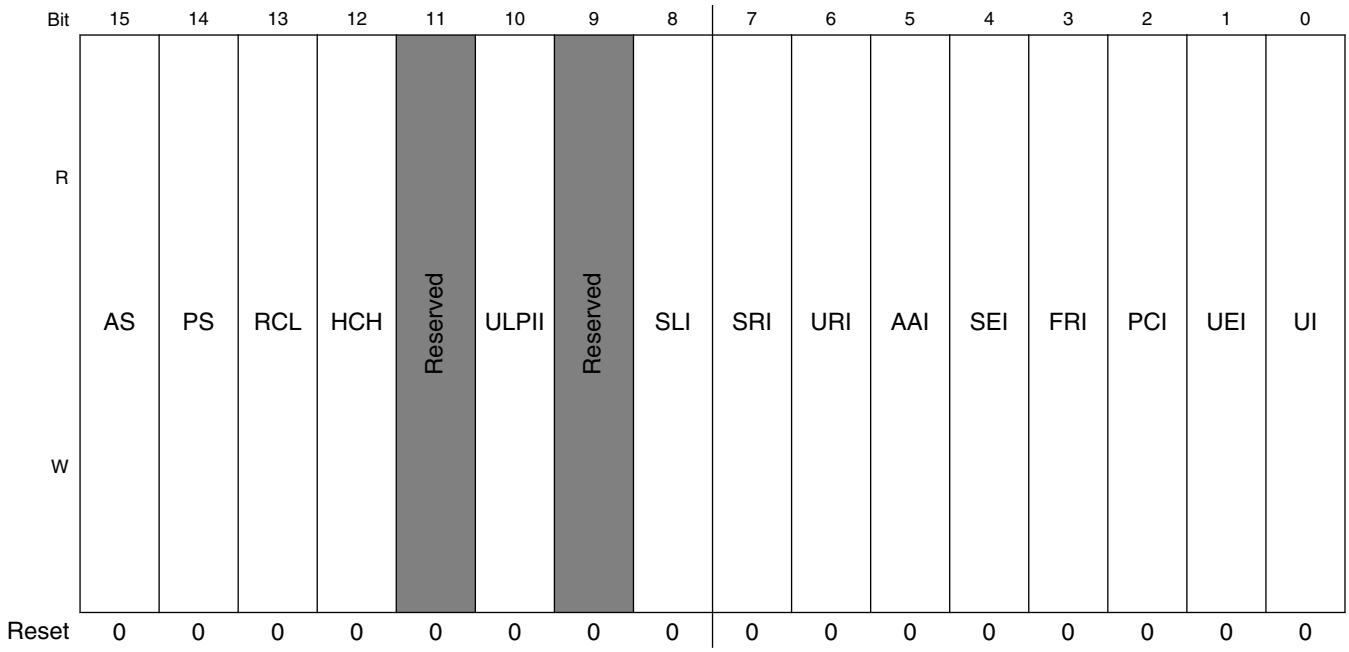
Field	Description
	When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Writing a one to this bit when the device is in the attached state is not recommended, because the effect on an attached host is undefined. In order to ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints should be flushed and the USBCMD Run/Stop bit should be set to 0.
0 RS	Run/Stop (RS) - Read/Write. Default 0b. 1=Run. 0=Stop. Host operation mode: When set to '1b', the Controller proceeds with the execution of the schedule. The Controller continues execution as long as this bit is set to a one. When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Controller has finished the transaction and has entered the stopped state. Software should not write a one to this field unless the controller is in the Halted state (that is, HCHalted in the USBSTS register is a one). Device operation mode: Writing a one to this bit will cause the controller to enable a pull-up on D+ and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up will become disabled upon transitioning into high-speed mode. Software should use this bit to prevent an attach event before the controller has been properly initialized. Writing a 0 to this will cause a detach event.

54.7.19 USB Status Register (USB_*n*USBSTS)

This register indicates various states of the Host/Device Controller and any pending interrupts. This register does not indicate status resulting from a transaction on the serial bus.

Address: 4033_0000h base + 144h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved						TI1	TI0	Reserved				UPI	UAI	Reserved	NAKI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USB_*n*USBSTS field descriptions

Field	Description
31–26 -	This field is reserved. Reserved
25 TI1	General Purpose Timer Interrupt 1(GPTINT1)--R/WC. This bit is set when the counter in the GPTIMER1CTRL register transitions to zero, writing a one to this bit will clear it.
24 TI0	General Purpose Timer Interrupt 0(GPTINT0)--R/WC. This bit is set when the counter in the GPTIMER0CTRL register transitions to zero, writing a one to this bit clears it.
23–20 -	This field is reserved. Reserved
19 UPI	USB Host Periodic Interrupt (USBHSTPERINT) This bit is set by the Host Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set and the TD was from the periodic schedule. This bit is also set by the Host Controller when a short packet is detected AND the packet is on the periodic schedule. A short packet is when the actual number of bytes received was less than the expected number of bytes. This bit is not used by the device controller and will always be 0.
18 UAI	USB Host Asynchronous Interrupt (USBHSTASYNCINT) This bit is set by the Host Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set and the TD was from the asynchronous schedule. This bit is also set by the Host Controller when a short packet is detected AND the packet is on the asynchronous schedule. A short packet is when the actual number of bytes received was less than the expected number of bytes. This bit is not used by the device controller and will always be 0.
17 -	This field is reserved. Reserved

Table continues on the next page...

USB_*n*USBSTS field descriptions (continued)

Field	Description
16 NAKI	NAK Interrupt Bit--RO. This bit is set by hardware when for a particular endpoint both the TX/RX Endpoint NAK bit and corresponding TX/RX Endpoint NAK Enable bit are set. This bit is automatically cleared by hardware when all Enabled TX/RX Endpoint NAK bits are cleared.
15 AS	Asynchronous Schedule Status - Read Only. This bit reports the current real status of the Asynchronous Schedule. When set to zero the asynchronous schedule status is disabled and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0). Only used in the host operation mode.
14 PS	Periodic Schedule Status - Read Only. This bit reports the current real status of the Periodic Schedule. When set to zero the periodic schedule is disabled, and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0). Only used in the host operation mode.
13 RCL	Reclamation - Read Only. This is a read-only status bit used to detect an empty asynchronous schedule. Only used in the host operation mode.
12 HCH	HCHalted - Read Only. This bit is a zero whenever the Run/Stop bit is a one. The Controller sets this bit to one after it has stopped executing because of the Run/Stop bit being set to 0, either by software or by the Controller hardware (for example, an internal error). Only used in the host operation mode. Default value is '0b' for OTG core, and '1b' for Host1/Host2/Host3 core. This is because OTG core is not operating as host in default. Please see CM bit in USB_n_USBMODE register. **NOTE:** HCH bit reset value: '0b' for OTG controller core, '1b' for Host-only controller core.
11 -	This field is reserved. Reserved
10 ULPII	ULPI Interrupt - R/WC. This bit will be set '1b' by hardware when there is an event completion in ULPI viewport. This bit is usable only if the controller support UPLI interface mode.
9 -	This field is reserved. Reserved
8 SLI	DCSuspend - R/WC. When a controller enters a suspend state from an active state, this bit will be set to a one. The device controller clears the bit upon exiting from a suspend state. Only used in device operation mode.
7 SRI	SOF Received - R/WC.

Table continues on the next page...

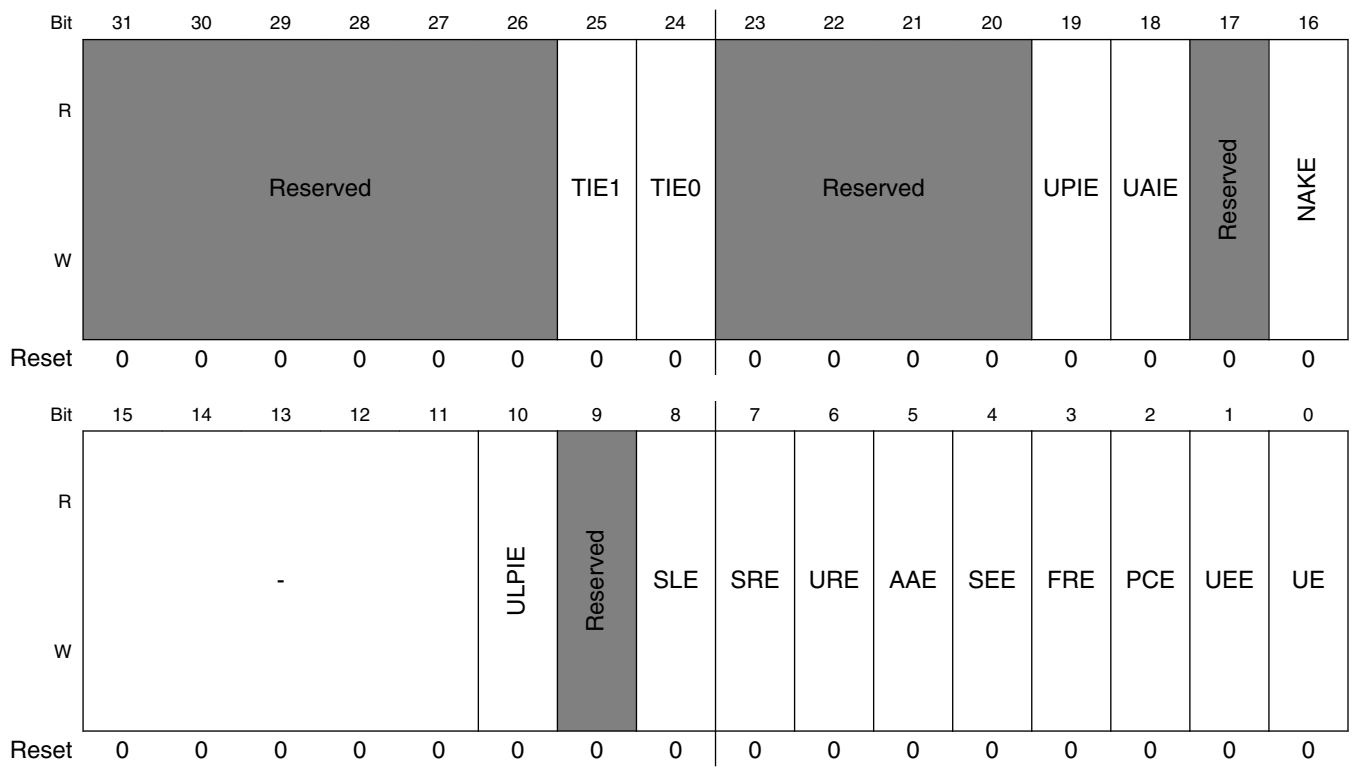
USB_*n*USBSTS field descriptions (continued)

Field	Description
	When the device controller detects a Start Of (micro) Frame, this bit will be set to a one. When a SOF is extremely late, the device controller will automatically set this bit to indicate that an SOF was expected. Therefore, this bit will be set roughly every 1ms in device FS mode and every 125ms in HS mode and will be synchronized to the actual SOF that is received. Because the device controller is initialized to FS before connect, this bit will be set at an interval of 1ms during the prelude to connect and chirp. In host mode, this bit will be set every 125us and can be used by host controller driver as a time base. Software writes a 1 to this bit to clear it.
6 URI	USB Reset Received - R/WC. When the device controller detects a USB Reset and enters the default state, this bit will be set to a one. Software can write a 1 to this bit to clear the USB Reset Received status bit. Only used in device operation mode.
5 AAI	Interrupt on Async Advance - R/WC. System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the n_USBCMD register. This status bit indicates the assertion of that interrupt source. Only used in host operation mode.
4 SEI	System Error- R/WC. This bit is will be set to '1b' when an Error response is seen to a read on the system interface.
3 FRI	Frame List Rollover - R/WC. The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example. If the frame list size (as programmed in the Frame List Size field of the USB_n_USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX [13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FHINDEX [12] toggles. Only used in host operation mode.
2 PCI	Port Change Detect - R/WC. The Host Controller sets this bit to a one when on any port a Connect Status occurs, a Port Enable/ Disable Change occurs, or the Force Port Resume bit is set as the result of a J-K transition on the suspended port. The Device Controller sets this bit to a one when the port controller enters the full or high-speed operational state. When the port controller exits the full or high-speed operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received bit and the DCSuspend bits respectively.
1 UEI	USB Error Interrupt (USBERRINT) - R/WC. When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set. The device controller detects resume signaling only.
0 UI	USB Interrupt (USBINT) - R/WC. This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.

54.7.20 Interrupt Enable Register (USB_*n*USBINTR)

The interrupts to software are enabled with this register. An interrupt is generated when a bit is set and the corresponding interrupt source is active. The USB Status register (n_USBSTS) still shows interrupt sources even if they are disabled by the n_USBINTR register, allowing polling of interrupt events by the software.

Address: 4033_0000h base + 148h offset + (65536d × i), where i=0d to 1d



USB_*n*USBINTR field descriptions

Field	Description
31–26 -	This field is reserved. Reserved
25 TIE1	General Purpose Timer #1 Interrupt Enable When this bit is one and the TI1 bit in n_USBSTS register is a one the controller will issue an interrupt.
24 TIE0	General Purpose Timer #0 Interrupt Enable When this bit is one and the TI0 bit in n_USBSTS register is a one the controller will issue an interrupt.
23–20 -	This field is reserved. Reserved
19 UPIE	USB Host Periodic Interrupt Enable

Table continues on the next page...

USB_*n*USBINTR field descriptions (continued)

Field	Description
	When this bit is one, and the UPI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.
18 UAIE	USB Host Asynchronous Interrupt Enable When this bit is one, and the UAI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.
17 -	This field is reserved. Reserved
16 NAKE	NAK Interrupt Enable When this bit is one and the NAKI bit in n_USBSTS register is a one the controller will issue an interrupt.
15–11 -	These bits are reserved and should be set to zero.
10 ULPIE	ULPI Interrupt Enable When this bit is one and the UPLII bit in n_USBSTS register is a one the controller will issue an interrupt. This bit is usable only if the controller support UPLI interface mode.
9 -	This field is reserved. Reserved
8 SLE	Sleep Interrupt Enable When this bit is one and the SLI bit in n_n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
7 SRE	SOF Received Interrupt Enable When this bit is one and the SRI bit in n_USBSTS register is a one the controller will issue an interrupt.
6 URE	USB Reset Interrupt Enable When this bit is one and the URI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
5 AAE	Async Advance Interrupt Enable When this bit is one and the AAI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
4 SEE	System Error Interrupt Enable When this bit is one and the SEI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
3 FRE	Frame List Rollover Interrupt Enable When this bit is one and the FRI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
2 PCE	Port Change Detect Interrupt Enable When this bit is one and the PCI bit in n_USBSTS register is a one the controller will issue an interrupt.
1 UEE	USB Error Interrupt Enable When this bit is one and the UEI bit in n_USBSTS register is a one the controller will issue an interrupt.
0 UE	USB Interrupt Enable When this bit is one and the UI bit in n_USBSTS register is a one the controller will issue an interrupt.

54.7.21 USB Frame Index (USB_*n*FRINDEX)

This register is used by the host controller to index the periodic frame list. The register updates every 125 microseconds (once each micro-frame). Bits [N: 3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the n_USBCMD register.

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the 'Halted' state as indicated by the HCHalted bit. A write to this register while the Run/Stop hit is set to a one produces undefined results. Writes to this register also affect the SOF value.

In device mode this register is read only and, the device controller updates the FRINDEX [13:3] register from the frame number indicated by the SOF marker. Whenever a SOF is received by the USB bus, FRINDEX [13:3] will be checked against the SOF marker. If FRINDEX [13:3] is different from the SOF marker, FRINDEX [13:3] will be set to the SOF value and FRINDEX [2:0] will be set to zero (that is, SOF for 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX [2:0] will be increment (that is, SOF for 125 us micro-frame.).

Address: 4033_0000h base + 14Ch offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																FRINDEX															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*FRINDEX field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
FRINDEX	Frame Index. The value, in this register, increments at the end of each time frame (micro-frame). Bits [N: 3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the USBCMD register, when used in host mode. USBCMD [Frame List Size] Number Elements N In device mode the value is the current frame number of the last frame transmitted. It is not used as an index. In either mode bits 2:0 indicate the current microframe. 000 (1024) 12 001 (512) 11

Table continues on the next page...

USB_*n*FRINDEX field descriptions (continued)

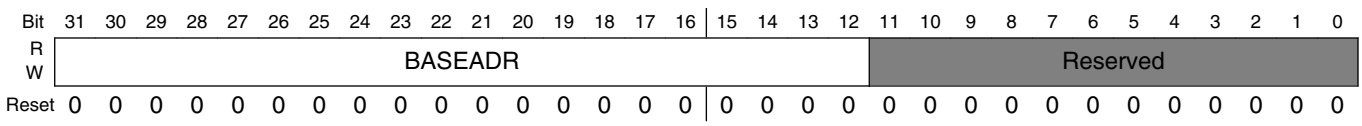
Field	Description
	010 (256) 10 011 (128) 9 100 (64) 8 101 (32) 7 110 (16) 6 111 (8) 5

54.7.22 Frame List Base Address (USB_*n*PERIODICLISTBASE)

Host Controller only

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. HCD loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (USB_n_FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

Address: 4033_0000h base + 154h offset + (65536d × i), where i=0d to 1d



USB_*n*PERIODICLISTBASE field descriptions

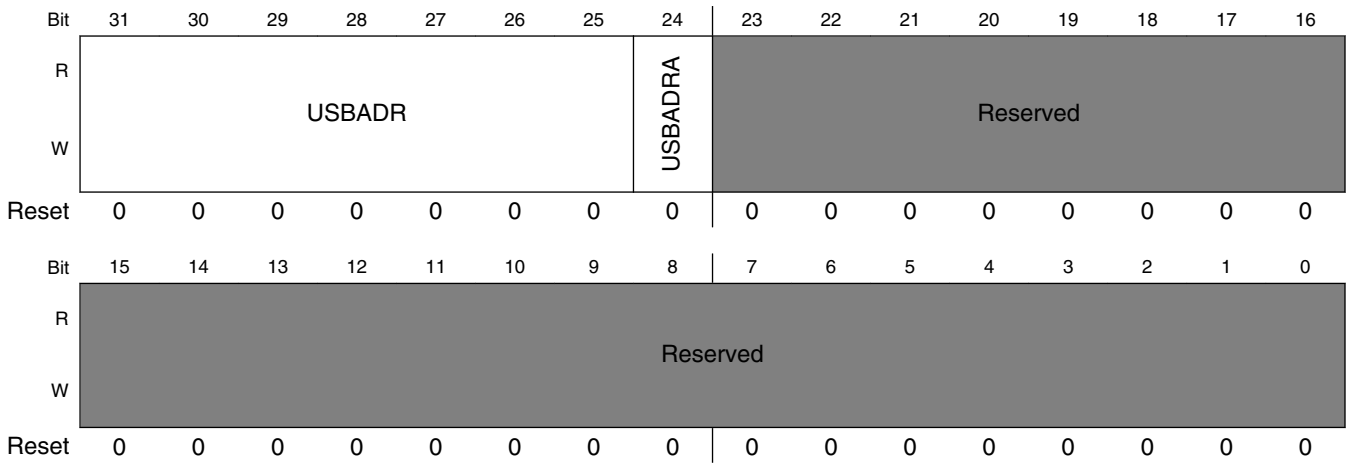
Field	Description
31–12 BASEADR	Base Address (Low). These bits correspond to memory address signals [31:12], respectively. Only used by the host controller.
-	This field is reserved. Reserved

54.7.23 Device Address (USB_*n*DEVICEADDR)

Device Controller only

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. Software shall reprogram the address after receiving a SET_ADDRESS descriptor.

Address: 4033_0000h base + 154h offset + (65536d × i), where i=0d to 1d



USB_*n*DEVICEADDR field descriptions

Field	Description
31–25 USBADR	Device Address. These bits correspond to the USB device address
24 USBADRA	Device Address Advance. Default=0. When this bit is '0', any writes to USBADR are instantaneous. When this bit is written to a '1' at the same time or before USBADR is written, the write to the USBADR field is staged and held in a hidden register. After an IN occurs on endpoint 0 and is ACKed, USBADR will be loaded from the holding register. Hardware will automatically clear this bit on the following conditions: 1) IN is ACKed to endpoint 0. (USBADR is updated from staging register). 2) OUT/SETUP occur to endpoint 0. (USBADR is not updated). 3) Device Reset occurs (USBADR is reset to 0). **NOTE:** After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism will ensure this specification is met when the DCD can not write of the device address within 2ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA=1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR will be programmed instantly at the correct time and meet the 2ms USB requirement.
-	This field is reserved. Reserved

54.7.24 Next Asynch. Address (USB_*n*ASYNCLISTADDR)

Host Controller only

This 32-bit register contains the address of the next asynchronous queue head to be executed by the host. Bits [4:0] of this register cannot be modified by the system software and will always return a zero when read.

Address: 4033_0000h base + 158h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ASYBASE																											Reserved				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ASYNCLISTADDR field descriptions

Field	Description
31–5 ASYBASE	Link Pointer Low (LPL). These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH). Only used by the host controller.
-	This field is reserved. Reserved

54.7.25 Endpoint List Address (USB_*n*ENDPTLISTADDR)

Device Controller only

In device mode, this register contains the address of the top of the endpoint list in system memory. Bits [10:0] of this register cannot be modified by the system software and will always return a zero when read.

The memory structure referenced by this physical memory pointer is assumed 64-byte.

Address: 4033_0000h base + 158h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EPBASE																					Reserved										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTLISTADDR field descriptions

Field	Description
31–11 EPBASE	Endpoint List Pointer(Low). These bits correspond to memory address signals [31:11], respectively. This field will reference a list of up to 32 Queue Head (QH) (that is, one queue head per endpoint & direction).
-	This field is reserved. Reserved

54.7.26 Programmable Burst Size (USB_*n*BURSTSIZE)

This register is used to control the burst size used during data movement on the AHB master interface. This register is ignored if AHBBRST bits in SBUSCFG register is non-zero value.

Address: 4033_0000h base + 160h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																TXPBURST								RXPBURST							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*BURSTSIZE field descriptions

Field	Description
31–17 -	This field is reserved. Reserved
16–8 TXPBURST	Programmable TX Burst Size. Default value is determined by TXBURST bits in n_HWTXBUF. This register represents the maximum length of a the burst in 32-bit words while moving data from system memory to the USB bus.
RXPBURST	Programmable RX Burst Size. Default value is determined by TXBURST bits in n_HWRXBUF. This register represents the maximum length of a the burst in 32-bit words while moving data from the USB bus to system memory.

54.7.27 TX FIFO Fill Tuning (USB_*n*TXFILLTUNING)

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include the how much data to post into the FIFO and an estimate for how long that operation should take in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_{ff} = Time to fetch packet into TX FIFO up to specified level.

T_s = Total Packet Flight Time (send-only) packet

$T_s = T_0 + T_1$

T_p = Total Packet Time (fetch and send) packet

$T_p = T_{ff} + T_0 + T_1$

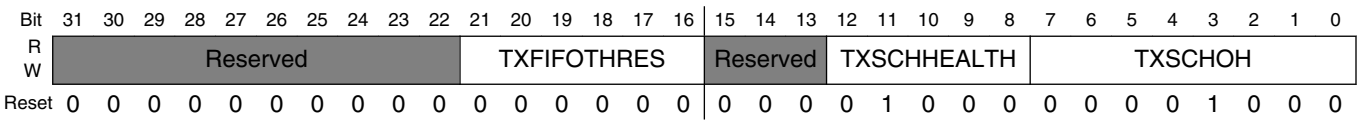
Upon discovery of a transmit (OUT/SETUP) packet in the data structures, host controller checks to ensure T_p remains before the end of the [micro]frame. If so it proceeds to pre-fill the TX FIFO. If at anytime during the pre-fill operation the time remaining the [micro]frame is < T_s then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller will eventually recover, a

mark will be made the scheduler health counter to note the occurrence of a "back-off" event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that will begin after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus should be minimized (not necessarily eliminated). Back-offs can be minimized with use of the n_TSCHHEALTH (T_{ff}) described below.

NOTE

The reset value could vary from instance to instance. Please see the detail in bit field description and ignore reset value in summary table in this case!

Address: 4033_0000h base + 164h offset + (65536d × i), where i=0d to 1d



USB_*n*TXFILLTUNING field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–16 TXFIFOTHRES	FIFO Burst Threshold. (Read/Write) This register controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on to the bus. The minimum value is 2 and this value should be a low as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can be replenished from system memory. This value is ignored if the Stream Disable bit in USB_n_USBMODE register is set. Default value is '00h' for OTG controller core, and '02h' for Host-only controller core.
15–13 -	This field is reserved. Reserved
12–8 TXSCHHEALTH	Scheduler Health Counter. (Read/Write To Clear) This register increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame. This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register will clear the counter and this counter will max. at 31. Default value is '08h' for OTG controller core, and '00h' for Host-only controller core.
TXSCHOH	Scheduler Overhead. (Read/Write) [Default = 0] This register adds an additional fixed offset to the schedule time estimator described above as Tff. As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a highly utilized bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267us when a device is connected in High-Speed Mode. The time unit represented in this register is 6.333us when a device is connected in Low/Full Speed Mode. Default value is '08h' for OTG controller core, and '00h' for Host-only controller core.

54.7.28 Endpoint NAK (USB_*n*ENDPTNAK)

Address: 4033_0000h base + 178h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								EPTN								Reserved								EPRN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTNAK field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 EPTN	TX Endpoint NAK - R/WC. Each TX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received IN token for the corresponding endpoint. Bit [N] - Endpoint #[N], N is 0-7
15–8 -	This field is reserved. Reserved
EPRN	RX Endpoint NAK - R/WC. Each RX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received OUT or PING token for the corresponding endpoint. Bit [N] - Endpoint #[N], N is 0-7

54.7.29 Endpoint NAK Enable (USB_*n*ENDPTNAKEN)

Address: 4033_0000h base + 17Ch offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								EPTNE								Reserved								EPRNE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTNAKEN field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 EPTNE	TX Endpoint NAK Enable - R/W. Each bit is an enable bit for the corresponding TX Endpoint NAK bit. If this bit is set and the corresponding TX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7
15–8 -	This field is reserved. Reserved
EPRNE	RX Endpoint NAK Enable - R/W.

Table continues on the next page...

USB_*n*ENDPTNAKEN field descriptions (continued)

Field	Description
	Each bit is an enable bit for the corresponding RX Endpoint NAK bit. If this bit is set and the corresponding RX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7

54.7.30 Configure Flag Register (USB_*n*CONFIGFLAG)

Address: 4033_0000h base + 180h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															CF
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

USB_*n*CONFIGFLAG field descriptions

Field	Description
31–1 -	This field is reserved. Reserved
0 CF	Configure Flag Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. 0 Port routing control logic default-routes each port to an implementation dependent classic host controller. 1 Port routing control logic default-routes all ports to this host controller.

54.7.31 Port Status & Control (USB_*n*PORTSC1)

Host Controller

A host controller could implement one to eight port status and control registers. The number is determined by N_PORTs bits in HWSPARAMs register (please see Host Controller Structural Parameters (USB_*n*HCSPARAMS)). Software could read this parameter register to determine how many ports need service.

All controller cores on this product are Single-Port, so there is only one port status and control register for each controller core.

This register is only reset by power on reset or controller core reset. The initial conditions of a port are:

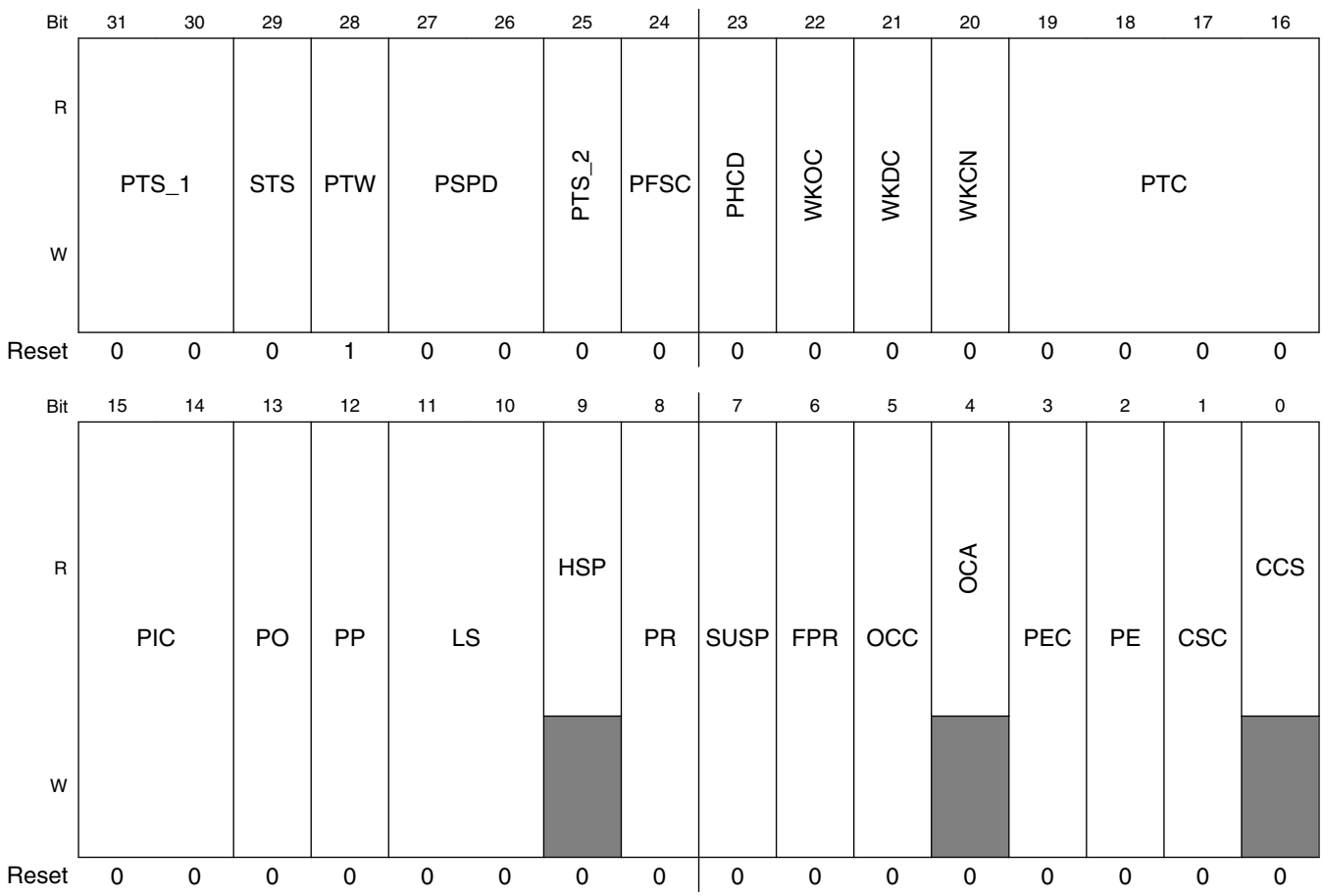
- No device connected
- Port disabled

If the port supports power control, this state remains until port power is supplied (by software).

Device Controller

A controller operating in device mode has only port register one (PORTSC1) and it does not support power control in that mode. Port control in device mode is only used for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling and allows software to put the PHY into low power suspend mode and disable the PHY clock.

Address: 4033_0000h base + 184h offset + (65536d × i), where i=0d to 1d



USB_*n*PORTSC1 field descriptions

Field	Description
31–30 PTS_1	Bit field {bit25, bit31, bit30}: "000b" UTMI/UTMI+ "001b" Reserved "010b" ULPI "011b" Serial/USB 1.1 PHY/IC-USB (FS Only) "100b" HSIC **NOTE:** All USB port interface modes are listed in this field description, but not all are supported. For detail feature of each controller core, please see Features . The behaviour is unknown when unsupported interface mode is selected.
29 STS	Serial Transceiver Select - Read Only Serial Transceiver Select 1 Serial Interface Engine is selected 0 Parallel Interface signals is selected Serial Interface Engine can be used in combination with UTMI+/ULPI physical interface to provide FS/LS signaling instead of the parallel interface signals. When this bit is set '1b', serial interface engine will be used instead of parallel interface signals. This bit has no effect unless PTS bits is set to select UTMI+/ULPI interface. The Serial/USB1.1 PHY/IC-USB will use the serial interface engine for FS/LS signaling regardless of this bit value.
28 PTW	Parallel Transceiver Width This bit has no effect if serial interface engine is used. For OTG1/OTG2/Host1 core, it is Read-Only. Reset value is '1b'. 0 Select the 8-bit UTMI interface [60MHz] 1 Select the 16-bit UTMI interface [30MHz]
27–26 PSPD	Port Speed - Read Only. This register field indicates the speed at which the port is operating. 00 Full Speed 01 Low Speed 10 High Speed 11 Undefined
25 PTS_2	See description at bits 31-30
24 PFSC	Port Force Full Speed Connect - Read/Write. Default = 0b. When this bit is set to '1b', the port will be forced to only connect at Full Speed, It disables the chirp sequence that allows the port to identify itself as High Speed. 1 Forced to full speed 0 Normal operation
23 PHCD	PHY Low Power Suspend - Clock Disable (PLPSCD) - Read/Write. Default = 0b. When this bit is set to '1b', the PHY clock is disabled. Reading this bit will indicate the status of the PHY clock.

Table continues on the next page...

USB_*n*PORTSC1 field descriptions (continued)

Field	Description
	NOTE: The PHY clock cannot be disabled if it is being used as the system clock. In device mode, The PHY can be put into Low Power Suspend when the device is not running (USBCMD Run/Stop=0b) or the host has signalled suspend (PORTSC1 SUSPEND=1b). PHY Low power suspend will be cleared automatically when the host initials resume. Before forcing a resume from the device, the device controller driver must clear this bit. In host mode, the PHY can be put into Low Power Suspend when the downstream device has been put into suspend mode or when no downstream device is connected. Low power suspend is completely under the control of software. 1 Disable PHY clock 0 Enable PHY clock
22 WKOC	Wake on Over-current Enable (WKOC_E) - Read/Write. Default = 0b. Writing this bit to a one enables the port to be sensitive to over-current conditions as wake-up events. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero.
21 WKDC	Wake on Disconnect Enable (WKDSCNNT_E) - Read/Write. Default=0b. Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero or in device mode.
20 WKCN	Wake on Connect Enable (WKCNNT_E) - Read/Write. Default=0b. Writing this bit to a one enables the port to be sensitive to device connects as wake-up events. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero or in device mode.
19–16 PTC	Port Test Control - Read/Write. Default = 0000b. Refer to Port Test Mode for the operational model for using these test modes and the USB Specification Revision 2.0, Chapter 7 for details on each test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS/FS/LS} values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point. **NOTE:** *Low speed operations are not supported as a peripheral device.* Any other value than zero indicates that the port is operating in test mode. Value Specific Test 0000 TEST_MODE_DISABLE 0001 J_STATE 0010 K_STATE 0011 SE0 (host) / NAK (device) 0100 Packet 0101 FORCE_ENABLE_HS 0110 FORCE_ENABLE_FS 0111 FORCE_ENABLE_LS 1000-1111 Reserved
15–14 PIC	Port Indicator Control - Read/Write. Default = Ob. Writing to this field has no effect if the P_INDICATOR bit in the HCSPARAMS register is a zero. Refer to the USB Specification Revision 2.0 for a description on how these bits are to be used. This field is zero if *Port Power* is zero.

Table continues on the next page...

USB_*n*PORTSC1 field descriptions (continued)

Field	Description
	Bit Value Meaning 00 Port indicators are off 01 Amber 10 Green 11 Undefined
13 PO	Port Owner-Read/Write. Default = 0. This bit unconditionally goes to a 0 when the configured bit in the CONFIGFLAG register makes a 0 to 1 transition. This bit unconditionally goes to 1 whenever the Configured bit is zero System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that an internal companion controller owns and controls the port. Port owner handoff is not supported in all controller cores, therefore this bit will always be 0.
12 PP	Port Power (PP)-Read/Write or Read Only. The function of this bit depends on the value of the Port Power Switching (PPC) field in the HCSPARAMS register. The behavior is as follows: PPC PP Operation 0 1b *Read Only - Host controller does not have port power control switches. Each port is hard-wired to power.* 1 1b/0b - *Read/Write. Host/OTG controller requires port power control switches. This bit represents the current setting of the switch (0=off, 1=on). When power is not available on a port (that is, PP equals a 0), the port is non-functional and will not report attaches, detaches, etc.* When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitional by the host controller driver from a one to a zero (removing power from the port). This feature is implemented in all controller cores (PPC = 1).
11–10 LS	Line Status-Read Only. These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. In host mode, the use of linestate by the host controller driver is not necessary (unlike EHCI), because the port controller state machine and the port routing manage the connection of LS and FS. In device mode, the use of linestate by the device controller driver is not necessary. The encoding of the bits are: Bits [11:10] Meaning 00 SE0 10 J-state 01 K-state 11 Undefined
9 HSP	High-Speed Port - Read Only. Default = 0b. When the bit is one, the host/device connected to the port is in high-speed mode and if set to zero, the host/device connected to the port is not in a high-speed mode.

Table continues on the next page...

USB_*n*PORTSC1 field descriptions (continued)

Field	Description
	NOTE: HSP is redundant with PSPD(bit 27, 26) but remained for compatibility.
8 PR	Port Reset - Read/Write or Read Only. Default = 0b. In Host Mode: Read/Write. 1=Port is in Reset. 0=Port is not in Reset. Default 0. When software writes a one to this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 is started. *This bit will automatically change to zero after the reset sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the reset duration is timed in the driver.* In Device Mode: This bit is a read only status bit. Device reset from the USB bus is also indicated in the USBSTS register. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero.
7 SUSP	Suspend - Read/Write or Read Only. Default = 0b. 1=Port in suspend state. 0=Port not in suspend state. In Host Mode: Read/Write. Port Enabled Bit and Suspend bit of this register define the port states as follows: Bits [Port Enabled, Suspend] Port State 0x Disable 10 Enable 11 Suspend When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. The host controller will unconditionally set this bit to zero when software sets the *Force Port Resume* bit to zero. The host controller ignores a write of zero to this bit. If host software sets this bit to a one when the port is not enabled (that is, *Port enabled* bit is a zero) the results are undefined. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: Read Only. In device mode this bit is a read only status bit.
6 FPR	Force Port Resume -Read/Write. 1= Resume detected/driven on port. 0=No resume (K-state) detected/driven on port. Default = 0. In Host Mode: Software sets this bit to one to drive resume signaling. The Host Controller sets this bit to one if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the *Port Change Detect* bit in the USBSTS register is also set to one. *This bit will automatically change to zero after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the resume duration is timed in the driver.* Note that when the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. This bit will remain a one until the port has switched to the high-speed idle. Writing a zero has no effect because the port controller will time the resume operation, clear the bit the port control state switches to HS or FS idle.

Table continues on the next page...

USB_*n*PORTSC1 field descriptions (continued)

Field	Description
	This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. This bit is not-EHCI compatible. In Device mode: After the device has been in Suspend State for 5ms or more, software must set this bit to one to drive resume signaling before clearing. The Device Controller will set this bit to one if a J-to-K transition is detected while the port is in the Suspend state. The bit will be cleared when the device returns to normal operation. Also, when this bit wil be cleared because a K-to-J transition detected, the *Port Change Detect* bit in the USBSTS register is also set to one.
5 OCC	Over-current Change-R/WC. Default=0. This bit is set '1b' by hardware when there is a change to Over-current Active. Software can clear this bit by writing a one to this bit position.
4 OCA	Over-current Active-Read Only. Default 0. This bit will automatically transition from one to zero when the over current condition is removed. 1 This port currently has an over-current condition 0 This port does not have an over-current condition.
3 PEC	Port Enable/Disable Change-R/WC. 1=Port enabled/disabled status has changed. 0=No change. Default = 0. In Host Mode: For the root hub, this bit is set to a one only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). Software clears this by writing a one to it. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero. In Device mode: The device port is always enabled, so this bit is always '0b'.
2 PE	Port Enabled/Disabled-Read/Write. 1=Enable. 0=Disable. Default 0. In Host Mode: Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by the host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled, (0b) downstream propagation of data is blocked except for reset. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: The device port is always enabled, so this bit is always '1b'.
1 CSC	Connect Status Change-R/WC. 1 =Change in Current Connect Status. 0=No change. Default 0. In Host Mode: Indicates a change has occurred in the port's Current Connect Status. The host/device controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be 'setting' an already-set bit (that is, the bit will remain set). Software clears this bit by writing a one to it.

Table continues on the next page...

USB_*n*PORTSC1 field descriptions (continued)

Field	Description
	This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: This bit is undefined in device controller mode.
0 CCS	Current Connect Status-Read Only. In Host Mode: 1=Device is present on port. 0=No device is present. Default = 0. This value reflects the current state of the port, and may not correspond directly to the event that caused the *Connect Status Change* bit (Bit 1) to be set. This field is zero if *Port Power(*Port Status & Control (USB_*n*PORTSC1)*)* is zero in host mode. In Device Mode: 1=Attached. 0=Not Attached. Default=0. A one indicates that the device successfully attached and is operating in either high speed or full speed as indicated by the High Speed Port bit in this register. A zero indicates that the device did not attach successfully or was forcibly disconnected by the software writing a zero to the Run bit in the USBCMD register. It does not state the device being disconnected or suspended.

54.7.32 On-The-Go Status & control (USB_*n*OTGSC)

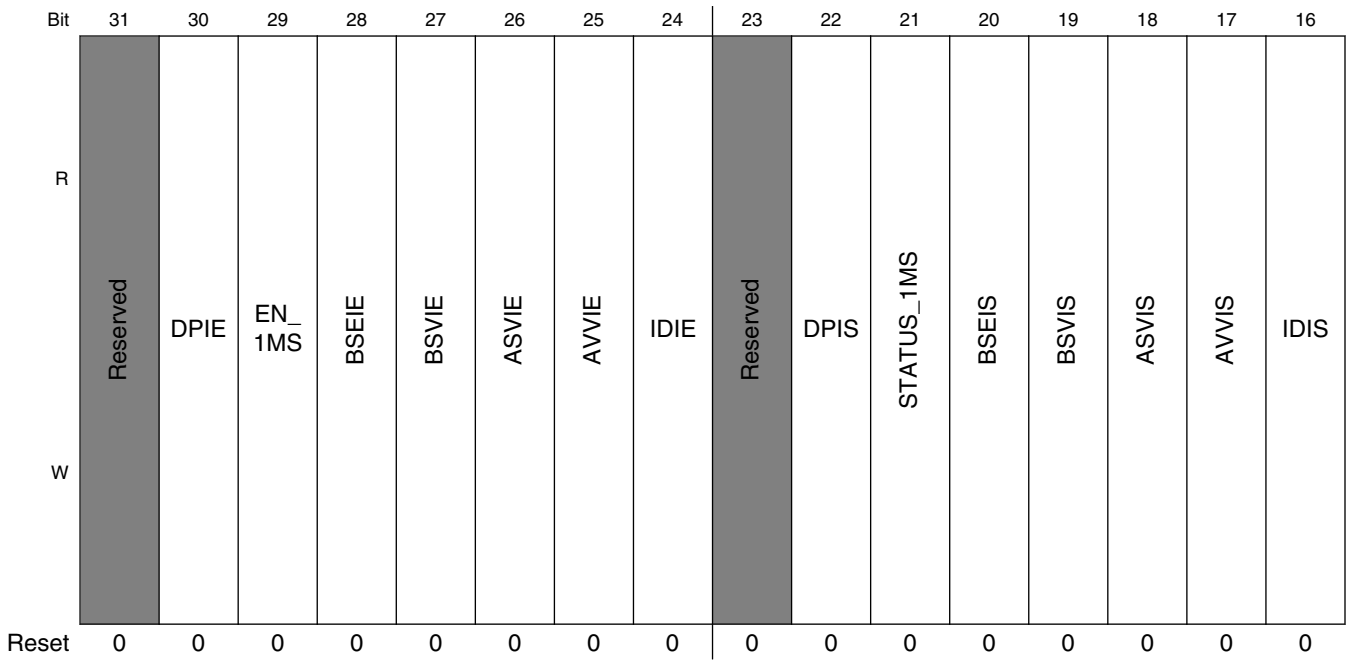
This register is availabe only in OTG controller core. It has four sections:

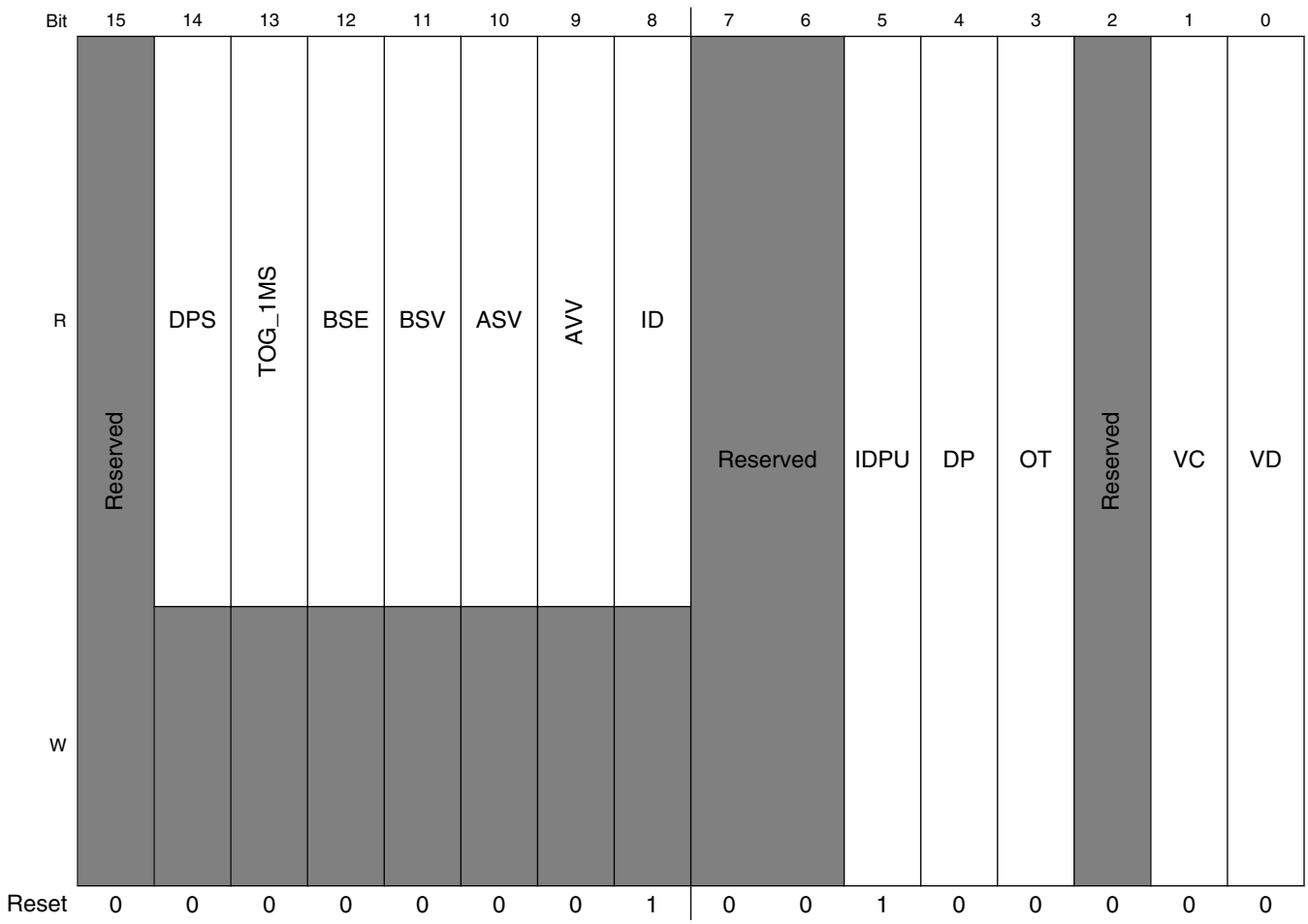
- OTG Interrupt enables (Read/Write)
- OTG Interrupt status (Read/Write to Clear)
- OTG Status inputs (Read Only)
- OTG Controls (Read/Write)

The status inputs are debounced using a 1 ms time constant. Values on the status inputs that do not persist for more than 1 ms does not cause an update of the status input register, or cause an OTG interrupt.

See also USB Device Mode (USB_*n*USBMODE) register.

Address: 4033_0000h base + 1A4h offset + (65536d × i), where i=0d to 1d





USB_*n*OTGSC field descriptions

Field	Description
31 -	This field is reserved. Reserved
30 DPIE	Data Pulse Interrupt Enable
29 EN_1MS	1 millisecond timer Interrupt Enable - Read/Write
28 BSEIE	B Session End Interrupt Enable - Read/Write. Setting this bit enables the B session end interrupt.
27 BSVIE	B Session Valid Interrupt Enable - Read/Write. Setting this bit enables the B session valid interrupt.
26 ASVIE	A Session Valid Interrupt Enable - Read/Write. Setting this bit enables the A session valid interrupt.
25 AVVIE	A VBus Valid Interrupt Enable - Read/Write. Setting this bit enables the A VBus valid interrupt.
24 IDIE	USB ID Interrupt Enable - Read/Write. Setting this bit enables the USB ID interrupt.
23 -	This field is reserved. Reserved
22 DPIS	Data Pulse Interrupt Status - Read/Write to Clear. This bit is set when data bus pulsing occurs on DP or DM. Data bus pulsing is only detected when USBMODE.CM = Host (11) and PORTSC1(0)[PP] = 0. Software must write a one to clear this bit.
21 STATUS_1MS	1 millisecond timer Interrupt Status - Read/Write to Clear. This bit is set once every millisecond. Software must write a one to clear this bit.
20 BSEIS	B Session End Interrupt Status - Read/Write to Clear. This bit is set when VBus has fallen below the B session end threshold. Software must write a one to clear this bit.
19 BSVIS	B Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the B session valid threshold. Software must write a one to clear this bit.
18 ASVIS	A Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the A session valid threshold. Software must write a one to clear this bit.
17 AVVIS	A VBus Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the VBus valid threshold on an A device. Software must write a one to clear this bit.
16 IDIS	USB ID Interrupt Status - Read/Write. This bit is set when a change on the ID input has been detected. Software must write a one to clear this bit.

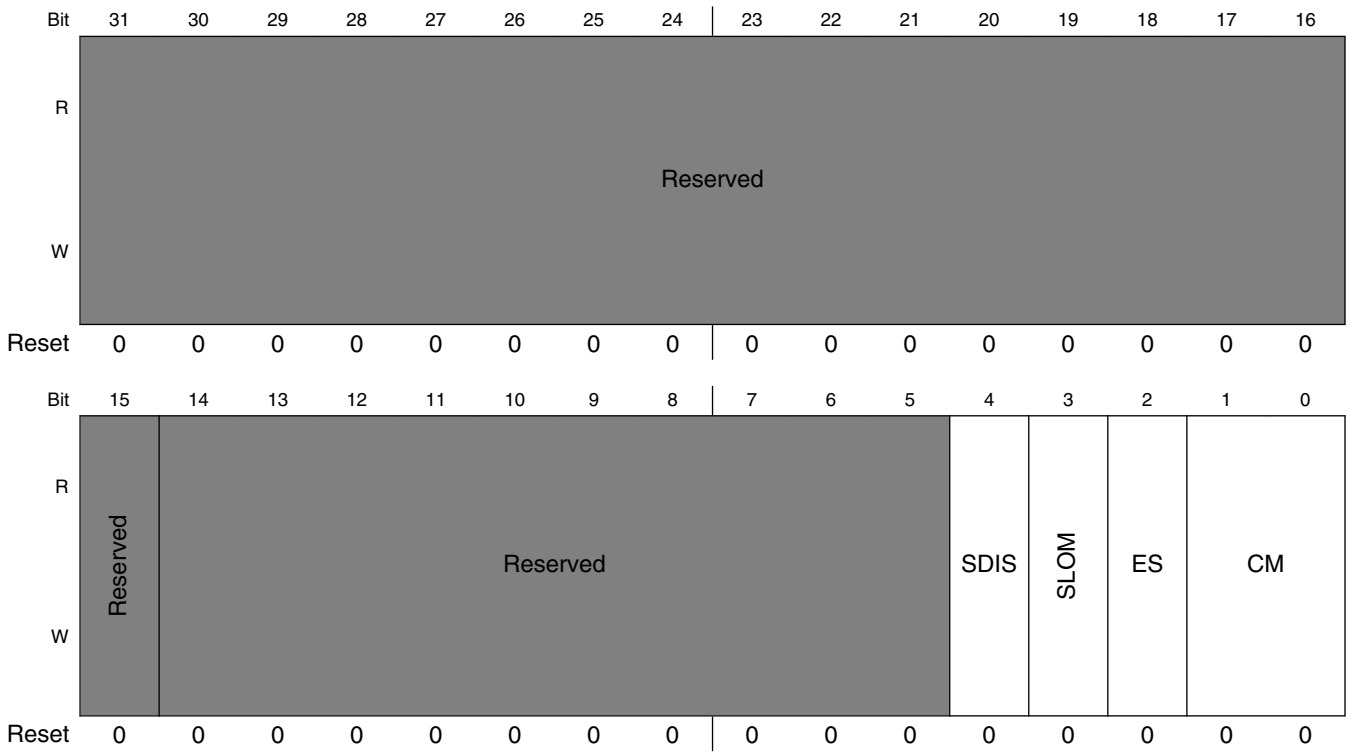
Table continues on the next page...

USB_*n*OTGSC field descriptions (continued)

Field	Description
15 -	This field is reserved. Reserved
14 DPS	Data Bus Pulsing Status - Read Only. A '1' indicates data bus pulsing is being detected on the port.
13 TOG_1MS	1 millisecond timer toggle - Read Only. This bit toggles once per millisecond.
12 BSE	B Session End - Read Only. Indicates VBus is below the B session end threshold.
11 BSV	B Session Valid - Read Only. Indicates VBus is above the B session valid threshold.
10 ASV	A Session Valid - Read Only. Indicates VBus is above the A session valid threshold.
9 AVV	A VBus Valid - Read Only. Indicates VBus is above the A VBus valid threshold.
8 ID	USB ID - Read Only. 0 = A device, 1 = B device
7–6 -	This field is reserved. Reserved
5 IDPU	ID Pullup - Read/Write This bit provide control over the ID pull-up resistor; 0 = off, 1 = on [default]. When this bit is 0, the ID input will not be sampled.
4 DP	Data Pulsing - Read/Write. Setting this bit causes the pullup on DP to be asserted for data pulsing during SRP.
3 OT	OTG Termination - Read/Write. This bit must be set when the OTG device is in device mode, this controls the pulldown on DM.
2 -	This field is reserved. Reserved
1 VC	VBUS Charge - Read/Write. Setting this bit causes the VBus line to be charged. This is used for VBus pulsing during SRP.
0 VD	VBUS_Discharge - Read/Write. Setting this bit causes VBus to discharge through a resistor.

54.7.33 USB Device Mode (USB_*n*USBMODE)

Address: 4033_0000h base + 1A8h offset + (65536d × i), where i=0d to 1d



USB_*n*USBMODE field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14–5 -	This field is reserved. Reserved
4 SDIS	Stream Disable Mode. (0 - Inactive [default]; 1 - Active) Device Mode: Setting to a '1' disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double buffering scheme is disabled to prevent overruns/underruns in bandwidth limited systems. Note: In High Speed Mode, all packets received are responded to with a NYET handshake when stream disable is active. Host Mode: Setting to a '1' ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the TX latency is filled to capacity before the packet is launched onto the USB. **NOTE:** Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TX FIFO Fill Tuning (USB_*n*TXFILLTUNING) and TXTTFILLTUNING [MPH Only] to characterize the adjustments needed for the scheduler when using this feature.

Table continues on the next page...

USB_*n*USBMODE field descriptions (continued)

Field	Description
	NOTE: The use of this feature substantially limits of the overall USB performance that can be achieved.
3 SLOM	Setup Lockout Mode. In device mode, this bit controls behavior of the setup lock mechanism. See Control Endpoint Operation Model . 0 Setup Lockouts On (default); 1 Setup Lockouts Off (DCD requires use of Setup Data Buffer Tripwire in USB Command Register (USB_*n*USBCMD) .
2 ES	Endian Select - Read/Write. This bit can change the byte alignment of the transfer buffers to match the host microprocessor. The bit fields in the microprocessor interface and the data structures are unaffected by the value of this bit because they are based upon the 32-bit word. Bit Meaning 0 Little Endian [Default] 1 Big Endian
CM	Controller Mode - R/WO. Controller mode is defaulted to the proper mode for host only and device only implementations. For those designs that contain both host & device capability, the controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. For combination host/device controllers, this register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the *RESET* bit in the USBCMD register before reprogramming this register. For OTG controller core, reset value is '00b'. For Host-only controller core, reset value is '11b'. 00 Idle [Default for combination host/device] 01 Reserved 10 Device Controller [Default for device only controller] 11 Host Controller [Default for host only controller]

54.7.34 Endpoint Setup Status (USB_*n*ENDPTSETUPSTAT)

Address: 4033_0000h base + 1ACh offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved																ENDPTSETUPSTAT																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTSETUPSTAT field descriptions

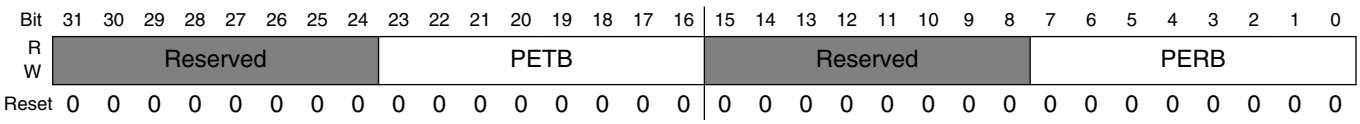
Field	Description
31–16 -	This field is reserved. Reserved
ENDPTSETUPSTAT	Setup Endpoint Status. For every setup transaction that is received, a corresponding bit in this register is set to one. Software must clear or acknowledge the setup transfer by writing a one to a respective bit after it has read the setup data from Queue head. The response to a setup packet as in the order of operations and total response time is crucial to limit bus time outs while the setup lock our mechanism is engaged. See Managing Endpoints in the Device Operational Model. This register is only used in device mode.

54.7.35 Endpoint Prime (USB_*n*ENDPTPRIME)

This register is only used in device mode.

When software sets the prime bit for a given endpoint, the device controller loads the transfer descriptor, pointed to by the queue head, such that the endpoint is ready to transmit or receive when the host sends a request (IN/OUT token). The endpoint will NAK all requests from the host until the endpoint is primed. The controller will automatically re-prime the endpoint with a new transfer descriptor when one is found via the next_dtd pointer of the current transfer descriptor. Hence, the prime bit must only be set by software when a descriptor is added to the queue head.

Address: 4033_0000h base + 1B0h offset + (65536d × i), where i=0d to 1d



USB_*n*ENDPTPRIME field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 PETB	Prime Endpoint Transmit Buffer - R/WS. For each endpoint a corresponding bit is used to request that a buffer is prepared for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. Software should write a one to the corresponding bit when posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a transmit buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed. **NOTE:** These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PETB[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
PERB	Prime Endpoint Receive Buffer - R/WS. For each endpoint, a corresponding bit is used to request a buffer prepare for a receive operation for when a USB host initiates a USB OUT transaction. Software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed. **NOTE:** These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PERB[N] - Endpoint #N, N is in 0..7

54.7.36 Endpoint Flush (USB_*n*ENDPTFLUSH)

This register is only used in device mode.

Address: 4033_0000h base + 1B4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								FETB								Reserved								FERB							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTFLUSH field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 FETB	Flush Endpoint Transmit Buffer - R/WS. Writing one to a bit(s) in this register causes the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful. FETB[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
FERB	Flush Endpoint Receive Buffer - R/WS. Writing one to a bit(s) causes the assocUOGiated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful. FERB[N] - Endpoint #N, N is in 0..7\

54.7.37 Endpoint Status (USB_*n*ENDPTSTAT)

This register is only used in device mode.

Address: 4033_0000h base + 1B8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								ETBR								Reserved								ERBR							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTSTAT field descriptions

Field	Description
31–24 -	This field is reserved. Reserved

Table continues on the next page...

USB_*n*ENDPTSTAT field descriptions (continued)

Field	Description
23–16 ETBR	Endpoint Transmit Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to one by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There is always a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. **NOTE:** These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ETBR[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
ERBR	Endpoint Receive Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to a one by the hardware as a response to receiving a command from a corresponding bit in the ENDPRIME register. There is always a delay between setting a bit in the ENDPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. **NOTE:** These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ERBR[N] - Endpoint #N, N is in 0..7

54.7.38 Endpoint Complete (USB_*n*ENDPTCOMPLETE)

This register is only used in device mode.

Address: 4033_0000h base + 1BCh offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								ETCE								Reserved								ERCE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTCOMPLETE field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 ETCE	Endpoint Transmit Complete Event - R/WC. Each bit indicates a transmit event (IN/INTERRUPT) occurred and software should read the corresponding endpoint queue to determine the endpoint status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the *USBINT* . Writing one clears the corresponding bit in this register. ETCE[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved

Table continues on the next page...

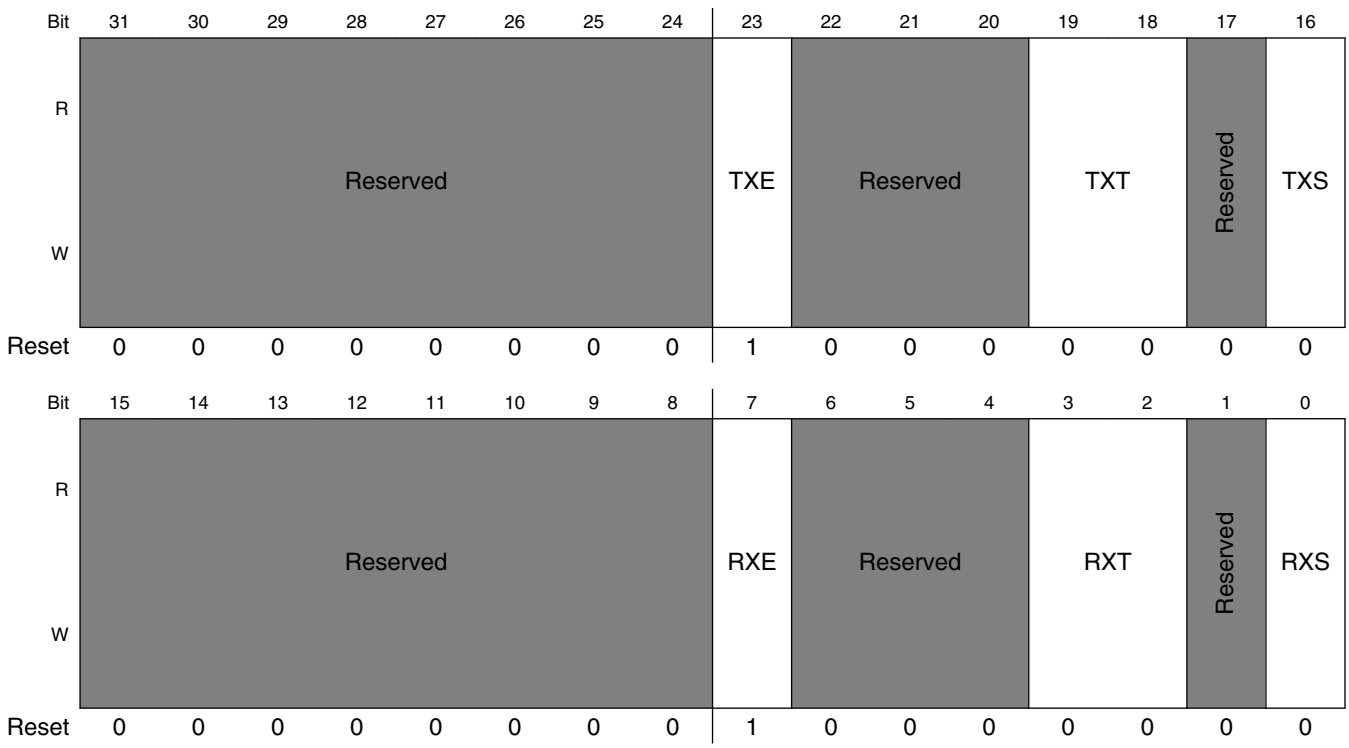
USB_*n*ENDPTCOMPLETE field descriptions (continued)

Field	Description
ERCE	Endpoint Receive Complete Event - RW/C. Each bit indicates a received event (OUT/SETUP) occurred and software should read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the *USBINT* . Writing one clears the corresponding bit in this register. ERCE[N] - Endpoint #N, N is in 0..7

54.7.39 Endpoint Control0 (USB_*n*ENDPTCTRL0)

Every Device implements Endpoint 0 as a control endpoint.

Address: 4033_0000h base + 1C0h offset + (65536d × i), where i=0d to 1d



USB_*n*ENDPTCTRL0 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 1 Enabled Endpoint0 is always enabled.

Table continues on the next page...

USB_*n*ENDPTCTRL0 field descriptions (continued)

Field	Description
22–20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 - Control Endpoint0 is fixed as a Control End Point.
17 -	This field is reserved. Reserved
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK [Default] 1 End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. **NOTE:** There is a slight delay (50 clocks max.) between the endptsetupstat being cleared and hardware continuing to clear this bit. In most systems it is unlikely the DCD software will observe this delay. However, should the dcd observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a newsetup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 1 Enabled Endpoint0 is always enabled.
6–4 -	This field is reserved. Reserved
3–2 RXT	RX Endpoint Type - Read/Write 00 Control Endpoint0 is fixed as a Control End Point.
1 -	This field is reserved. Reserved
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.

Table continues on the next page...

USB_*n*ENDPTCTRL0 field descriptions (continued)

Field	Description
	NOTE: There is a slight delay (50 clocks max.) between the endptsetupstat being cleared and hardware continuing to clear this bit. In most systems it is unlikely the dcd software will observe this delay. However, should the dcd observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a newsetup has been received by checking the associated endptsetupstat bit.

54.7.40 Endpoint Control 1 (USB_*n*ENDPTCTRL1)

This is endpoint control register for endpoint 1 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 4033_0000h base + 1C4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTCTRL1 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved

Table continues on the next page...

USB_*n*ENDPTCTRL1 field descriptions (continued)

Field	Description
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

54.7.41 Endpoint Control 2 (USB_*n*ENDPTCTRL2)

This is endpoint control register for endpoint 2 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 4033_0000h base + 1C8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTCTRL2 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.

Table continues on the next page...

USB_*n*ENDPTCTRL2 field descriptions (continued)

Field	Description
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence

Table continues on the next page...

USB_*n*ENDPTCTRL2 field descriptions (continued)

Field	Description
	Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

54.7.42 Endpoint Control 3 (USB_*n*ENDPTCTRL3)

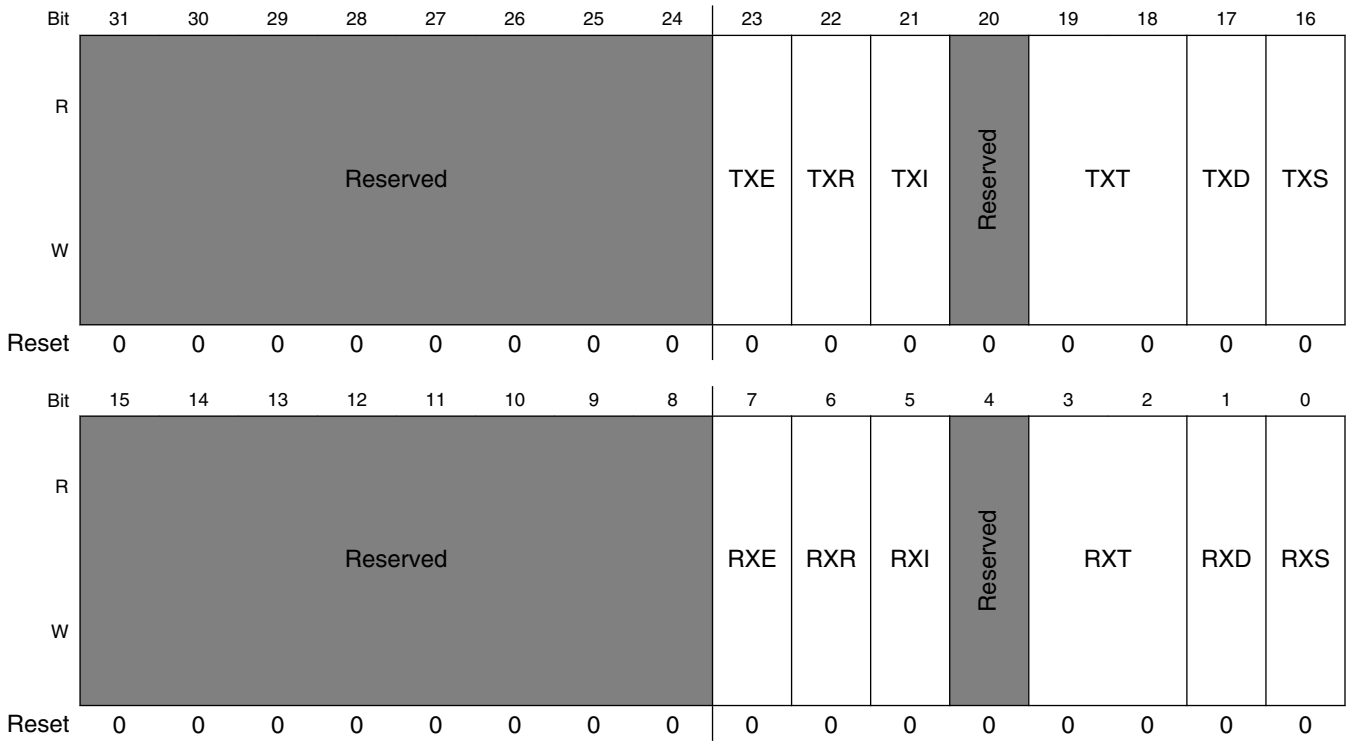
This is endpoint control register for endpoint 3 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control

causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 4033_0000h base + 1CCh offset + (65536d × i), where i=0d to 1d



USB_*n*ENDPTCTRL3 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved

Table continues on the next page...

USB_*n*ENDPTCTRL3 field descriptions (continued)

Field	Description
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control

Table continues on the next page...

USB_*n*ENDPTCTRL3 field descriptions (continued)

Field	Description
	01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

54.7.43 Endpoint Control 4 (USB_*n*ENDPTCTRL4)

This is endpoint control register for endpoint 4 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 4033_0000h base + 1D0h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USB_*n*ENDPTCTRL4 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous

Table continues on the next page...

USB_*n*ENDPTCTRL4 field descriptions (continued)

Field	Description
	10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved

Table continues on the next page...

USB_*n*ENDPTCTRL4 field descriptions (continued)

Field	Description
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

54.7.44 Endpoint Control 5 (USB_*n*ENDPTCTRL5)

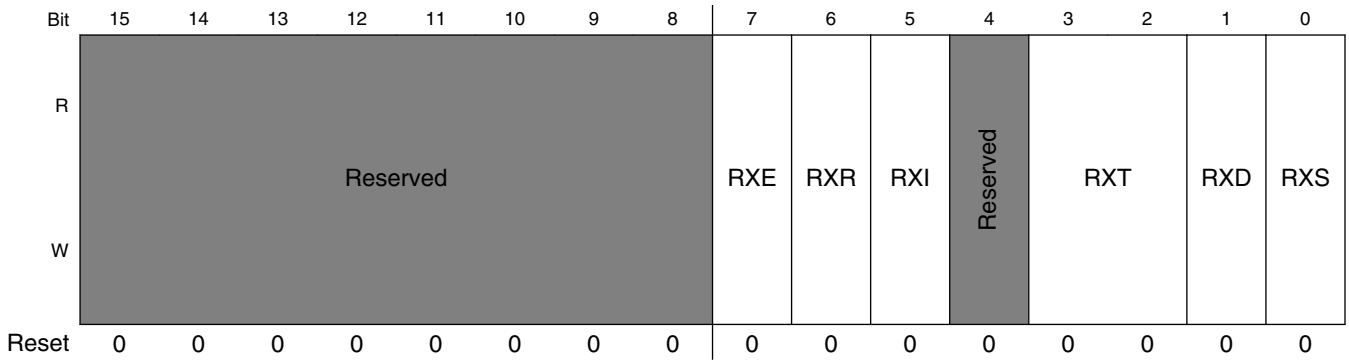
This is endpoint control register for endpoint 5 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 4033_0000h base + 1D4h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USB_*n*ENDPTCTRL5 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

USB_*n*ENDPTCTRL5 field descriptions (continued)

Field	Description
	This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled

Table continues on the next page...

USB_*n*ENDPTCTRL5 field descriptions (continued)

Field	Description
	This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

54.7.45 Endpoint Control 6 (USB_*n*ENDPTCTRL6)

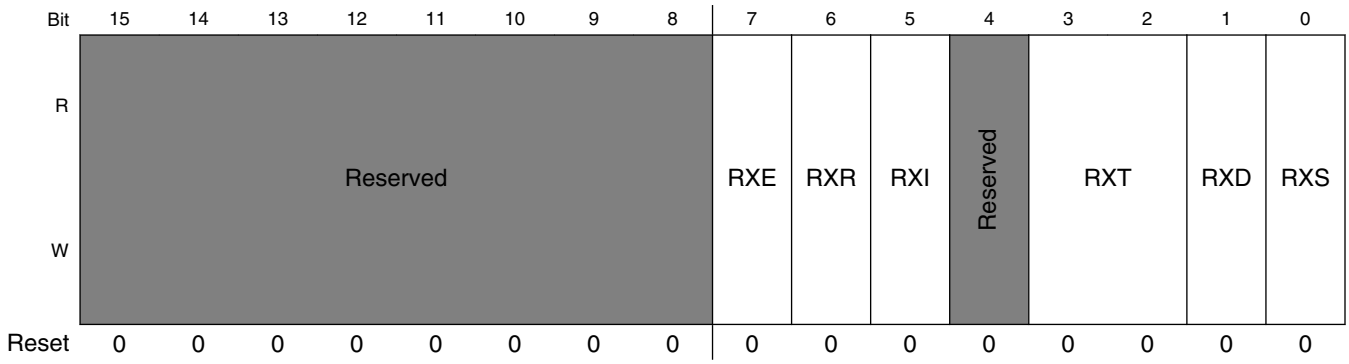
This is endpoint control register for endpoint 6 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 4033_0000h base + 1D8h offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USB_*n*ENDPTCTRL6 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

USB_*n*ENDPTCTRL6 field descriptions (continued)

Field	Description
	This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled

Table continues on the next page...

USB_*n*ENDPTCTRL6 field descriptions (continued)

Field	Description
	This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

54.7.46 Endpoint Control 7 (USB_*n*ENDPTCTRL7)

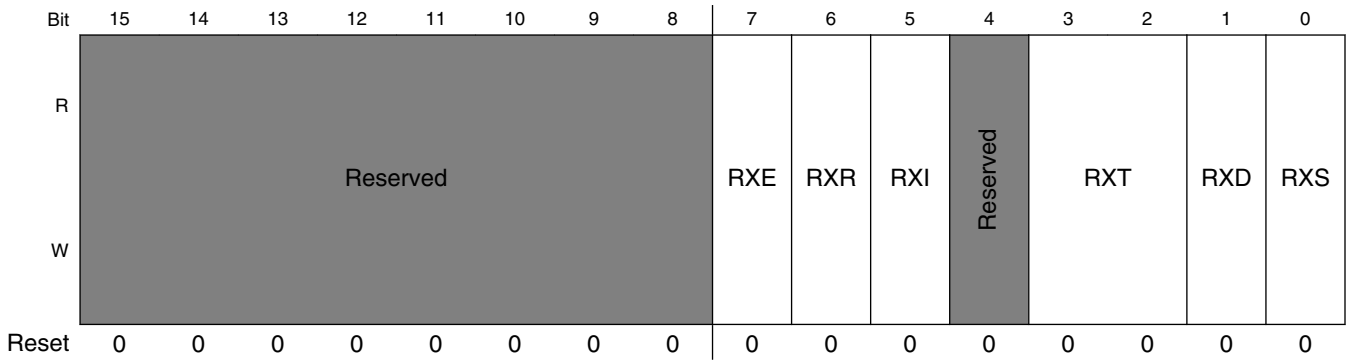
This is endpoint control register for endpoint 7 in device operation mode.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 4033_0000h base + 1DCh offset + (65536d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



USB_*n*ENDPTCTRL7 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

USB_*n*ENDPTCTRL7 field descriptions (continued)

Field	Description
	This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled

Table continues on the next page...

USB_*n*ENDPTCTRL7 field descriptions (continued)

Field	Description
	This bit is set automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpointand this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. This control will continue to STALL until this bit is either cleared by software or automatically cleared as above for control endpoints. **NOTE:** [CONTROL ENDPOINT TYPES ONLY]: there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the stall bit is not set after writing a one to it then follow this procedure: continually write this stall bit until it is set or until a new setup has been received by checking the associated endptsetupstat bit.

Chapter 55
Universal System Bus Physical Layer (USB-PHY)

55.1 Chip-specific USB-PHY information

Table 55-1. Reference links to related information

Topic	Related module	Reference
Full description	USB-PHY	USB-PHY
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

55.1.1 USB PHY

This device integrates one USB 2.0 PHY macrocells capable of connecting to USB host/device systems at low-speed, full-speed, and high-speed. The USB PHY provides a standard UTMI interface for connection to the USBHS controller. It has the integrated PLL and DCD block, with an IP bus interface for configuration. The USB_DP and USB_DN pins connect directly to the USB connector. The following table shows the configuration of the USB PHY.

Table 55-2. USB PHY configuration

Parameter	Description
Name	USB PHY
Supported standard version	USB2.0 Specification
Instances	1
Configurable features	NA
Interface speed	480 Mbps
External I/O pins	See the IOMUXC spreadsheet attached with this RM.

55.2 USB PHY Overview

The chip contains one integrated USB 2.0 PHY macrocell capable of connecting to USB host/device systems at the USB low-speed (LS) rate of 1.5 Mbits/s, the full-speed (FS) rate of 12 Mbits/s, or the USB 2.0 high-speed (HS) rate of 480 Mbits/s.

See Figure 55-1 for a block diagram of the PHY. The integrated PHY provides a standard UTMI+ interface. It has an integrated 480MHz PLL and DCD (Device Charger Detector) analog circuits, with an IP bus interface for configurations. For this product the Device Charger Detector functions are detailed in the separate USB Device Charger Detection Module (USBDCD) chapter in this Reference Manual. The USB_DP and USB_DM pins connect directly to a USB connector.

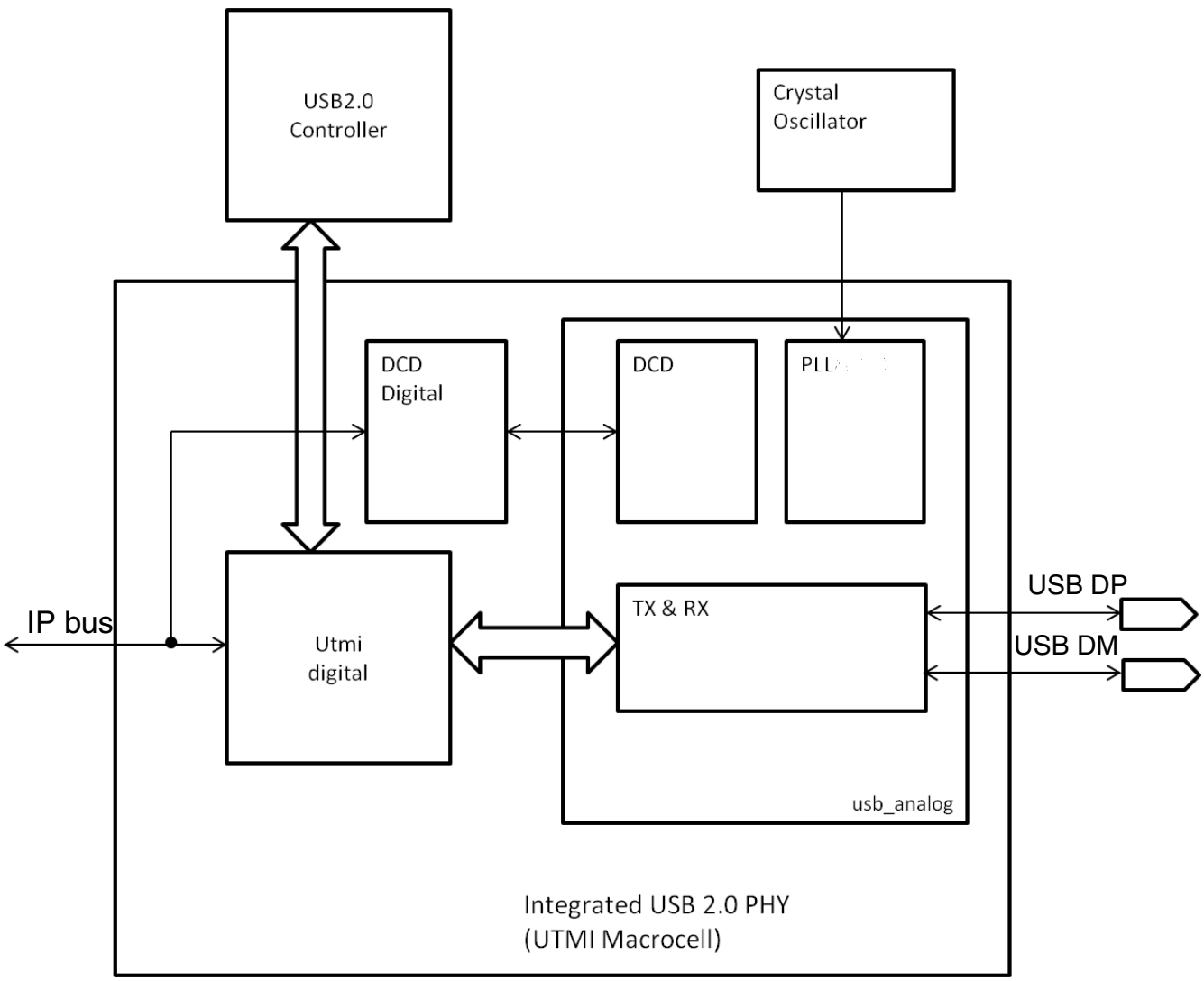


Figure 55-1. USB 2.0 PHY Block Diagram

The following subsections describe the external interfaces, internal interfaces, major blocks, and programmable registers that comprise the integrated USB 2.0 High-Speed PHY.

55.3 Operation

The UTM provides a 16-bit interface to the USB controller. This interface is clocked at 30 MHz.

- The digital portions of the USBPHY block include the UTMI, digital transmitter, digital receiver, and the programmable registers.
- The analog transceiver section comprises an analog receiver and an analog transmitter, as shown in Figure 55-2.

55.3.1 UTMI

The UTMI block handles the line_state bits, reset buffering, suspend distribution, transceiver speed selection, and transceiver termination selection.

The PLL in the USB PHY supplies a 480MHz clock to UTMI digital. The UTMI block will divide the clock to generate the 30MHz clock used in the interface.

55.3.2 Initialization and application information

In order to bring up the internal 480MHz USB PLL clock, four external conditions should be met as shown below.

- The reference clock for USB PLL is the SOSC. So, one needs to enable it via SCG0_SOSCCSR[SOSCEN].
- Enable some settings via the USBPHY_PLL Control/Status Register
  - USBPHY_PLL_SIC[PLL_BYPASS]: Clear this bit to clear bypass feature
  - USBPHY_PLL_SIC[PLL_REG_ENABLE]: Enable USB PLL regulator
  - Delay more than 15 us.
  - USBPHY_PLL_SIC[PLL_BYPASS]: Clear this bit to clear bypass
  - USBPHY_PLL_SIC[PLL_POWER]: Power-up the USB PLL

- USBPHY_PLL_SIC[PLL_EN_USB_CLKS]: Enable the USB clock
- USBPHY_PLL_SIC[PLL_ENABLE]: Enable the clock output from the USB PLL

- The system oscillator ranges from 16 MHz to 32 MHz. USBPHY_PLL_SIC[PLL_DIV_SEL[2:0] should be programmed properly as per the external reference selected.

55.3.3 Digital Transmitter

The digital transmitter receives the 16-bit transmit data from the USB controller and handles the tx_valid, tx_validh and tx_ready handshake.

In addition, it contains the transmit serializer that converts the 16-bit parallel words at 30 MHz to a single bitstream at 480 Mbit for high-speed or 12 Mbit for full-speed or 1.5 Mbit for low-speed. It does this while implementing the bit-stuffing algorithm and the NRZI encoder that are used to remove the DC component from the serial bitstream. The output of this encoder is sent to the low-speed (LS), full-speed (FS) or high-speed (HS) drivers in the analog transceiver section's transmitter block.

55.3.4 Digital Receiver

The digital receiver receives the raw serial bitstream from the low speed (LS) differential transceiver, full speed (FS) differential transceiver, and 480 MHz, 9X oversampled data from the high speed (HS) differential transceiver.

As the phase of the USB host transmitter shifts relative to the local PLL, the receiver section's HS DLL tracks these changes to give a reliable sample of the incoming 480 Mbit/s bitstream. Since this sample point shifts relative to the PLL phase used by the digital logic, a rate-matching elastic buffer is provided to cross this clock domain boundary. Once the bitstream is in the local clock domain, an NRZI decoder and bit unstuffer restore the original payload data bitstream and pass it to a deserializer and holding register. The receive state machine handles the rx_valid, rx_validh, and handshake with the USB controller. The handshake is not interlocked, in that there is no rx_ready signal coming from the controller. The controller must take each 16-bit value as presented by the PHY. The receive state machine provides an rx_active signal to the controller that indicates when it is inside a valid packet (SYNC detected, and so on).

55.3.5 Analog Receiver

The analog receiver comprises five differential receivers, two single-ended receivers, and a 9X, 480 MHz HS data sampling module, as shown in the figure below and described further in this section.

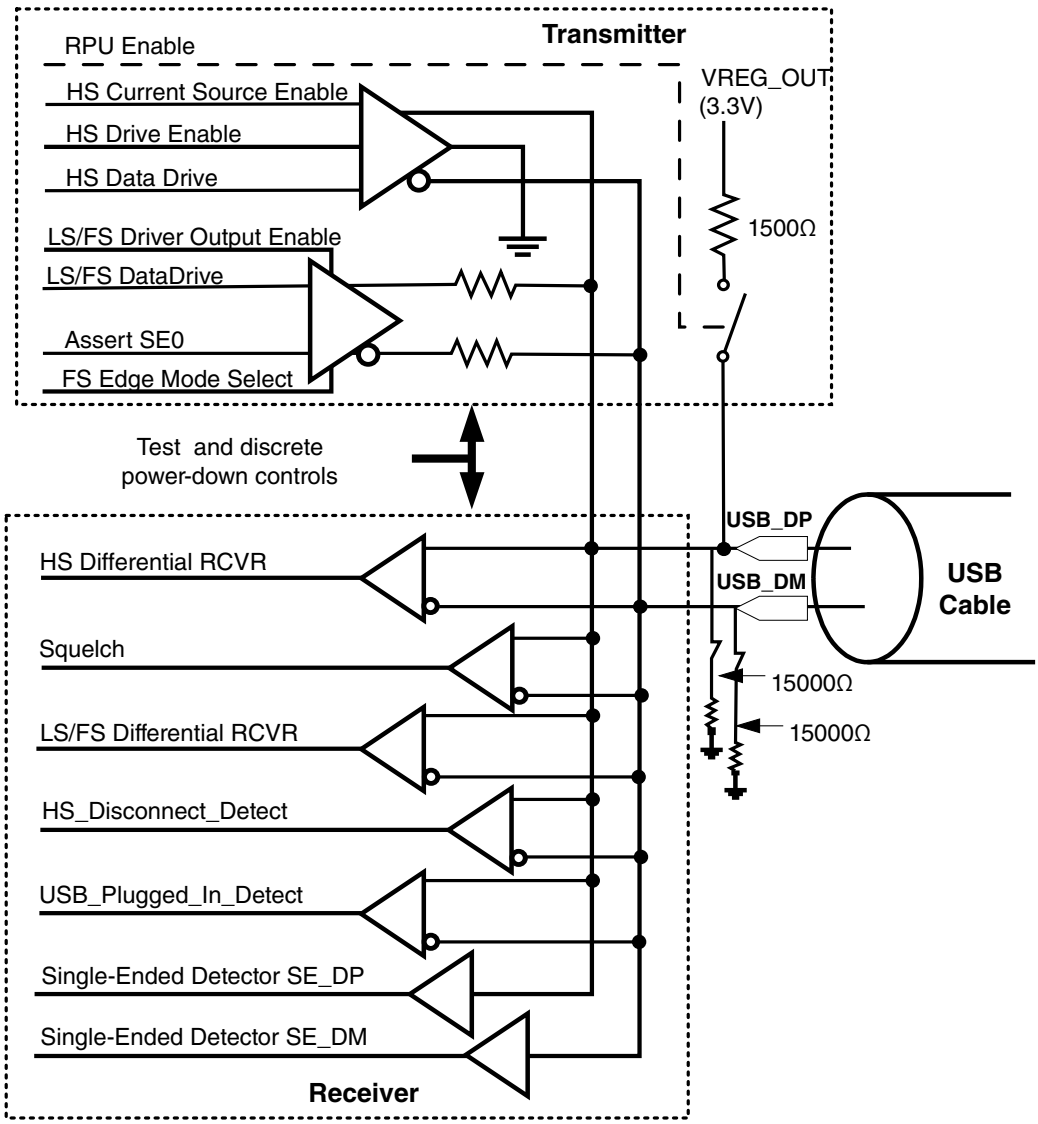


Figure 55-2. USB 2.0 PHY Analog Transceiver Block Diagram

55.3.5.1 HS Differential Receiver

The high-speed differential receiver is both a differential analog receiver and threshold comparator. Its output is high (1'b1) if the differential signal is greater than a 0-V threshold.

Otherwise, its output is low (1'b0). Its purpose is to discriminate the ± 400-mV differential voltage resulting from the high-speed drivers current flow into the dual 45Ω terminations found on each pin of the differential pair. The envelope or squelch detector, described below, ensures that the differential signal has sufficient magnitude to be valid. The HS differential receiver tolerates up to 500 mV of common mode offset.

55.3.5.2 Squelch Detector

The squelch detector is a differential analog receiver and threshold comparator.

Its output is high (1'b1) if the differential magnitude is less than a nominal 100 mV threshold. Otherwise, its output is low (1'b0).

Its purpose is to invalidate the HS differential receiver when the incoming signal is simply too low to receive reliably.

55.3.5.3 LS/FS Differential Receiver

The low-speed/full-speed differential receiver is both a differential analog receiver and threshold comparator.

The crossover voltage falls between 1.3 V and 2.0 V. Its output is 1, when the USB_DP line is above the crossover point and the USB_DM line is below the crossover point. The digital receiver section decodes the receiver data into J or K state according to the speed.

55.3.5.4 HS Disconnect Detector

It is a differential analog receiver and threshold comparator. It outputs high when differential magnitude is greater than a nominal 575-mV threshold. Otherwise, it outputs low.

55.3.5.5 USB Plugged-In (Cable Attach) Detection

The USB HS PHY when operating in device mode provides three different methods for the local USB device to detect the attachment of a cable also attached to a remote USB host or hub. Only one of these methods should be enabled at a time when waiting to identify the cable attachment event, and all of them must be disabled during USB data communication.

The preferred standards-based method is to use the Data Contact Detect function per the *USB Battery Charging Specification, Revision 1.2*. The use of this method is described in the USB Device Charger Detection Module (USBDCD) chapter in this Reference Manual.

The second method, called the USB plugged-in detector, looks for both USB_DP and USB_DM to be high. There is a pair of large on-chip switchable pullup resistors (200 KΩ) that hold both USB_DP and USB_DM high when the USB cable is not attached. The USB plugged-in detector signals a 0 in this case. When operating in device mode, the upstream port in the remote host/hub interface contains a 15 KΩ pulldown resistor which could easily override the 200 KΩ pullup resistor. When the cable attachment event occurs, at least one signal in the pair will be low, which will force the plugged-in detector output high. The USB plugged-in detector is controlled through the USBPHY_CTRL[ENVPLUGINDET] bit field and results are observable through the USBPHY_STATUS[DEVPLUGIN_STATUS] bit field.

The final method is a legacy resistive detection function described in the now-obsolete *USB Battery Charging Specification, Revision 1.0*. This method enables a 125 KΩ pullup resistor on the USB_DP pin and a 375 KΩ pulldown resistor on the USB_DM pin. When no cable is plugged in, the USB_DP pin will be high and the USB_DM pin will be low. If the local device has a cable attached to a remote host or hub, due to the host 15 KΩ pulldowns both the USB_DP and USB_DM pins will be low. If the local device has a cable attached to a dedicated charger with no pullup/pulldown resistors, both the USB_DP and USB_DM pins will be high. This method is enabled through the USBPHY_USB1_VBUS_DETECT[EN_CHARGER_RESISTOR] bit field. Results can be observed using the single ended receiver status in the USBPHY_CHRG_DET_STAT[DP_STATE, DM_STATE] bit fields.

55.3.5.6 Single-Ended USB_DP Receiver

The single-ended USB_DP receiver output is high whenever the USB_DP input is above its nominal 1.8 V threshold.

55.3.5.7 Single-Ended USB_DM Receiver

The single-ended USB_DM receiver output is high whenever the USB_DM input is above its nominal 1.8 V threshold.

55.3.5.8 9X Oversample Module

The 9X oversample module uses nine identically spaced phases of the 480 MHz clock to sample a high speed bit data. The squelch signal is sampled only 1X.

55.3.6 Analog Transmitter

The analog transmitter comprises two differential drivers: one for high-speed signaling and one for full-speed signaling. It also contains the switchable 1.5 KΩ pullup resistor.

See Figure 55-3.

55.3.6.1 Switchable High-Speed 45Ω Termination Resistors

High-speed current mode differential signaling requires good 90 Ω differential termination at each end of the USB cable. This results from switching in 45 Ω terminating resistors from each signal line to ground at each end of the cable.

Because each signal is parallel terminated with 45 Ω at each end, each driver sees a 22.5 Ω load. This load impedance is much too low for full-speed signaling levels—hence the need for switchable high-speed terminating resistors. Switchable trimming resistors are provided to tune the actual termination resistance of each device, as shown in Figure 55-3. The HW_USBPHY_TX_TXCAL45DP bit field, for example, allows one of 16 trimming resistor values to be placed in parallel with the 45Ω terminator on the USB_DP signal.

55.3.6.2 Low-Speed/Full-Speed Differential Driver

The low-speed/full-speed differential drivers are essentially a pair of low-impedance pullup/pulldown devices that are switched in a differential mode for most low-speed or full-speed signaling. One output is driven high while the other output is driven low to signal the "J" state or the "K" state. Both outputs are driven low for the low-speed/full-speed "SE0" state.

55.3.6.3 High-Speed Differential Driver

The high-speed differential driver receives a 17.78 mA current from the constant current source (Iref) and essentially steers it down either the USB_DP signal or the USB_DM signal or alternatively to ground.

This current will produce approximately a 400 mV drop across the 22.5 Ω termination seen by the driver when it is steered onto one of the signal lines. The approximately 17.78 mA current source is referenced back to the integrated voltage-band-gap (Vbg) circuit.

55.3.6.4 Switchable 1.5KΩ USB_DP Pullup Resistor

This product contains a switchable 1.5 KΩ pullup resistor on the USB_DP signal.

This resistor is switched on to indicate to the host/hub controller that a full-speed-capable device is on the USB cable, powered on, and ready. This resistor is switched off at power-on reset so the host does not recognize a USB device until the processor software enables the announcement of a full-speed device.

55.3.6.5 Switchable 15KΩ USB_DP and USB_DM Pulldown Resistors

This product contains switchable 15 KΩ pulldown resistors on both USB_DP and USB_DM signals. They are enabled in host mode to indicate to the device controller on the far end of the USB connection that a host is present. These resistors are also used during certain Battery Charging detector functions.

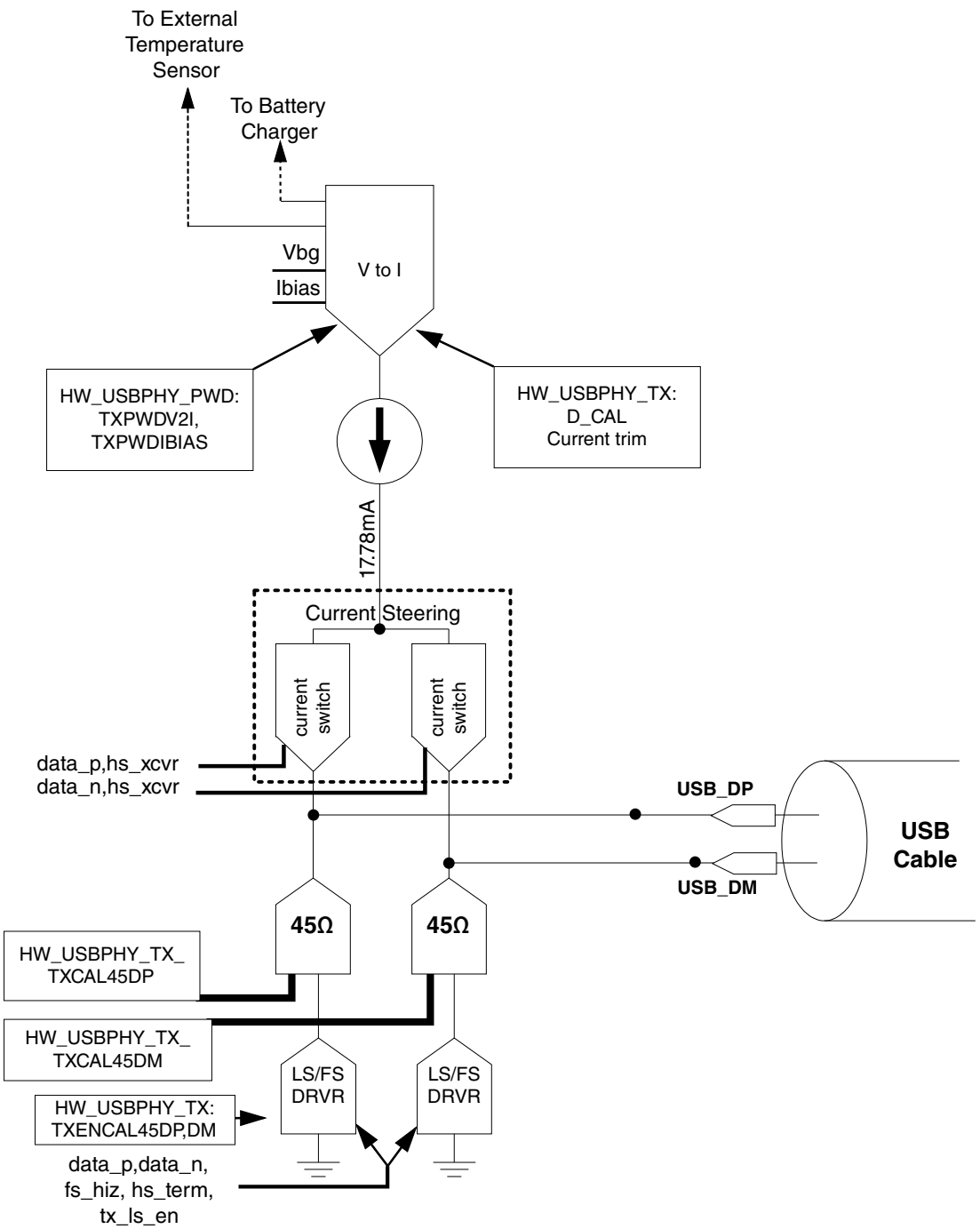


Figure 55-3. USB 2.0 PHY Transmitter Block Diagram

55.3.7 Recommended Register Configuration for USB Certification

The USB HS PHY has programmability to adjust several transceiver parameters.

- The TX HS driver termination impedance/FS driver output impedance and HS driver output currents can be adjusted through bit fields of the USBPHY_TX register. The ability to override the default parametric trim bit fields using the USBPHY_TX register has to be enabled by setting bit fields in the USBPHY_TRIM_OVERRIDE register.
- The HS RX thresholds for the Envelope Detector/Squelch comparator and the Host Disconnect comparator can be adjusted through bit fields of the USBPHY_RX register.

The default values of the bit fields for the transceiver parametric trims are centered with no changes needed for USB Certification testing when the product is used with boards optimized for signal integrity on the HS USB port.

In other cases, such as when external components are inserted between the USB DP/DM pins and the USB connector or other compromises are made on USB DP/DM signal routing, changes to the parametric trim bit fields may be useful.

55.4 USB PHY Memory Map/Register Definition

USBPHY Hardware Register Format Summary

Information for using the Set, Clear, Toggle register features is located in the **Register Macro Usage** section later in this chapter.

USBPHY memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4035_0000	USB PHY Power-Down Register (USBPHY_PWD)	32	R/W	001E_1C00h	55.4.1/2557
4035_0004	USB PHY Power-Down Register (USBPHY_PWD_SET)	32	R/W	001E_1C00h	55.4.1/2557
4035_0008	USB PHY Power-Down Register (USBPHY_PWD_CLR)	32	R/W	001E_1C00h	55.4.1/2557
4035_000C	USB PHY Power-Down Register (USBPHY_PWD_TOG)	32	R/W	001E_1C00h	55.4.1/2557
4035_0010	USB PHY Transmitter Control Register (USBPHY_TX)	32	R/W	See section	55.4.2/2559
4035_0014	USB PHY Transmitter Control Register (USBPHY_TX_SET)	32	R/W	See section	55.4.2/2559

Table continues on the next page...

USBPHY memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4035_0018	USB PHY Transmitter Control Register (USBPHY_TX_CLR)	32	R/W	See section	55.4.2/2559
4035_001C	USB PHY Transmitter Control Register (USBPHY_TX_TOG)	32	R/W	See section	55.4.2/2559
4035_0020	USB PHY Receiver Control Register (USBPHY_RX)	32	R/W	0000_0000h	55.4.3/2562
4035_0024	USB PHY Receiver Control Register (USBPHY_RX_SET)	32	R/W	0000_0000h	55.4.3/2562
4035_0028	USB PHY Receiver Control Register (USBPHY_RX_CLR)	32	R/W	0000_0000h	55.4.3/2562
4035_002C	USB PHY Receiver Control Register (USBPHY_RX_TOG)	32	R/W	0000_0000h	55.4.3/2562
4035_0030	USB PHY General Control Register (USBPHY_CTRL)	32	R/W	C000_0000h	55.4.4/2564
4035_0034	USB PHY General Control Register (USBPHY_CTRL_SET)	32	R/W	C000_0000h	55.4.4/2564
4035_0038	USB PHY General Control Register (USBPHY_CTRL_CLR)	32	R/W	C000_0000h	55.4.4/2564
4035_003C	USB PHY General Control Register (USBPHY_CTRL_TOG)	32	R/W	C000_0000h	55.4.4/2564
4035_0040	USB PHY Status Register (USBPHY_STATUS)	32	R/W	0000_0000h	55.4.5/2568
4035_0050	USB PHY Debug Register 0 (USBPHY_DEBUG0)	32	R/W	7F18_0000h	55.4.6/2570
4035_0054	USB PHY Debug Register 0 (USBPHY_DEBUG0_SET)	32	R/W	7F18_0000h	55.4.6/2570
4035_0058	USB PHY Debug Register 0 (USBPHY_DEBUG0_CLR)	32	R/W	7F18_0000h	55.4.6/2570
4035_005C	USB PHY Debug Register 0 (USBPHY_DEBUG0_TOG)	32	R/W	7F18_0000h	55.4.6/2570
4035_0070	UTMI Debug Status Register 1 (USBPHY_DEBUG1)	32	R/W	0000_1000h	55.4.7/2572
4035_0074	UTMI Debug Status Register 1 (USBPHY_DEBUG1_SET)	32	R/W	0000_1000h	55.4.7/2572
4035_0078	UTMI Debug Status Register 1 (USBPHY_DEBUG1_CLR)	32	R/W	0000_1000h	55.4.7/2572
4035_007C	UTMI Debug Status Register 1 (USBPHY_DEBUG1_TOG)	32	R/W	0000_1000h	55.4.7/2572
4035_0080	UTMI RTL Version (USBPHY_VERSION)	32	R	See section	55.4.8/2573
4035_00A0	USB PHY PLL Control/Status Register (USBPHY_PLL_SIC)	32	R/W	00D1_2000h	55.4.9/2574
4035_00A4	USB PHY PLL Control/Status Register (USBPHY_PLL_SIC_SET)	32	R/W	00D1_2000h	55.4.9/2574
4035_00A8	USB PHY PLL Control/Status Register (USBPHY_PLL_SIC_CLR)	32	R/W	00D1_2000h	55.4.9/2574
4035_00AC	USB PHY PLL Control/Status Register (USBPHY_PLL_SIC_TOG)	32	R/W	00D1_2000h	55.4.9/2574
4035_00C0	USB PHY VBUS Detect Control Register (USBPHY_USB1_VBUS_DETECT)	32	R/W	0070_0004h	55.4.10/ 2577
4035_00C4	USB PHY VBUS Detect Control Register (USBPHY_USB1_VBUS_DETECT_SET)	32	R/W	0070_0004h	55.4.10/ 2577
4035_00C8	USB PHY VBUS Detect Control Register (USBPHY_USB1_VBUS_DETECT_CLR)	32	R/W	0070_0004h	55.4.10/ 2577
4035_00CC	USB PHY VBUS Detect Control Register (USBPHY_USB1_VBUS_DETECT_TOG)	32	R/W	0070_0004h	55.4.10/ 2577
4035_00D0	USB PHY VBUS Detector Status Register (USBPHY_USB1_VBUS_DET_STAT)	32	R	0000_0000h	55.4.11/ 2580
4035_00E0	USB PHY Charger Detect Control Register (USBPHY_USB1_CHRG_DETECT)	32	R/W	8018_0000h	55.4.12/ 2582
4035_00E4	USB PHY Charger Detect Control Register (USBPHY_USB1_CHRG_DETECT_SET)	32	R/W	8018_0000h	55.4.12/ 2582

Table continues on the next page...

USBPHY memory map (continued)

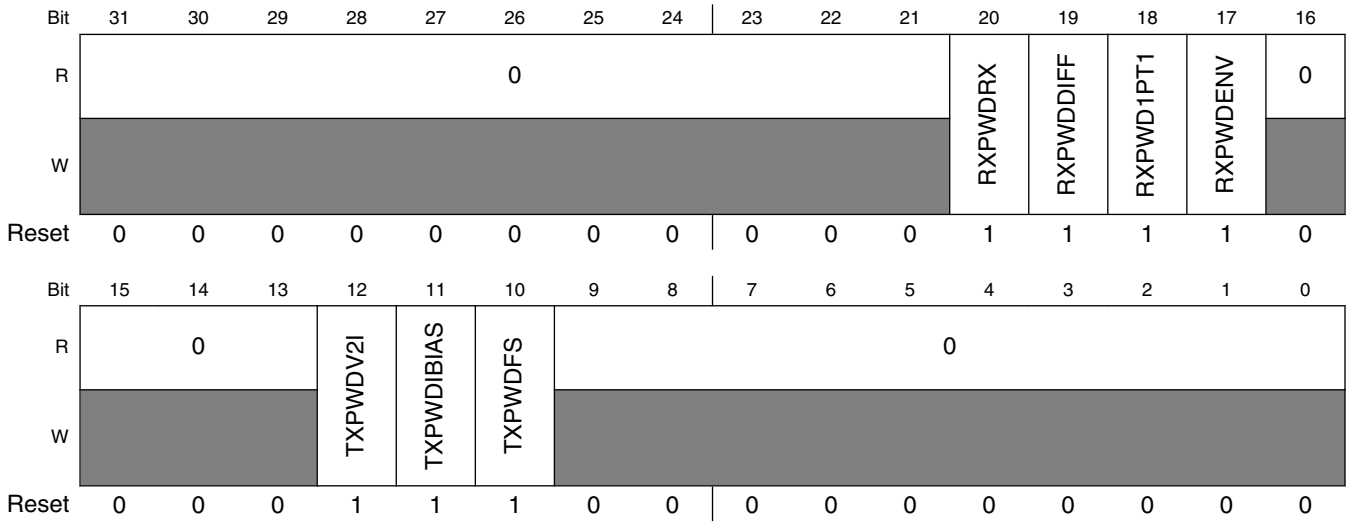
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4035_00E8	USB PHY Charger Detect Control Register (USBPHY_USB1_CHRG_DETECT_CLR)	32	R/W	8018_0000h	55.4.12/ 2582
4035_00EC	USB PHY Charger Detect Control Register (USBPHY_USB1_CHRG_DETECT_TOG)	32	R/W	8018_0000h	55.4.12/ 2582
4035_00F0	USB PHY Charger Detect Status Register (USBPHY_USB1_CHRG_DET_STAT)	32	R	0000_0000h	55.4.13/ 2583
4035_0100	USB PHY Analog Control Register (USBPHY_ANACTRL)	32	R/W	See section	55.4.14/ 2586
4035_0104	USB PHY Analog Control Register (USBPHY_ANACTRL_SET)	32	R/W	See section	55.4.14/ 2586
4035_0108	USB PHY Analog Control Register (USBPHY_ANACTRL_CLR)	32	R/W	See section	55.4.14/ 2586
4035_010C	USB PHY Analog Control Register (USBPHY_ANACTRL_TOG)	32	R/W	See section	55.4.14/ 2586
4035_0110	USB PHY Loopback Control/Status Register (USBPHY_USB1_LOOPBACK)	32	R/W	0055_0000h	55.4.15/ 2589
4035_0114	USB PHY Loopback Control/Status Register (USBPHY_USB1_LOOPBACK_SET)	32	R/W	0055_0000h	55.4.15/ 2589
4035_0118	USB PHY Loopback Control/Status Register (USBPHY_USB1_LOOPBACK_CLR)	32	R/W	0055_0000h	55.4.15/ 2589
4035_011C	USB PHY Loopback Control/Status Register (USBPHY_USB1_LOOPBACK_TOG)	32	R/W	0055_0000h	55.4.15/ 2589
4035_0120	USB PHY Loopback Packet Number Select Register (USBPHY_USB1_LOOPBACK_HSFSCNT)	32	R/W	0004_0010h	55.4.16/ 2591
4035_0124	USB PHY Loopback Packet Number Select Register (USBPHY_USB1_LOOPBACK_HSFSCNT_SET)	32	R/W	0004_0010h	55.4.16/ 2591
4035_0128	USB PHY Loopback Packet Number Select Register (USBPHY_USB1_LOOPBACK_HSFSCNT_CLR)	32	R/W	0004_0010h	55.4.16/ 2591
4035_012C	USB PHY Loopback Packet Number Select Register (USBPHY_USB1_LOOPBACK_HSFSCNT_TOG)	32	R/W	0004_0010h	55.4.16/ 2591
4035_0130	USB PHY Trim Override Enable Register (USBPHY_TRIM_OVERRIDE_EN)	32	R/W	0000_007Fh	55.4.17/ 2591
4035_0134	USB PHY Trim Override Enable Register (USBPHY_TRIM_OVERRIDE_EN_SET)	32	R/W	0000_007Fh	55.4.17/ 2591
4035_0138	USB PHY Trim Override Enable Register (USBPHY_TRIM_OVERRIDE_EN_CLR)	32	R/W	0000_007Fh	55.4.17/ 2591
4035_013C	USB PHY Trim Override Enable Register (USBPHY_TRIM_OVERRIDE_EN_TOG)	32	R/W	0000_007Fh	55.4.17/ 2591

55.4.1 USB PHY Power-Down Register (USBPHY_PWD*n*)

The USB PHY Power-Down Register provides overall control of the PHY power state.

Before programming this register, the PHY clocks must be enabled in the USBPHY_CTRL and USBPHY_PLL_SIC registers.

Address: 4035_0000h base + 0h offset + (4d × i), where i=0d to 3d



USBPHY_PWD*n* field descriptions

Field	Description
31–21 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
20 RXPWDRX	This bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHY_CTRL is enabled. 0 Normal operation 1 Power-down the entire USB PHY receiver block except for the full-speed differential receiver
19 RXPWDDIFF	Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHY_CTRL is enabled. 0 Normal operation. 1 Power-down the USB high-speed differential receiver
18 RXPWD1PT1	Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHY_CTRL is enabled. 0 Normal operation 1 Power-down the USB full-speed differential receiver.
17 RXPWDENV	Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHY_CTRL is enabled. 0 Normal operation. 1 Power-down the USB high-speed receiver envelope detector (squelch signal)
16–13 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
12 TXPWDV2I	Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHY_CTRL is enabled. Note that these circuits are shared with the battery charge circuit. Setting this to 1 does not power-down these circuits, unless the corresponding bit in the battery charger is also set for power-down.

Table continues on the next page...

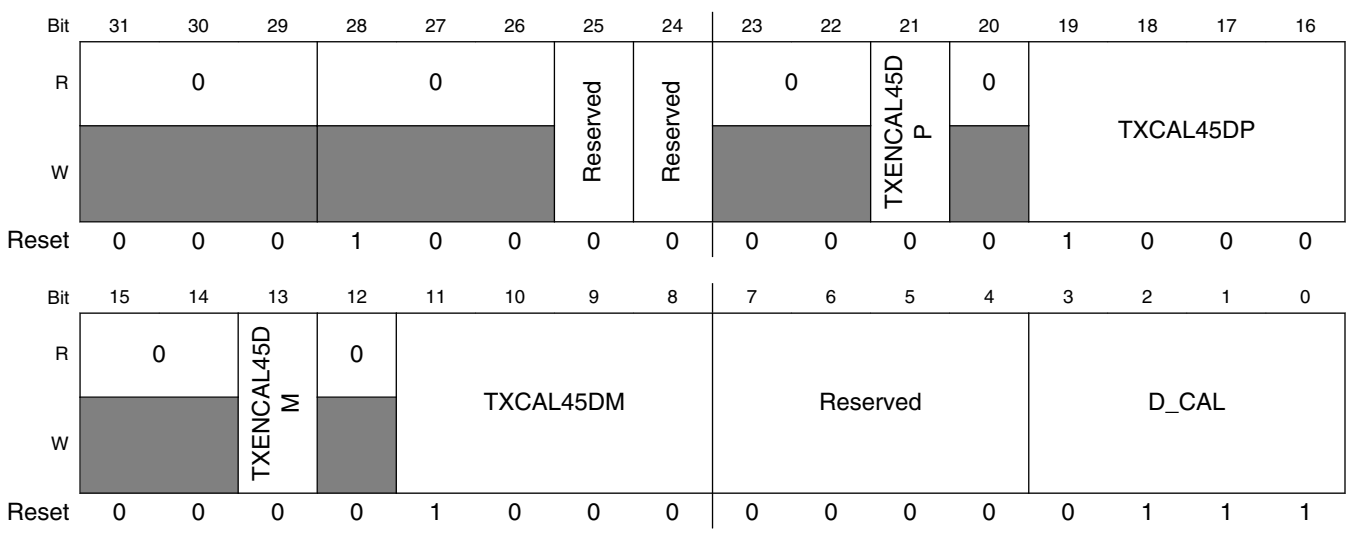
USBPHY_PWD*n* field descriptions (continued)

Field	Description
	0 Normal operation. 1 Power-down the USB PHY transmit V-to-I converter and the current mirror
11 TXPWDIBIAS	Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHY_CTRL is enabled. Note that these circuits are shared with the battery charge circuit. Setting this bit to 1 does not power-down these circuits, unless the corresponding bit in the battery charger is also set for power-down. 0 Normal operation 1 Power-down the USB PHY current bias block for the transmitter. This bit should be set only when the USB is in suspend mode. This effectively powers down the entire USB transmit path
10 TXPWDFS	Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHY_CTRL is enabled. 0 Normal operation. 1 Power-down the USB full-speed drivers. This turns off the current starvation sources and puts the drivers into high-impedance output
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

55.4.2 USB PHY Transmitter Control Register (USBPHY_TX*n*)

The USB PHY Transmitter Control Register handles the transmit controls.

Address: 4035_0000h base + 10h offset + (4d × i), where i=0d to 3d



USBPHY_TX*n* field descriptions

Field	Description
31–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
28–26 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
25 Reserved	This field is reserved. This bit field must remain at value 1'b0.
24 Reserved	This field is reserved. This bit field must remain at value 1'b0.
23–22 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
21 TXENCAL45DP	Enable resistance calibration on DP.
20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–16 TXCAL45DP	Decode to trim the nominal 45Ω series termination resistance to the USB_DP output pin. Maximum resistance = 0000. Resistance is centered by design at 1000. Trimming this resistance will impact both the overshoot/undershoot of the Full Speed TX output and the amplitude of the High Speed TX output. 0000 +19.95% 0001 +17.35% 0010 +14.85% 0011 +12.46% 0100 +9.07% 0101 +5.87% 0110 +2.85% 0111 0% 1000 -2.70% 1001 -5.25% 1010 -7.67% 1011 -9.98% 1100 -12.17% 1101 -14.25% 1110 -18.14% 1111 -21.68%
15–14 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
13 TXENCAL45DM	Enable resistance calibration on DM.
12 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
11–8 TXCAL45DM	Decode to trim the nominal 45Ω series termination resistance to the USB_DM output pin. Maximum resistance = 0000. Resistance is centered by design at 1000.

Table continues on the next page...

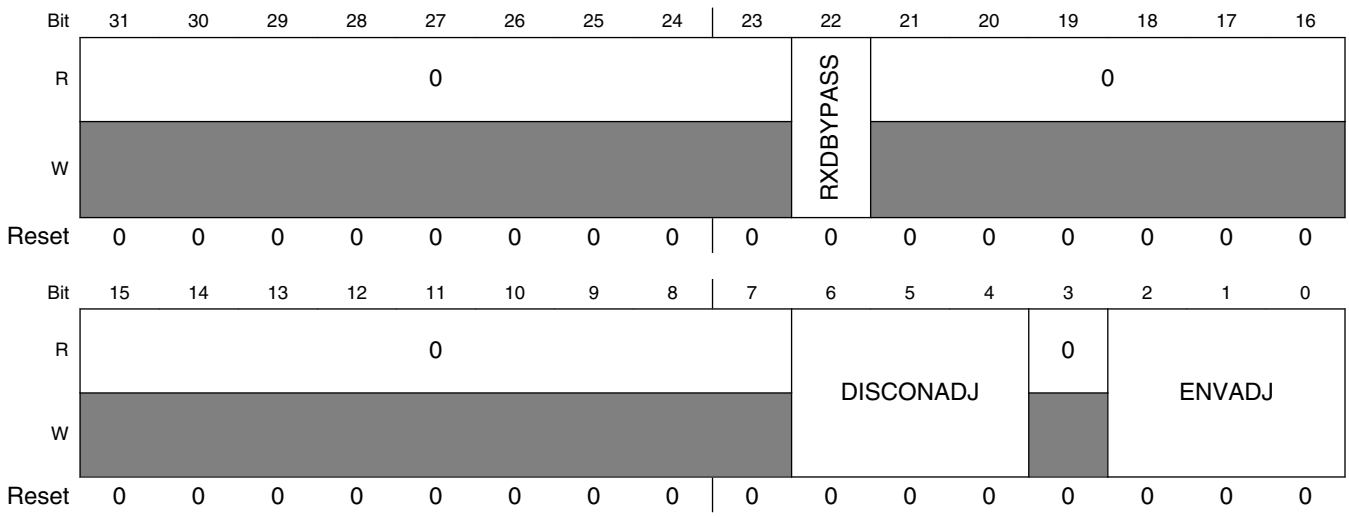
USBPHY_TX*n* field descriptions (continued)

Field	Description
	Trimming this resistance will impact both the overshoot/undershoot of the Full Speed TX output and the amplitude of the High Speed TX output. 0000 +19.95% 0001 +17.35% 0010 +14.85% 0011 +12.46% 0100 +9.07% 0101 +5.87% 0110 +2.85% 0111 0% 1000 -2.70% 1001 -5.25% 1010 -7.67% 1011 -9.98% 1100 -12.17% 1101 -14.25% 1110 -18.14% 1111 -21.68%
7–4 Reserved	This field is reserved. Reserved. **Note:** This bit should remain clear.
D_CAL	Decode to trim the nominal 17.78mA current source for the High Speed TX drivers on USB_DP and USB_DM. This current is directly proportional to the amplitude of the High Speed TX eye diagram. 0000 Maximum current, approximately 19% above nominal. 0111 Nominal 1111 Minimum current, approximately 19% below nominal.

55.4.3 USB PHY Receiver Control Register (USBPHY_RX*n*)

The USB PHY Receiver Control Register handles receive path controls.

Address: 4035_0000h base + 20h offset + (4d × i), where i=0d to 3d



USBPHY_RX*n* field descriptions

Field	Description
31–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22 RXDBYPASS	This test mode is intended for lab use only, replace FS differential receiver with DP single ended receiver. 0 Normal operation. 1 Use the output of the USB_DP single-ended receiver in place of the full-speed differential receiver
21–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 DISCONADJ	The DISCONADJ field adjusts the trip point for the disconnect detector. 000 Trip-Level Voltage is 0.56875 V 001 Trip-Level Voltage is 0.55000 V 010 Trip-Level Voltage is 0.58125 V 011 Trip-Level Voltage is 0.60000 V 1XX Reserved
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ENVADJ	The ENVADJ field adjusts the trip point for the envelope detector. Values shown below are nominal DC settings to indicate effect of changing these bits. AC values measured during compliance testing will be somewhat higher. 000 Trip-Level Voltage is 0.1000 V 001 Trip-Level Voltage is 0.1125 V

Table continues on the next page...

USBPHY_RX*n* field descriptions (continued)

Field	Description
	010 Trip-Level Voltage is 0.1250 V 011 Trip-Level Voltage is 0.0875 V 1XX Reserved

55.4.4 USB PHY General Control Register (USBPHY_CTRL*n*)

The USB PHY General Control Register handles OTG and Host controls. This register also includes interrupt enables and connectivity detect enables and results.

Address: 4035_0000h base + 30h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SFTRST	CLKGATE	UTMI_SUSPENDM	HOST_FORCE_LS_SE0	OTG_ID_VALUE	Reserved	Reserved	FSDLL_RST_EN	Reserved	Reserved	Reserved	ENAUTOCLR_PHY_PWD	ENAUTOCLR_CLKGATE	AUTORESUME_EN	Reserved	Reserved
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENUTMILEVEL3	ENUTMILEVEL2	Reserved	DEVPLUGIN_IRQ	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	ENDEVPLUGINDET	HOSTDISCONDETECT_IRQ	Reserved	ENHOSTDISCONDETECT	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBPHY_CTRL*n* field descriptions

Field	Description
31 SFTRST	Writing a 1 to this bit will soft-reset the USBPHY_PWD, USBPHY_TX, USBPHY_RX, and USBPHY_CTRL registers. Set to 0 to release the PHY from reset.
30 CLKGATE	Gate UTMI Clocks. Clear to 0 to run clocks. Set to 1 to gate clocks. Set this to save power while the USB is not actively being used. Configuration state is kept while the clock is gated. Note this bit can be auto-cleared if there is any wakeup event when USB is suspended while ENAUTOCLR_CLKGATE bit of USBPHY_CTRL is enabled.
29 UTMI_SUSPENDM	Used by the PHY to indicate a powered-down state. If all the power-down bits in the USBPHY_PWD are enabled, UTMI_SUSPENDM will be 0, otherwise 1 when USB controller entering into Suspend mode. UTMI_SUSPENDM is negative logic, as required by the UTMI specification.
28 HOST_FORCE_LS_SE0	Forces the next FS packet that is transmitted to have a EOP with low-speed timing. This bit is used in host mode for the resume sequence. After the packet is transferred, this bit is cleared. The design can use this function to force the LS SE0 or use the USBPHY_CTRL_UTMI_SUSPENDM to trigger this event when leaving suspend. This bit is used in conjunction with USBPHY_DEBUG_HOST_RESUME_DEBUG.
27 OTG_ID_VALUE	Indicates the results of USB_ID pin while monitoring the cable plugged into the Micro- or Mini-AB receptacle. False (0) is when ID resistance is less than Ra_Plug_ID, indicating host (A) side. True (1) is when ID resistance is greater than Rb_Plug_ID, indicating device (B) side. Similar to the function of the OTGID_STATUS bit in the USBPHY_STATUS register, but OTG_ID_VALUE has debounce and system clock synchronization logic to filter the glitches on the USB_ID pad.
26 Reserved	Reserved This field is reserved. This bit field has no effect for this module.
25 Reserved	Reserved This field is reserved. This bit field has no effect for this module.
24 FSDLL_RST_EN	Enables the feature to reset the FSDLL lock detection logic at the end of each TX packet.
23 Reserved	This field is reserved. This bit field has no effect for this module. This bit field must remain at value 1'b0.
22 Reserved	This field is reserved. This bit field has no effect for this module. This bit field must remain at value 1'b0.
21 Reserved	This field is reserved. This bit field must remain at value 1'b0.
20 ENAUTOCLR_PHY_PWD	Enables the feature to auto-clear the PWD register bits in USBPHY_PWD if there is wakeup event while USB is suspended. This should be enabled if needed to support auto wakeup without software interaction.
19 ENAUTOCLR_CLKGATE	Enables the feature to auto-clear the CLKGATE bit if there is wakeup event while USB is suspended. This should be enabled if needed to support auto wakeup without software interaction.

Table continues on the next page...

USBPHY_CTRL*n* field descriptions (continued)

Field	Description
18 AUTORESUME_EN	Enable the auto resume feature, when set, HW will use 32KHz clock to send Resume to respond to the device remote wakeup(for host mode only). It's useful when PLL is off and reference clock is also powered down.
17 Reserved	This field is reserved. This bit field has no effect for this module.
16 Reserved	This field is reserved. This bit field has no effect for this module.
15 ENUTMILEVEL3	Enables UTMI+ Level 3 operation for the USB HS PHY. This should be enabled if an Embedded Host use case needs to support an external FS Hub with a LS device connected.
14 ENUTMILEVEL2	Enables UTMI+ Level 2 operation for the USB HS PHY. This should be enabled if an Embedded Host use case needs to support a LS device.
13 Reserved	This field is reserved. This bit field must remain at value 1'b0.
12 DEVPLUGIN_IRQ	Indicates that the device is connected. Reset this bit by writing a 1 to the SCT clear address space and not by a general write.
11 Reserved	This field is reserved. This bit field has no effect for this module.
10 Reserved	This field is reserved. This bit field has no effect for this module.
9 Reserved	This field is reserved. This bit field has no effect for this module.
8 Reserved	This field is reserved. This bit field has no effect for this module.
7 Reserved	This field is reserved. This bit field has no effect for this module. This bit field must remain at value 1'b0.
6 Reserved	This field is reserved. This bit field has no effect for this module.
5 Reserved	This field is reserved. This bit field has no effect for this module.
4 ENDEVPLUGINDET	Enables non-standard resistive plugged-in detection This bit field controls connection of nominal 200kΩ resistors to both the USB_DP and USB_DM pins as one method of detecting when a USB cable is attached in device mode. This bit field must remain at a value of 1'b0 for normal USB data communication, or when using the USBHSDCD module for battery charger detection per the *USB Battery Charger Specification Revision 1.2* or any other detection mechanism for USB cable plugin. The results of this detection method are reported in USBPHY_STATUS[6]. 0 Disables 200kΩ pullup resistors on USB_DP and USB_DM pins (Default) 1 Enables 200kΩ pullup resistors on USB_DP and USB_DM pins
3 HOSTDISCONDETECT_IRQ	Indicates that the device has disconnected in High-Speed mode. Reset this bit by writing a 1 to the SCT clear address space and not by a general write.
2 Reserved	This field is reserved. This bit field has no effect for this module.

Table continues on the next page...

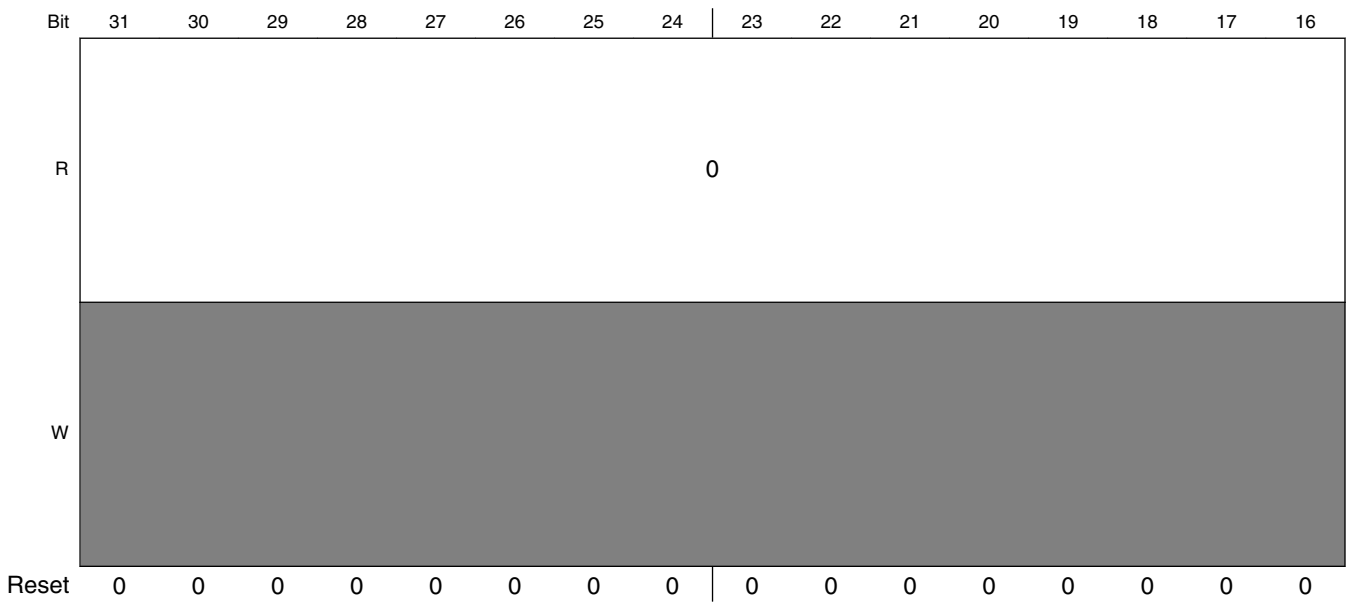
USBPHY_CTRL*n* field descriptions (continued)

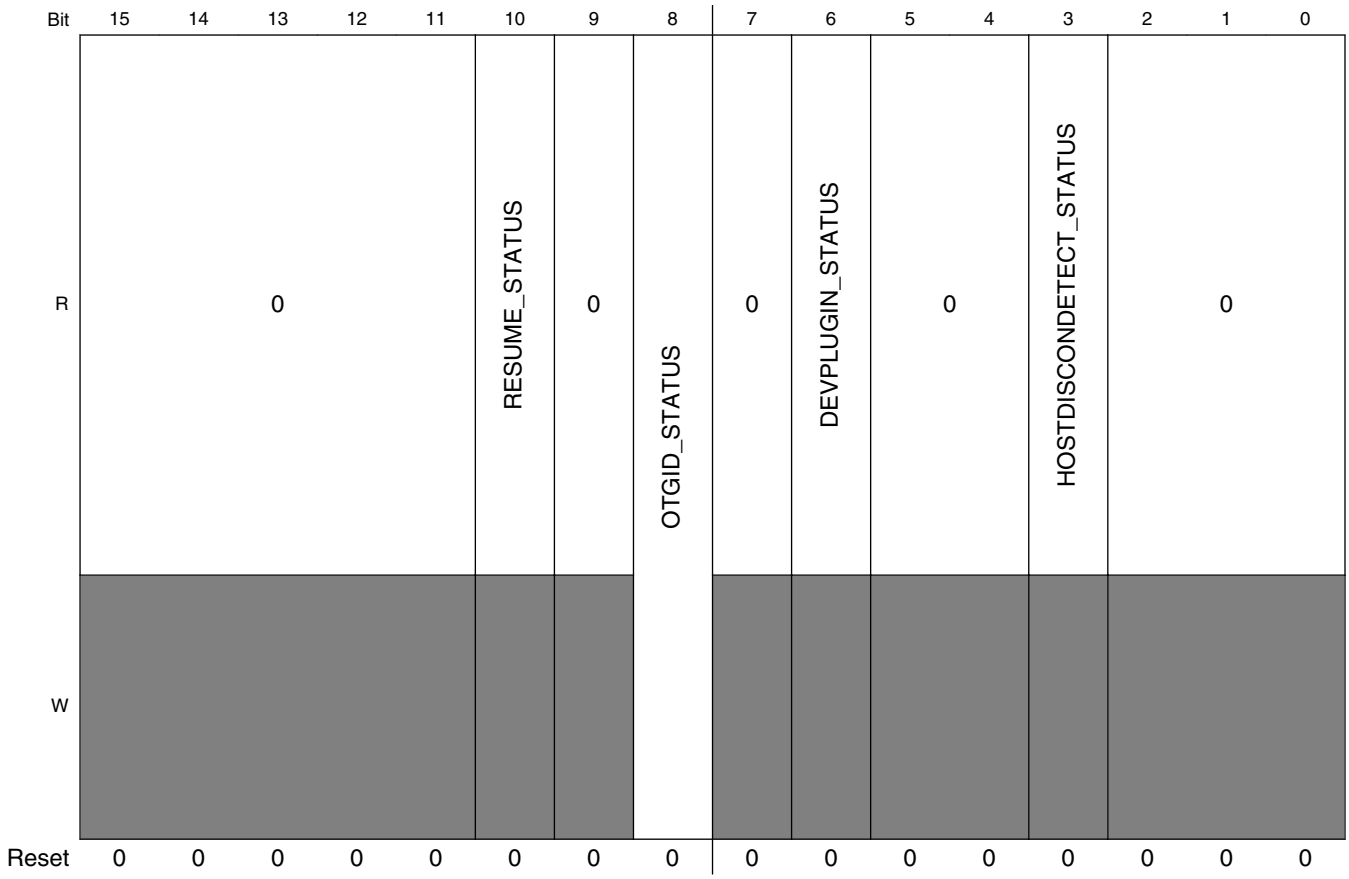
Field	Description
1 ENHOSTDISCONDETECT	For host mode, enables high-speed disconnect detector. This signal allows the override of enabling the detection that is normally done in the UTMI controller. The UTMI controller enables this circuit whenever the host sends a start-of-frame packet. It shall be set after HS device is connected.
0 Reserved	This field is reserved. This bit field has no effect for this module.

55.4.5 USB PHY Status Register (USBPHY_STATUS)

The USB PHY Status Register holds results of IRQ and other detects.

Address: 4035_0000h base + 40h offset = 4035_0040h





USBPHY_STATUS field descriptions

Field	Description
31–11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 RESUME_STATUS	Indicates that the host is sending a wake-up after Suspend and has triggered an interrupt.
9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 OTGID_STATUS	Indicates the results of USB_ID pin on the USB cable plugged into the local Micro- or Mini-AB receptacle. False (0) is when ID resistance to ground is less than Ra_Plug_ID, indicating host (A) side. True (1) is when ID resistance is greater than Rb_Plug_ID, indicating device (B) side.
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 DEVPLUGIN_STATUS	Status indicator for non-standard resistive plugged-in detection Indicates that the device has been connected on the USB_DP and USB_DM lines using the non-standard resistive plugged-in detection method controlled by USBPHY_CTRL[4]. When a USB cable attached to a remote host is attached to the local device, the 15kΩ host pulldowns will override the high value resistors used in this detection method.

Table continues on the next page...

USBPHY_STATUS field descriptions (continued)

Field	Description
	0 No attachment to a USB host is detected 1 Cable attachment to a USB host is detected
5–4 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
3 HOSTDISCONDETECT_STATUS	Indicates at the local host (downstream) port that the remote device has disconnected while in High-Speed mode. 0 USB cable disconnect has not been detected at the local host 1 USB cable disconnect has been detected at the local host
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

55.4.6 USB PHY Debug Register 0 (USBPHY_DEBUG0*n*)

This register is used to debug the USB PHY.

Address: 4035_0000h base + 50h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	CLKGATE	HOST_RESUME_DEBUG	SQUELCHRESETLENGTH				ENSQUELCHRESET	0			SQUELCHRESETCOUNT				
W																
Reset	0	1	1	1	1	1	1	1	0	0	0	1	1	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			ENTX2RXCOUNT	TX2RXCOUNT				0		ENHSTPULLDOWN	HSTPULLDOWN			DEBUG_INTERFACE_HOLD	OTGIDPIOLOCK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBPHY_DEBUG0*n* field descriptions

Field	Description
31 Reserved	Reserved This field is reserved.

Table continues on the next page...

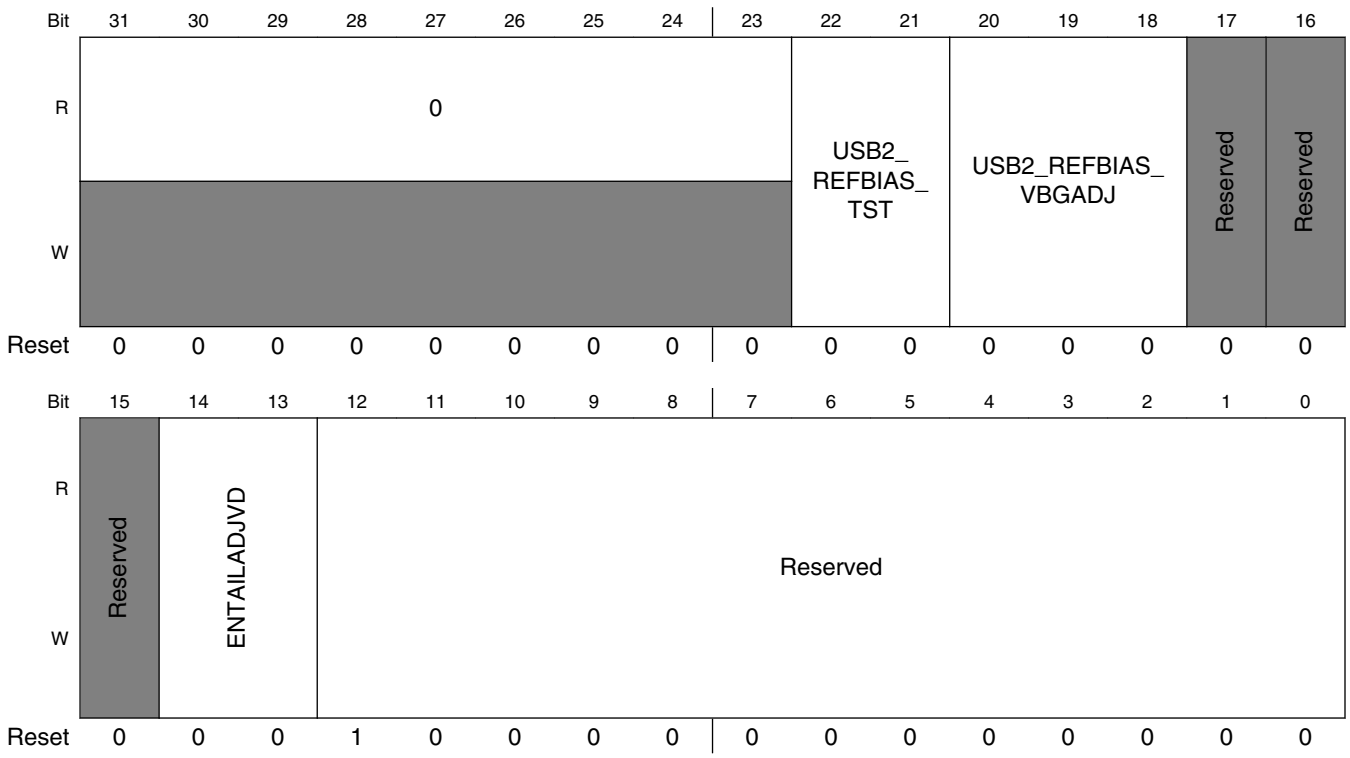
USBPHY_DEBUG0*n* field descriptions (continued)

Field	Description
	This read-only field is reserved and always has the value 0.
30 CLKGATE	Gate Test Clocks. Clear to 0 for running clocks. Set to 1 to gate clocks. Set this to save power while the USB is not actively being used. Configuration state is kept while the clock is gated.
29 HOST_RESUME_DEBUG	Choose to trigger the host resume SE0 with HOST_FORCE_LS_SE0 = 0 or UTMI_SUSPEND = 1.
28–25 SQUELCHRESETLENGTH	Duration of RESET in terms of the number of 480-MHz cycles.
24 ENSQUELCHRESET	Set bit to allow squelch to reset high-speed receive.
23–21 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
20–16 SQUELCHRESETCOUNT	Delay in between the detection of squelch to the reset of high-speed RX.
15–13 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
12 ENTX2RXCOUNT	Set this bit to allow a countdown to transition in between TX and RX.
11–8 TX2RXCOUNT	Delay in between the end of transmit to the beginning of receive. This is a Johnson count value and thus will count to 8.
7–6 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
5–4 ENHSTPULLDOWN	This bit field selects host pulldown overdrive mode. Set bit 5 to value 1'b1 to override the control of the USB_DP 15kΩ pulldown. Set bit 4 to value 1'b1 to override the control of the USB_DM 15Ω pulldown. Clear both bits to value 2'b00 to disable the host pulldown overdrive mode. When in host pulldown overdrive mode, the connection of the individual pulldown resistors is further controlled by the USBPHY_DEBUG[3:2] bit field.
3–2 HSTPULLDOWN	This bit field selects whether to connect pulldown resistors on the USB_DP/USB_DM pins if the corresponding pulldown overdrive mode is enabled through USBPHY_DEBUG[5:4} Set bit 3 to value 1'b1 to connect the 15Ω pulldown on USB_DP line. Set bit 2 to value 1'b1 to connect the 15Ω pulldown on the USB_DM line. Clear both bits to value 2'b00 to disconnect the pulldowns in override mode.
1 DEBUG_INTERFACE_HOLD	Use holding registers to assist in timing for external UTMI interface.
0 OTGIDPIOLOCK	Once OTG ID from USBPHY_STATUS_OTGID_STATUS is sampled, use this to hold the value. This is to save power for the comparators that are used to determine the ID status.

55.4.7 UTMI Debug Status Register 1 (USBPHY_DEBUG1*n*)

Chooses the muxing of the debug register to be shown in USBPHYx_DEBUG0_STATUS.

Address: 4035_0000h base + 70h offset + (4d × i), where i=0d to 3d



USBPHY_DEBUG1*n* field descriptions

Field	Description
31–23 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
22–21 USB2_ REFBIAS_TST	Bias current control for usb2_phy
20–18 USB2_ REFBIAS_ VBGADJ	Adjustment bits on bandgap
17 Reserved	This field is reserved. This field is unimplemented and not for use. This bit field must remain at value 1'b0.
16 Reserved	This field is reserved. This field is unimplemented and not for use. This bit field must remain at value 1'b0.

Table continues on the next page...

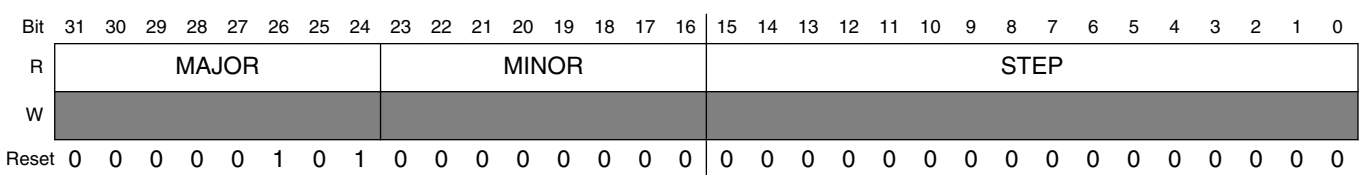
USBPHY_DEBUG1*n* field descriptions (continued)

Field	Description
15 Reserved	This field is reserved. This field is unimplemented and not for use. This bit field must remain at value 1'b0.
14–13 ENTAILADJVD	Delay increment of the rise of squelch: 00 Delay is nominal 01 Delay is +20% 10 Delay is -20% 11 Delay is -40%
Reserved	Reserved This field is reserved. This bit should remain clear.

55.4.8 UTMI RTL Version (USBPHY_VERSION)

Fields for RTL Version.

Address: 4035_0000h base + 80h offset = 4035_0080h



USBPHY_VERSION field descriptions

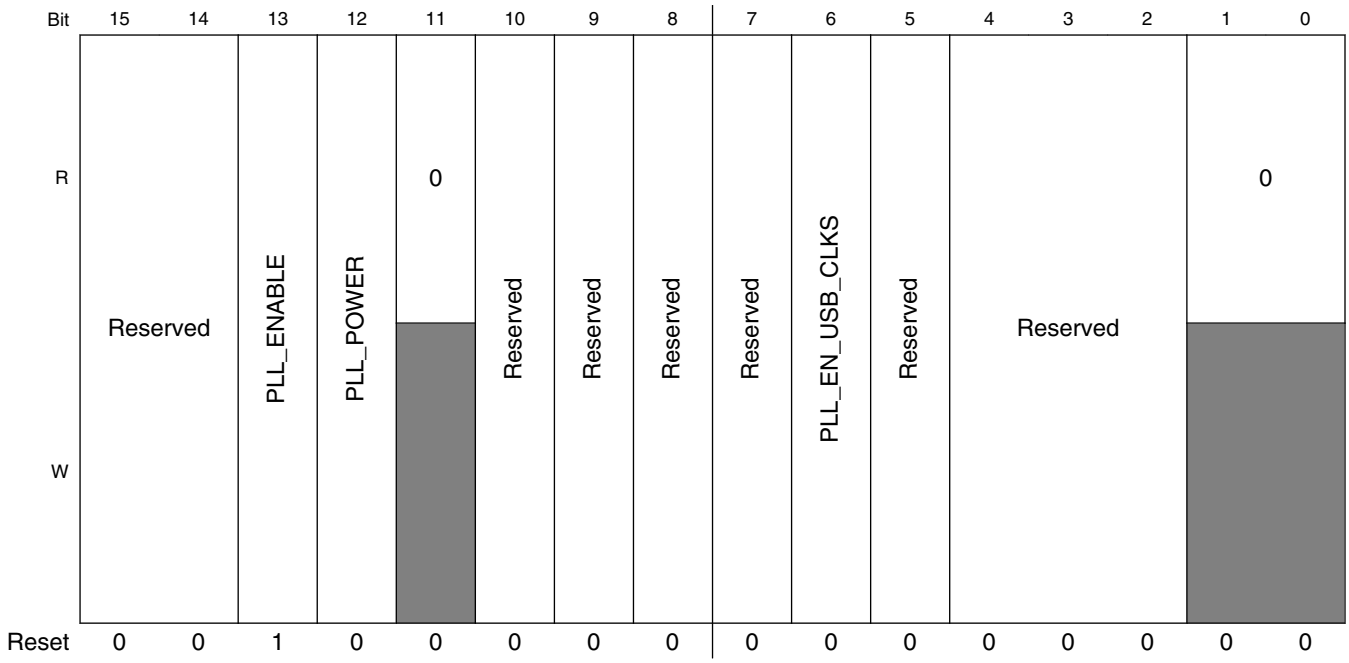
Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version
STEP	Fixed read-only value reflecting the stepping of the RTL version.

55.4.9 USB PHY PLL Control/Status Register (USBPHY_PLL_SIC*n*)

This register configures the 480 MHz USB PHY PLL.

Address: 4035_0000h base + A0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PLL_LOCK	0						PLL_DIV_SEL			PLL_REG_ENABLE	REFBIAS_PWD	REFBIAS_PWD_SEL	0	Reserved	PLL_BYPASS
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	1



USBPHY_PLL_SIC*n* field descriptions

Field	Description
31 PLL_LOCK	USB PLL lock status indicator 0 PLL is not currently locked 1 PLL is currently locked
30–25 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
24–22 PLL_DIV_SEL	This field controls the USB PLL feedback loop divider. Valid range for divider values: 13-240. Fout = Fin * div_select. The USB PLL is designed to produce a 480 MHz output clock. This bit field allows use of different frequency signals for the PLL reference clock input connected to the OSCCLK signal from the system oscillator. When override is enabled through USBPHY_TRIM_OVERRIDE_EN[0], the USB PLL will use this register value. 000 Divide by 13 001 Divide by 15 010 Divide by 16 011 Divide by 20 100 Divide by 22 101 Divide by 25 110 Divide by 30 111 Divide by 240
21 PLL_REG_ENABLE	This field controls the USB PLL regulator, set to enable the regulator. SW must set this bit 15 us before settiing PLL_POWER to avoid glitches on PLL output clock.
20 REFBIAS_PWD	Power down the reference bias This bit is only used when REFBIAS_PWD_SEL is set to 1.

Table continues on the next page...

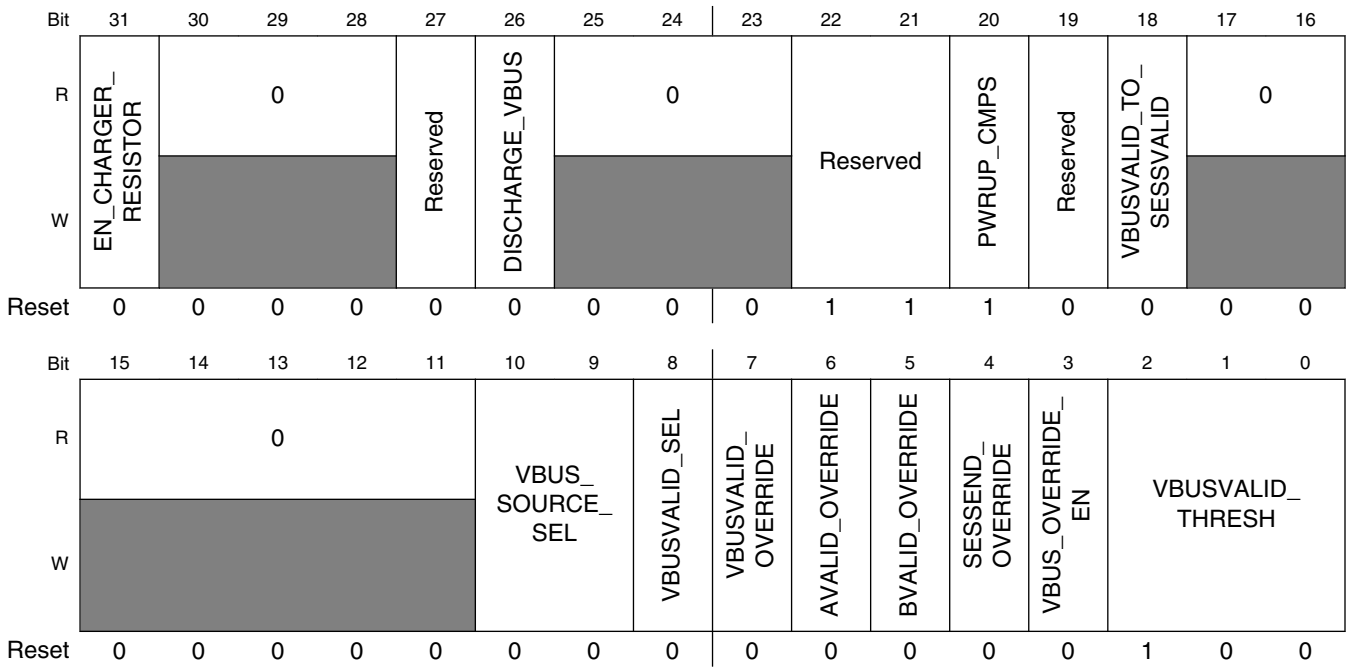
USBPHY_PLL_SIC*n* field descriptions (continued)

Field	Description
19 REFBIAS_PWD_SEL	Reference bias power down select. 0 Selects PLL_POWER to control the reference bias 1 Selects REFBIAS_PWD to control the reference bias.
18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17 Reserved	This field is reserved.
16 PLL_BYPASS	Bypass the USB PLL.
15–14 Reserved	This field is reserved. This bit field has no effect for this module.
13 PLL_ENABLE	Enables the clock output from the USB PLL. The real PLL output enable signal is also controlled by PLL power up control. Hardware will disable the PLL output before power down PLL, and enable the PLL output after power up PLL. The software only needs to set it when initializing the PLL.
12 PLL_POWER	Power up the USB PLL. The real PLL power up is also controlled by hardware. Hardware will power down PLL when USB is suspended and the device doesn't use it.
11 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
10 Reserved	This field is reserved. This bit field must remain at value 1'b0.
9 Reserved	This field is reserved. This bit field must remain at value 1'b0.
8 Reserved	This field is reserved. This bit field must remain at value 1'b0.
7 Reserved	This field is reserved. This bit field must remain at value 1'b0.
6 PLL_EN_USB_CLKS	Enables the USB clock from PLL to USB PHY
5 Reserved	This field is reserved. This bit field must remain at value 1'b0.
4–2 Reserved	This field is reserved. This bit field has no effect for this module.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

55.4.10 USB PHY VBUS Detect Control Register (USBPHY_USB1_VBUS_DETECT*n*)

This register defines controls for USB VBUS detect and some additional out of band signaling functions.

Address: 4035_0000h base + C0h offset + (4d × i), where i=0d to 3d



USBPHY_USB1_VBUS_DETECT*n* field descriptions

Field	Description
31 EN_CHARGER_RESISTOR	Enables resistors used for an older method of resistive battery charger detection 0 Disable resistive charger detection resistors on USB_DP and USB_DP 1 Enable resistive charger detection resistors on USB_DP and USB_DP
30–28 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
27 Reserved	This field is reserved. This bit field has no effect for this module.
26 DISCHARGE_VBUS	Controls VBUS discharge resistor This bit field controls a nominal 22kΩ resistor between the USB1_VBUS pin and ground. It can be used to accelerate the fall of the VBUS signal at the end of a session.

Table continues on the next page...

USBPHY_USB1_VBUS_DETECT*n* field descriptions (continued)

Field	Description
	0 VBUS discharge resistor is disabled (Default) 1 VBUS discharge resistor is enabled
25–23 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
22–21 Reserved	This field is reserved. This bit field has no effect for this module.
20 PWRUP_CMPS	Enables the VBUS_VALID comparator Powers up the comparator used for the VBUS_VALID detector. This bit field can be reset to value 1'b0 to save power if the internal VBUS_VALID comparator is not used. 0 Powers down the VBUS_VALID comparator 1 Enables the VBUS_VALID comparator (default)
19 Reserved	Reserved This field is reserved. This bit field has no effect for this module.
18 VBUSVALID_ TO_SESSVALID	Selects the comparator used for VBUS_VALID This bit field controls the comparator used to report the VBUS_VALID results in USBPHY_USB1_VBUS_DETECT[3] between the VBUS_VALID comparator and the Session Valid comparator. The VBUS_VALID comparator is the most accurate and has a programmable threshold set by USBPHY_USB_VBUS_DETECT[2:0]. The Session Valid comparator may be useful in systems using non-standard VBUS voltages. The mux selection in this bit field happens before any VBUS_VALID selection controlled by the USBPHY_USB1_VBUS_DETECT[10:8] bits. 0 Use the VBUS_VALID comparator for VBUS_VALID results 1 Use the Session End comparator for VBUS_VALID results. The Session End threshold is >0.8V and <4.0V.
17–11 Reserved	Reserved This field is reserved. This read-only field is reserved and always has the value 0.
10–9 VBUS_ SOURCE_SEL	Selects the source of the VBUS_VALID signal reported to the USB controller This is one of the bit fields that selects the source of the VBUS_VALID signal reported to the USB controller. The VBUS_VALID source selections in this bit field only take effect if both USBPHY_USB1_VBUS_DETECT[8] and USBPHY_USB1_VBUS_DETECT[3] each have the value 1'b0. This bit field does not impact the VBUS_VALID value reported in USBPHY_USB1_VBUS_DET_STAT[3]. 00 Use the VBUS_VALID comparator results for signal reported to the USB controller (Default) 01 Use the Session Valid comparator results for signal reported to the USB controller 10 Use the Session Valid comparator results for signal reported to the USB controller 11 Reserved, do not use

Table continues on the next page...

USBPHY_USB1_VBUS_DETECT*n* field descriptions (continued)

Field	Description
8 VBUSVALID_ SEL	Selects the source of the VBUS_VALID signal reported to the USB controller This is one of the bit fields that selects the source of the VBUS_VALID signal reported to the USB controller. The VBUS_VALID source selection in this bit field only takes effect if USBPHY_USB1_VBUS_DETECT[3] has the value 1'b0. This bit field does not impact the VBUS_VALID value reported in USBPHY_USB1_VBUS_DET_STAT[3]. 0 Use the VBUS_VALID comparator results for signal reported to the USB controller (Default) 1 Use the VBUS_VALID_3V detector results for signal reported to the USB controller
7 VBUSVALID_ OVERRIDE	Override value for VBUS_VALID signal sent to USB controller The bit field provides the value for VBUS_VALID reported to the USB controller if the value of USBPHY_USB1_VBUS_DETECT[3] is set to 1'b1. The value of this bit field does not affect the value of USBPHY_USB1_VBUS_DET_STAT[3].
6 AVALID_ OVERRIDE	Override value for A-Device Session Valid The bit field provides the value for USBPHY_USB1_VBUS_DET_STAT[2] if USBPHY_USB_VBUS_DETECT[3] is set to value 1'b1.
5 BVALID_ OVERRIDE	Override value for B-Device Session Valid The bit field provides the value for USBPHY_USB1_VBUS_DET_STAT[1] if USBPHY_USB_VBUS_DETECT[3] is set to value 1'b1.
4 SESSEND_ OVERRIDE	Override value for SESSEND The bit field provides the value for USBPHY_USB1_VBUS_DET_STAT[0] if USBPHY_USB_VBUS_DETECT[3] is set to value 1'b1.
3 VBUS_ OVERRIDE_EN	VBUS detect signal override enable This bit field allows SW to override the results from the VBUS_VALID and Session Valid comparators using the values in USBPHY_USB1_VBUS_DETECT[7:4]. The VBUS_VALID, AVALID, BVALID, and SESSEND signals sent to the USB controller are each affected by these bit selections. The values reported for AVALID, BVALID, and SESSEND in USBPHY_USB1_VBUS_DET_STAT[2:0] are also affected but the value reported for VBUS_VALID in USBPHY_USB1_VBUS_DET_STAT[3] is not affected. This override method may be useful if VBUS detection is not done with the internal VBUS_VALID or Session End comparators. 0 Use the results of the internal VBUS_VALID and Session Valid comparators for VBUS_VALID, AVALID, BVALID, and SESSEND (Default) 1 Use the override values for VBUS_VALID, AVALID, BVALID, and SESSEND
VBUSVALID_ THRESH	Sets the threshold for the VBUSVALID comparator. This comparator is the most accurate method to determine the presence of 5v, and includes hystersis to minimize the need for software debounce of the detection. This comparator has ~50mV of hystersis to prevent chattering at the comparator trip point. 000 4.0 V 001 4.1 V 010 4.2 V 011 4.3 V 100 4.4 V (Default)

Table continues on the next page...

USBPHY_USB1_VBUS_DETECT*n* field descriptions (continued)

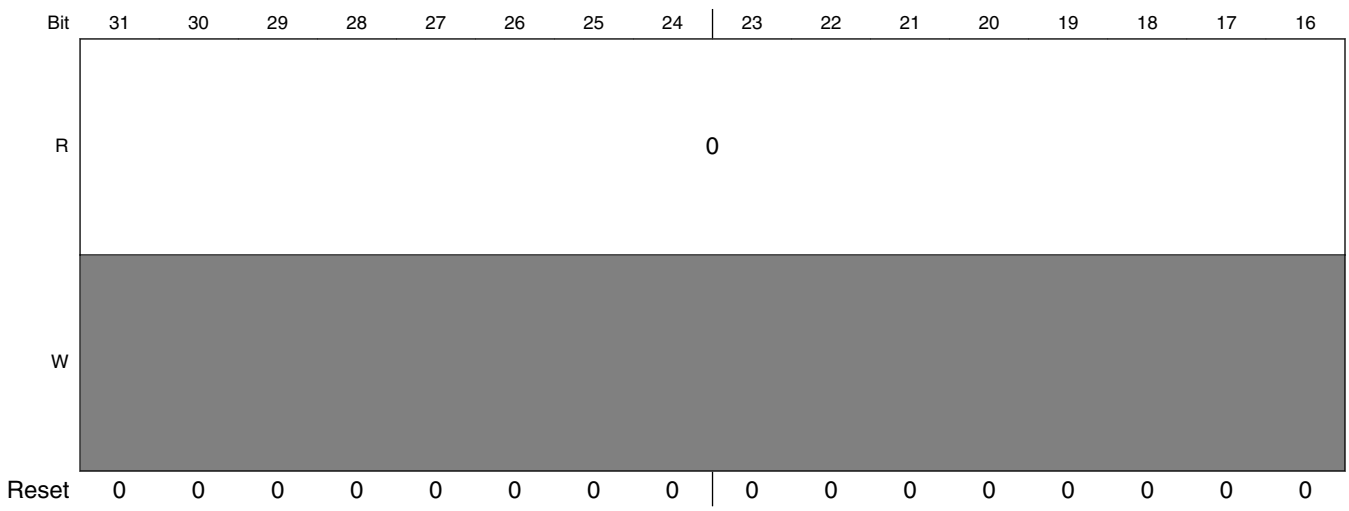
Field	Description
	101 4.5 V 110 4.6 V 111 4.7 V

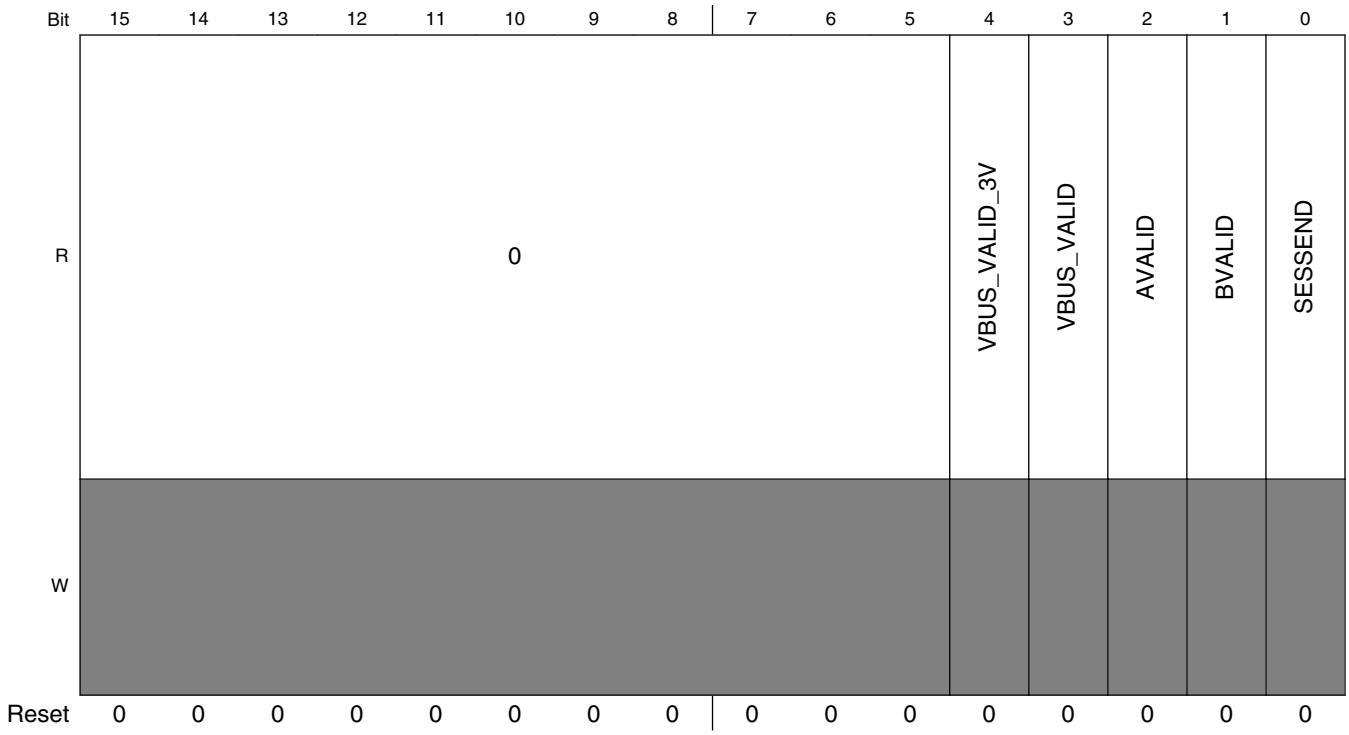
55.4.11 USB PHY VBUS Detector Status Register (USBPHY_USB1_VBUS_DET_STAT)

This register allows observation of status for USB VBUS detect functions.

The values reported in this register are synchronized by the apb_clk.

Address: 4035_0000h base + D0h offset = 4035_00D0h





USBPHY_USB1_VBUS_DET_STAT field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 VBUS_VALID_3V	VBUS_VALID_3V detector status The VBUS_VALID_3V detector has a lower threshold for the voltage on the USB1_VBUS pin than either the Session Valid or VBUS_VALID comparators. This signal may be useful for applications using a non-standard VBUS voltage. 0 VBUS voltage is below VBUS_VALID_3V threshold 1 VBUS voltage is above VBUS_VALID_3V threshold
3 VBUS_VALID	VBUS voltage status This bit field shows the result of VBUS_VALID detection for the USB1_VBUS pin. The VBUS_VALID comparator used is selected by USBPHY_USB1_VBUS_DETECT[18]. The VBUS_VALID source is not affected by the values of USBPHY_USB1_VBUS_DETECT[7, 3] and cannot be overwritten by SW. 0 VBUS is below the comparator threshold 1 VBUS is above the comparator threshold
2 AVALID	A-Device Session Valid status A-Device Session Valid status, determined by the Session Valid comparator. The default value of this bit is determined by the voltage on the USB1_VBUS pin. This bit can be overwritten by SW using the USBPHY_VBUS_DETECT[6, 3] bit fields.

Table continues on the next page...

USBPHY_USB1_VBUS_DET_STAT field descriptions (continued)

Field	Description
	0 The VBUS voltage is below the Session Valid threshold 1 The VBUS voltage is above the Session Valid threshold
1 BVALID	B-Device Session Valid status B-Device Session Valid status, determined by the Session Valid comparator. The default value of this bit is determined by the voltage on the USB1_VBUS pin. This bit can be overwritten by SW using the USBPHY_VBUS_DETECT[5, 3] bit fields. 0 The VBUS voltage is below the Session Valid threshold 1 The VBUS voltage is above the Session Valid threshold
0 SESSEND	Session End indicator Session End status, value inverted from Session Valid comparator. The default value of this bit is determined by the voltage on the USB1_VBUS pin. This bit can be overwritten by SW using the USBPHY_VBUS_DETECT[4, 3] bit fields. 0 The VBUS voltage is above the Session Valid threshold 1 The VBUS voltage is below the Session Valid threshold

55.4.12 USB PHY Charger Detect Control Register (USBPHY_USB1_CHRG_DETECT*n*)

This register defines controls for USB Battery Charging detection functions.

Address: 4035_0000h base + E0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								BGR_ IBIAS	0						
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													PULLUP_DP	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBPHY_USB1_CHRG_DETECT*n* field descriptions

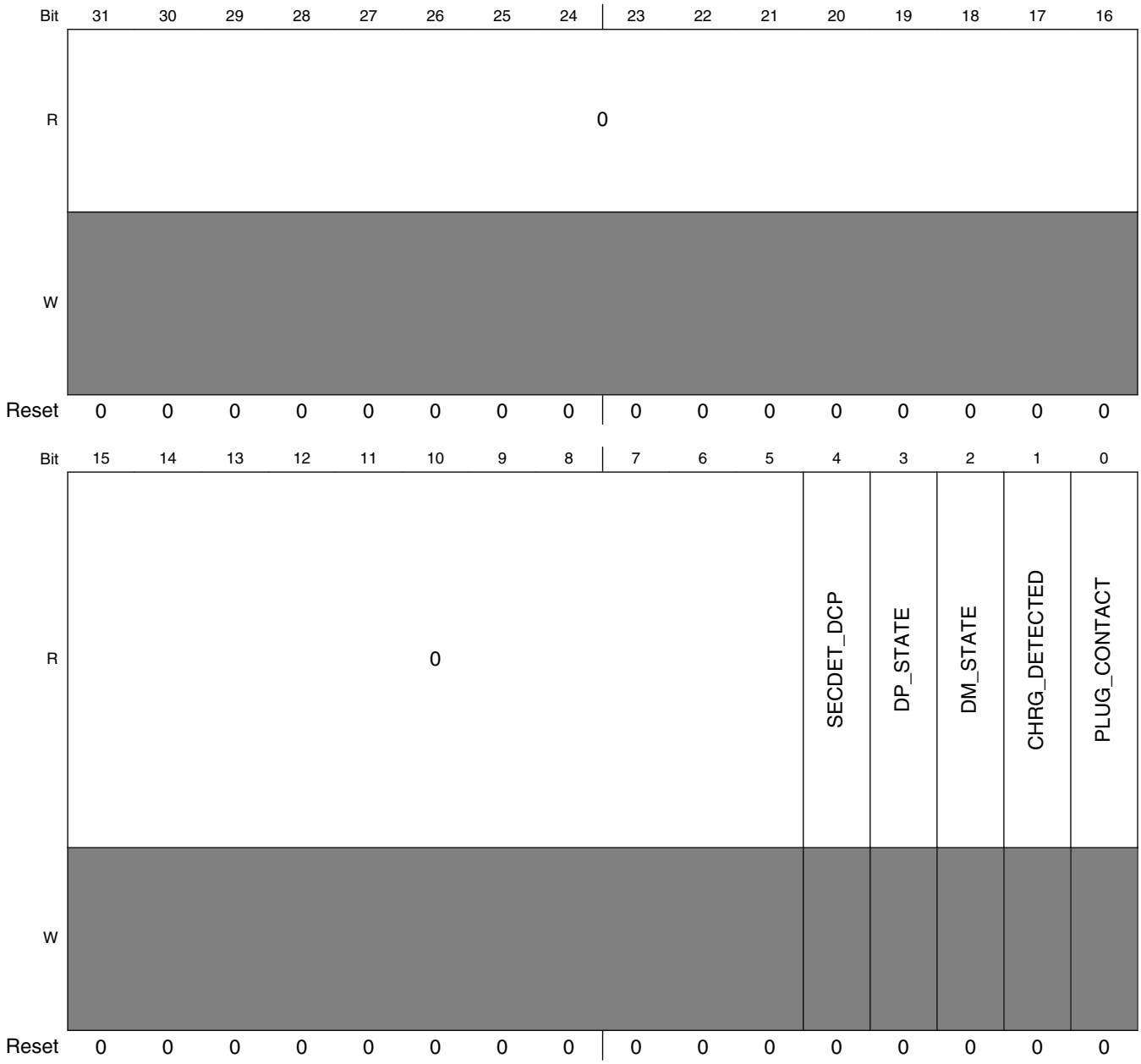
Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23 BGR_IBIAS	USB charge detector bias current reference This bit determines the reference for the bias current of the USB charge detector. This bit should always be set to 1. 0 Bias current is derived from the USB PHY internal current generator. 1 Bias current is derived from the reference generator of the bandgap.
22–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 PULLUP_DP	This bit is used to pull up DP, for digital charge detect.
1 Reserved	This field is reserved. This bit field has no effect for this module.
0 Reserved	This field is reserved. This bit field was named FORCE_DETECT on other products. The signal from this bit field has no useful function on this product. The bit field must remain at value 1'b0 to avoid interfering with USB Battery Charging detection.

55.4.13 USB PHY Charger Detect Status Register (USBPHY_USB1_CHRG_DET_STAT)

The control of the USB Battery Charging detection functions has been moved to the USBHSDCD instantiation of the USBDCD module for this product.

For standards-based charger detection purposes, the USBHSDCD registers should be used rather than this register. However, the status values in this register may be useful for debugging or in case other detection methods are used.

Address: 4035_0000h base + F0h offset = 4035_00F0h



USBPHY_USB1_CHRG_DET_STAT field descriptions

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 SECDET_DCP	Battery Charging Secondary Detection phase output During the USB Battery Charging Secondary Detection phase using the USBHSDCD module, this bit field indicates which kind of Charging Port was detected. 0 Charging Downstream Port (CDP) has been detected 1 Downstream Charging Port (DCP) has been detected

Table continues on the next page...

USBPHY_USB1_CHRG_DET_STAT field descriptions (continued)

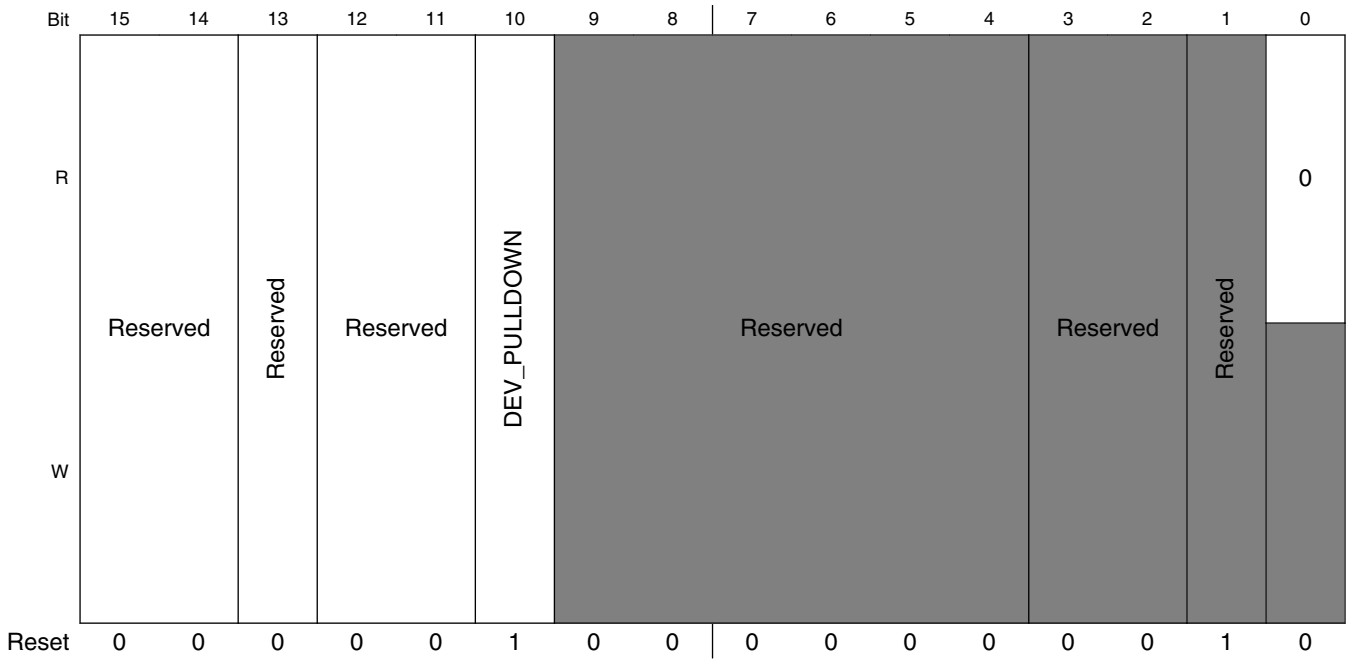
Field	Description
3 DP_STATE	Single ended receiver output for the USB_DP pin, from charger detection circuits. 0 USB_DP pin voltage is < 0.8V 1 USB_DP pin voltage is > 2.0V
2 DM_STATE	Single ended receiver output for the USB_DM pin, from charger detection circuits. 0 USB_DM pin voltage is < 0.8V 1 USB_DM pin voltage is > 2.0V
1 CHRG_ DETECTED	Battery Charging Primary Detection phase output During the USB Battery Charging Primary Detection phase using the USBHSDCD module, this bit field indicates whether a Standard Downstream Port or Charging Port was detected. 0 Standard Downstream Port (SDP) has been detected 1 Charging Port has been detected
0 PLUG_ CONTACT	Battery Charging Data Contact Detection phase output During the Data Contact Detection phase per the *USB Battery Charging Specification Revision 1.2* using the USBHSDCD module, this bit field indicates whether a USB cable has been attached between the remote host and the local device. 0 No USB cable attachment has been detected 1 A USB cable attachment between the device and host has been detected

55.4.14 USB PHY Analog Control Register (USBPHY_ANACTRL*n*)

The USBPHY_ANACTRL register a bit field to allow an additional type of control of 15 kohm pulldown resistors on both USB_DP and USB_DM pins.

Address: 4035_0000h base + 100h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved							0							
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0



USBPHY_ANACTRL*n* field descriptions

Field	Description
31 Reserved	This field is reserved.
30–23 Reserved	This field is reserved. This bit is unimplemented and not for use. For normal USB operation this bit field must remain cleared at value 8'h00.
22–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–14 Reserved	This field is reserved. This field is unimplemented and not for use. For normal USB operation, this bit field must remain at value 2'b00.
13 Reserved	This field is reserved. This field is unimplemented and not for use. For normal USB operation, this bit field must remain at value 1'b0.
12–11 Reserved	This field is reserved. This field is unimplemented and not for use. For normal USB operation, this bit field must remain at value 2'b00.
10 DEV_ PULLDOWN	Setting this field to 1'b1 will enable the 15kΩ pulldown resistors on both USB_DP and USB_DM pins. This feature can be used in device mode while the USB cable is disconnected to keep the data pins at known values, avoiding unnecessary interrupts from the single ended receivers. This bit must be reset to 1'b0 during normal USB data communication in device mode, or while battery charger detection using the USBHSDCD module is used. 0 The 15kΩ nominal pulldowns on the USB_DP and USB_DM pinsare disabled in device mode. 1 The 15kΩ nominal pulldowns on the USB_DP and USB_DM pinsare enabled in device mode.
9–4 Reserved	This field is reserved. This field is unimplemented and not for use. For normal USB operation, this bit field must remain at value 6'h00.

Table continues on the next page...

USBPHY_ANACTRL*n* field descriptions (continued)

Field	Description
3–2 Reserved	This field is reserved. This field is unimplemented and not for use. For normal USB operation, this bit field must remain at value 1'b00.
1 Reserved	This field is reserved. This field is unimplemented and not for use. For normal USB operation, this bit field must remain at value 1'b0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

55.4.15 USB PHY Loopback Control/Status Register (USBPHY_USB1_LOOPBACK*n*)

This register controls loopback testing of the USB PHY. Loopback mode is for test purposes only; it cannot be used during normal USB data communication.

Address: 4035_0000h base + 110h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TSTPKT							
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSTI_HSFS_MODE_EN	0						UTMO_DIG_TST1	UTMO_DIG_TST0	TSTI_TX_HIZ	TSTI_TX_EN	TSTI_TX_LS_MODE	TSTI_TX_HS_MODE	UTMI_DIG_TST1	UTMI_DIG_TST0	UTMI_TESTSTART
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBPHY_USB1_LOOPBACK*n* field descriptions

Field	Description
31–24 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
23–16 TSTPKT	Selects the packet data byte used for USB loopback testing in Pulse mode. Pulse mode is selected by setting USBPHY_USB1_LOOPBACK[2:1] to a value of 2'b01. Default value produces a data inversion at each unit interval.
15 TSTI_HSFS_ MODE_EN	Setting this bit field to value 1'b1 will enable the loopback test to dynamically change the packet speed. It will send a number of High Speed packets determined by USBPHY_USB1_LOOPBACK_HSFSCNT[TSTI_HS_NUMBER], followed by a number of Full Speed packets determined by USBPHY_USB1_LOOPBACK_HSFSCNT[TSTI_FS_NUMBER].
14–9 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
8 UTMO_DIG_ TST1	This read-only bit is a status bit for USB loopback test. Value = 1'b1 indicates a passing result. Test results are only valid while UTMI_TESTSTART is set to value 1'b1.
7 UTMO_DIG_ TST0	This read-only bit is a status bit for USB loopback test results. Value = 1'b0 indicates a passing result. Test results are only valid while UTMI_TESTSTART is set to value 1'b1.
6 TSTI_TX_HIZ	Sets TX Hi-Z for USB loopback test.
5 TSTI_TX_EN	Enable TX for USB loopback test.
4 TSTI_TX_LS_ MODE	Set to value 1'b1 to choose LS for USB loopback testing, set to value 1'b0 to choose HS or FS mode which is defined by TSTI1_TX_HS.
3 TSTI_TX_HS_ MODE	Select HS or FS mode for USB loopback testing. Set to value 1'b1 to choose HS for USB loopback testing, set to value 1'b0 to choose FS mode.
2 UTMI_DIG_TST1	Mode control for USB loopback test. Setting this bit to a value of 1'b1 while UTMI_DIG_TST0 remains at a value of 1'b0 selects Psuedorandom modes for the loopback test.
1 UTMI_DIG_TST0	Mode control for USB loopback test. Setting this bit to a value of 1'b1 while UTMI_DIG_TST1 remains at a value of 1'b0 selects Pulse mode for the loopback test.
0 UTMI_ TESTSTART	This bit enables the USB loopback test.

55.4.16 USB PHY Loopback Packet Number Select Register (USBPHY_USB1_LOOPBACK_HSFSCNT*n*)

Address: 4035_0000h base + 120h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TSTI_FS_NUMBER																TSTI_HS_NUMBER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

USBPHY_USB1_LOOPBACK_HSFSCNT*n* field descriptions

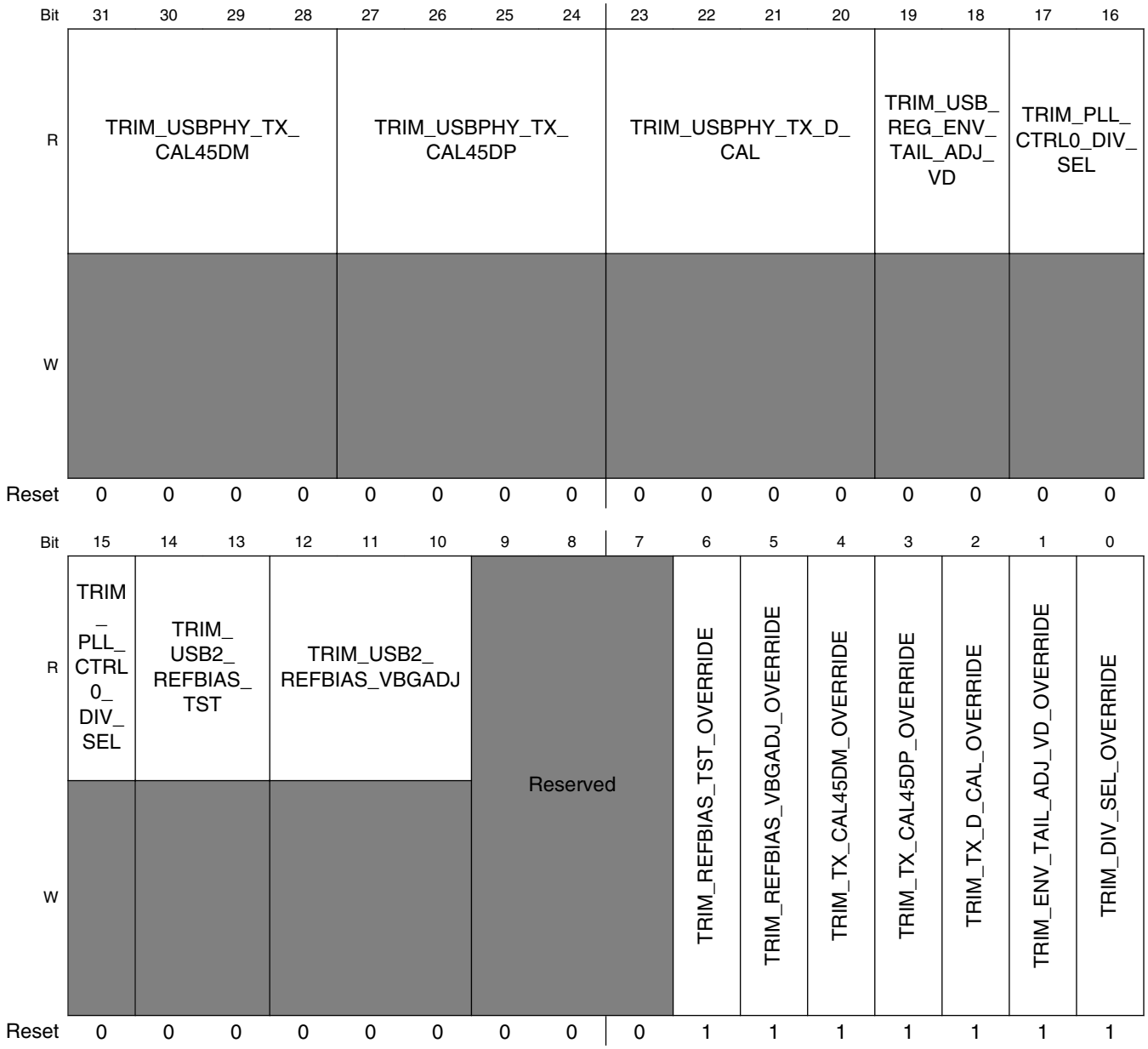
Field	Description
31–16 TSTI_FS_ NUMBER	Full speed packet number, used when USBPHY_USB1_LOOPBACK[TSTI_HSFS_MODE_EN] is set to value 1'b1.
TSTI_HS_ NUMBER	High speed packet number, used when USBPHY_USB1_LOOPBACK[TSTI_HSFS_MODE_EN] is set to value 1'b1.

55.4.17 USB PHY Trim Override Enable Register (USBPHY_TRIM_OVERRIDE_EN*n*)

The bit fields in this register allow observation of the default IFR settings of Phy parameters for TX_CAL45DM, TX_CAL45DP, TX_D_CAL, ENV_TAIL_ADJ, and PLL_DIV_SEL.

Additional bit fields also determine whether those values can be overridden by settings in the USBPHY_TX, USBPHY_DEBUG1, and USBPHY_PLL_SIC registers.

Address: 4035_0000h base + 130h offset + (4d × i), where i=0d to 3d



USBPHY_TRIM_OVERRIDE_EN*n* field descriptions

Field	Description
31–28 TRIM_USBPHY_TX_CAL45DM	IFR value of TX_CAL45DM.
27–24 TRIM_USBPHY_TX_CAL45DP	IFR value of TX_CAL45DP.
23–20 TRIM_USBPHY_TX_D_CAL	IFR value of TX_D_CAL.

Table continues on the next page...

USBPHY_TRIM_OVERRIDE_EN*n* field descriptions (continued)

Field	Description
19–18 TRIM_USB_ REG_ENV_ TAIL_ADJ_VD	IFR value of ENV_TAIL_ADJ.
17–15 TRIM_PLL_ CTRL0_DIV_SEL	IFR value of PLL_DIV_SEL.
14–13 TRIM_USB2_ REFBIAS_TST	Bias current control for usb2_phy
12–10 TRIM_USB2_ REFBIAS_ VBGADJ	Adjustment bits for bandgap
9–7 Reserved	This field is reserved. This field is unimplemented and not for use. This bit field must remain at value 1'b0.
6 TRIM_REFBIAS_ TST_OVERRIDE	Override enable for bias current control When this field is set, the register value in USBPHY_DEBUG1[22:21] will be used.
5 TRIM_REFBIAS_ VBGADJ_ OVERRIDE	Override enable for bandgap adjustment. When this field is set, the register value in USBPHY_DEBUG1[20:18] will be used.
4 TRIM_TX_ CAL45DM_ OVERRIDE	Override enable for TX_CAL45DM, when set, the register value in USBPHY_TX[11:8] will be used.
3 TRIM_TX_ CAL45DP_ OVERRIDE	Override enable for TX_CAL45DP, when set, the register value in USBPHY_TX[19:16] will be used.
2 TRIM_TX_D_ CAL_OVERRIDE	Override enable for TX_D_CAL, when set, the register value in USBPHY_TX[3:0] will be used.
1 TRIM_ENV_ TAIL_ADJ_VD_ OVERRIDE	Override enable for ENV_TAIL_ADJ, when set, the register value in USBPHY_DEBUG1[14:13] will be used.
0 TRIM_DIV_SEL_ OVERRIDE	Override enable for PLL_DIV_SEL, when set, the register value in USBPHY_PLL_SIC[1:0] will be used.

55.5 Register Macro Usage

A common operation is to update one field without disturbing the contents of the remaining fields in the register. Normally, this requires a read-modify-write (RMW) operation, where the CPU reads the register, modifies the target field, then writes the results back to the register. This is an expensive operation in terms of CPU cycles, because of the initial register read.

To address this issue, some hardware registers are implemented as a group, including registers that can be used to either set, clear, or toggle (SCT) individual bits of the primary register. When writing to an SCT register, all bits set to 1 perform the associated operation on the primary register, while all bits set to 0 are not affected. The SCT registers always read back 0, and should be considered write-only. The SCT registers are not implemented if the primary register is read-only.

With this architecture, it is possible to update one or more fields using only register writes. First, all bits of the target fields are cleared by a write to the associated clear register, then the desired value of the target fields is written to the set register. This sequence of two writes is referred to as a clear-set (CS) operation.

A CS operation does have one potential drawback. Whenever a field is modified, the hardware sees a value of 0 before the final value is written. For most fields, passing through the 0 state is not a problem. Nonetheless, this behavior is something to consider when using a CS operation.

Also, a CS operation is not required for fields that are one bit wide. While the CS operation works in this case, it is more efficient to simply set or clear the target bit (that is, one write instead of two). A simple set or clear operation is also atomic, while a CS operation is not.

Note that not all macros for set, clear, or toggle (SCT) are atomic. For registers that do not provide hardware support for this functionality, these macros are implemented as a sequence of read/modify/write operations. When atomic operation is required, the developer should pay attention to this detail, because unexpected behavior might result if an interrupt occurs in the middle of the critical section comprising the update sequence.

Chapter 56
USB DCD

56.1 Chip-specific USBDCD information

Table 56-1. Reference links to related information

Topic	Related module	Reference
Full description	USBDCD	USBDCD
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

56.1.1 USBDCD instantiation information

This device has only one USBHSDCD module instantiated in DP/DM PHY. The other USB instance has HSIC and ULPI ports.

56.2 Preface

56.2.1 References

The following publications are referenced in this document. For updates to these specifications, see www.usb.org.

- *USB Battery Charging Specification, Revision 1.1, USB Implementers Forum*
- *USB Battery Charging Specification (Including errata and ECNs through March 15, 2012), Revision 1.2, USB Implementers Forum*

- *Universal Serial Bus Specification Revision 2.0, USB Implementers Forum*
- *USB 2.0 Connect Timing Update ECN, USB Implementers Forum*

56.2.2 Acronyms and abbreviations

The following table contains acronyms and abbreviations used in this document.

Table 56-2. Acronyms and abbreviated terms

Term	Meaning
CDP	Charging Downstream Port, as defined in *USB Battery Charging Specification, Rev. 1.2*
Charging Port	A Charging Port supported by this module is either a CDP or a DCP
DCP	Dedicated Charging Port, as defined in *USB Battery Charging Specification, Rev. 1.2*
FS	Full speed (12 Mbit/s)
HS	High speed (480 Mbit/s)
I_{CFG_MAX}	Current limit for a USB device attached to an SDP, after configuration
I_{DEV_CHG}	Current limit for a USB device attached to a Charging Port
I_{DM_SINK}	Current sink for the D– line
I_{DP_SINK}	Current sink for the D+ line
I_{DP_SRC}	Current source for the D+ line
I_{SUSP}	Current drawn when the USB device is suspended
LDO	Low dropout
LS	Low Speed (1.5 Mbit/s)
N/A	Not applicable
OTG	On-The-Go
R_{DM_DWN}	D– pulldown resistance for data pin contact detect
SDP	Standard Downstream Port, as defined in *USB Battery Charging Specification, Rev 1.2*
V_{DAT_REF}	Data detect reference voltage for the voltage comparator
V_{DP_SRC}	Voltage source for the D+ line
V_{DM_SRC}	Voltage source for the D– line
V_{LGC}	Threshold voltage for logic high

56.2.3 Glossary

The following table shows a glossary of terms used in this document.

Table 56-3. Glossary of terms

Term	Definition
Transceiver	Module that implements the physical layer of the USB standard (FS or LS only).
PHY	Module that implements the physical layer of the USB standard (HS capable).
Attached	Device is physically plugged into USB port, but has *not enabled* either D+ or D– pullup resistor.
Connected	Device is physically plugged into USB port, and has *enabled* either D+ or D– pullup resistor.
Suspended	After 3 ms of no bus activity, the USB device enters suspend mode.
Component	The hardware and software that make up a subsystem.

56.3 Introduction

The USBDCD module works with the USB transceiver to detect whether the USB device is attached to a Charging Port, either a Dedicated Charging Port (DCP) or a Charging Downstream Port (CDP). System software coordinates the detection activites of the module and controls an off-chip integrated circuit that performs the battery charging.

There can be separate instantiations of the USBDCD module for each USB port. The instantiation for the High-Speed capable USB port is named USBHSDCD, so that it can be differentiated from the instantiation for Full-Speed/Low-Speed USB port which retains the name USBDCD. See the chip-specific information for the specific instantiation of your device.

The use of the term "USB transceiver" in this module documentation applies to the USB physical layer instance on the chip, whether it is a FS/LS only transceiver or a HS/FS/LS capable PHY.

56.4 Block diagram

The following figure is a high level block diagram of the module.

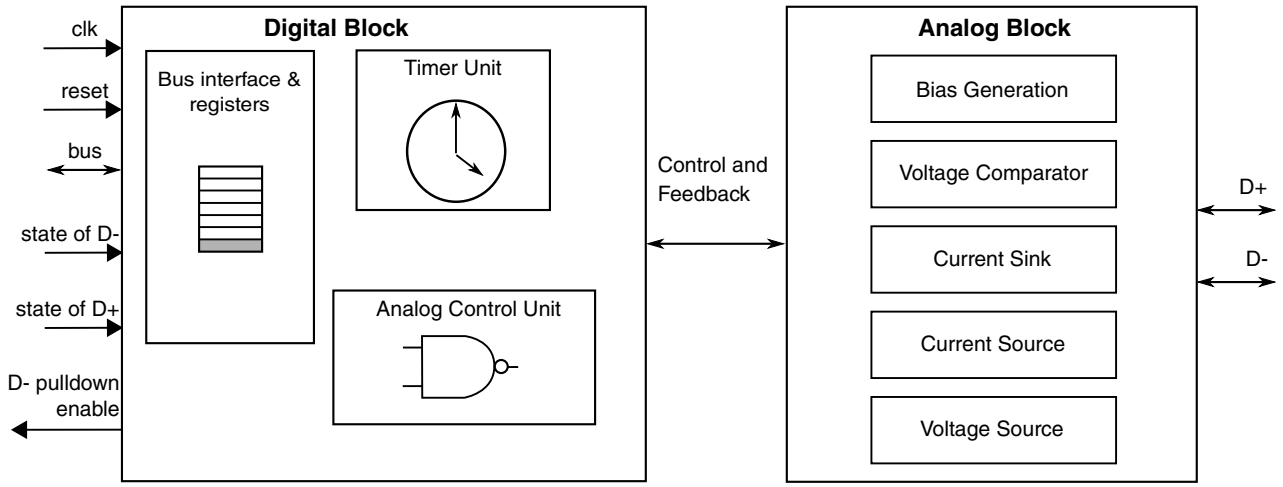


Figure 56-1. Block diagram

The USBDCD module consists of two main blocks:

- A digital block provides the programming interface (memory-mapped registers) and includes the timer unit and the analog control unit.
- An analog block provides the circuitry for the physical detection of the charger, including the bias generation, voltage source, current source, current sink, and voltage comparator circuitry. This block also contains necessary muxes between the source/sink circuits and the D- and D+ pins.

56.5 Features

The USBDCD module offers the following features:

- Compliant with the latest industry standard specification: *USB Battery Charging Specification, Revision 1.1 and 1.2*
- Programmable timing parameters default to values required by the industry standards:
  - Having standard default values allows for easy configuration- simply set the clock frequency before enabling the module.
  - Programmability allows the flexibility to meet future updates of the standards.

56.6 Modes of operation

The operating modes of the USBDCD module are shown in the following table.

Table 56-4. Module modes and their conditions

Module mode	Description	Conditions when used
Enabled	The module performs the charger detection sequence.	System software should enable the module only when *all* of the following conditions are true: • The system uses a rechargeable battery. • The device is being used in an FS USB device application. • The device has detected that it is attached to the USB cable.
Disabled	The module is not active and is held in a low power state.	System software should disable the module when *either* of the following conditions is true: • The charger detect sequence is complete. • The conditions for being enabled are not met.
Powered Off	The digital supply voltage dvdd is removed.	Low system performance requirements allow putting the device into a very low-power stop mode.

Operating mode transitions are shown in the following table.

Table 56-5. Entering and exiting module modes

Module mode	Entering	Exiting	Mode after exiting
Enabled	Set CONTROL[START].	Set CONTROL[SR].[1]	Disabled
Disabled	Take *either* of the following actions: • Set CONTROL[SR].[1] • Reset the module. By default, the module is disabled.	Set CONTROL[START].	Enabled
Powered Off	Perform the following actions: 1. Put the device into very low-power stop mode. 2. Adjust the supply voltages.	Perform the following actions: 1. Restore the supply voltages. 2. Take the device out of very low-power stop mode.	Disabled

1. The effect of setting the SR bit is immediate; that is, the module is disabled even if the sequence has not completed.

56.7 Module signal descriptions

This section describes the module signals. The following table shows a summary of module signals that interface with the pins of the device.

Table 56-6. Signal descriptions

Signal	Description	I/O
USB_DM	USB D– analog data signal. The analog block interfaces directly to the D– signal on the USB bus.	I/O
USB_DP	USB D+ analog data signal. The analog block interfaces directly to the D+ signal on the USB bus.	I/O
avdd33 [1]	3.3 V regulated analog supply	I
vss_bulk	Digital/Analog ground	I
dvdd	Core voltage digital supply	I

1. Voltage must be 3.3 V ± 10% for full functionality of the module. That is, the charger detection function does not work when this voltage is below 3.0 V, and the CONTROL[START] bit should not be set.

NOTE

The transceiver module also interfaces to the USB_DM and USB_DP signals. Both modules and the USB host/hub use these signals as bidirectional, tristate signals.

Information about the signal integrity aspects of the lines including shielding, isolated return paths, input or output impedance, packaging, suggested external components, ESD, and other protections can be found in the USB 2.0 specification and in Application information.

56.8 Memory map/Register definition

This section describes the memory map and registers for the USBHSDCD module.

56.8.1 USBDCD register descriptions

56.8.1.1 USBDCD Memory map

DCD base address: 4035_0800h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control register (CONTROL)	32	RW	0001_0000h
4h	Clock register (CLOCK)	32	RW	0000_00C1h
8h	Status register (STATUS)	32	RO	0000_0000h
Ch	Signal Override Register (SIGNAL_OVERRIDE)	32	RW	0000_0000h
10h	TIMER0 register (TIMER0)	32	RW	0010_0000h
14h	TIMER1 register (TIMER1)	32	RW	000A_0028h
18h	TIMER2_BC11 register (TIMER2_BC11)	32	RW	0028_0001h
18h	TIMER2_BC12 register (TIMER2_BC12)	32	RW	0001_0028h

56.8.1.2 Control register (CONTROL)

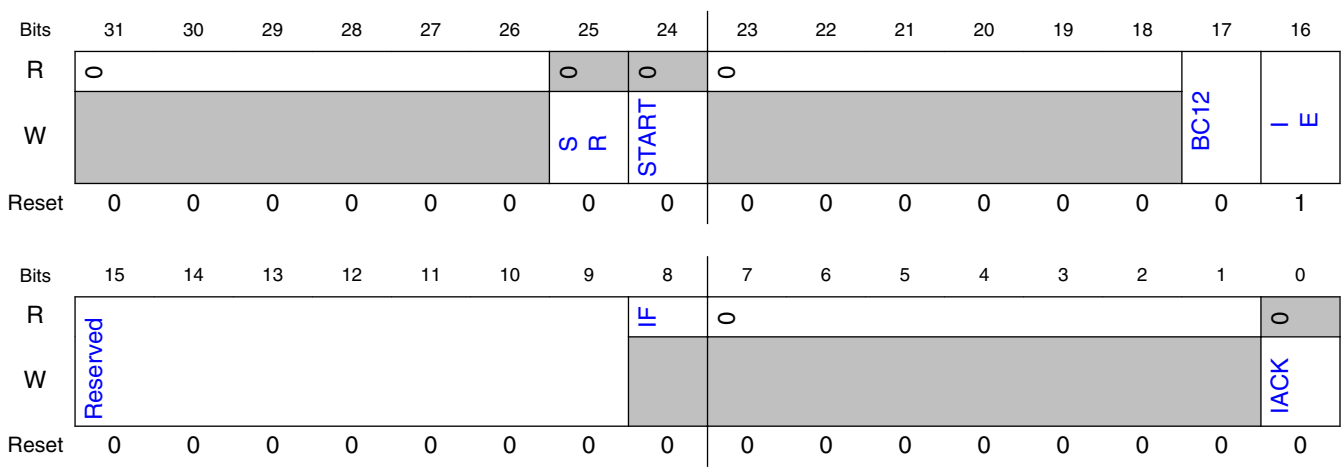
56.8.1.2.1 Offset

Register	Offset
CONTROL	0h

56.8.1.2.2 Function

Contains the control and interrupt bit fields.

56.8.1.2.3 Diagram



56.8.1.2.4 Fields

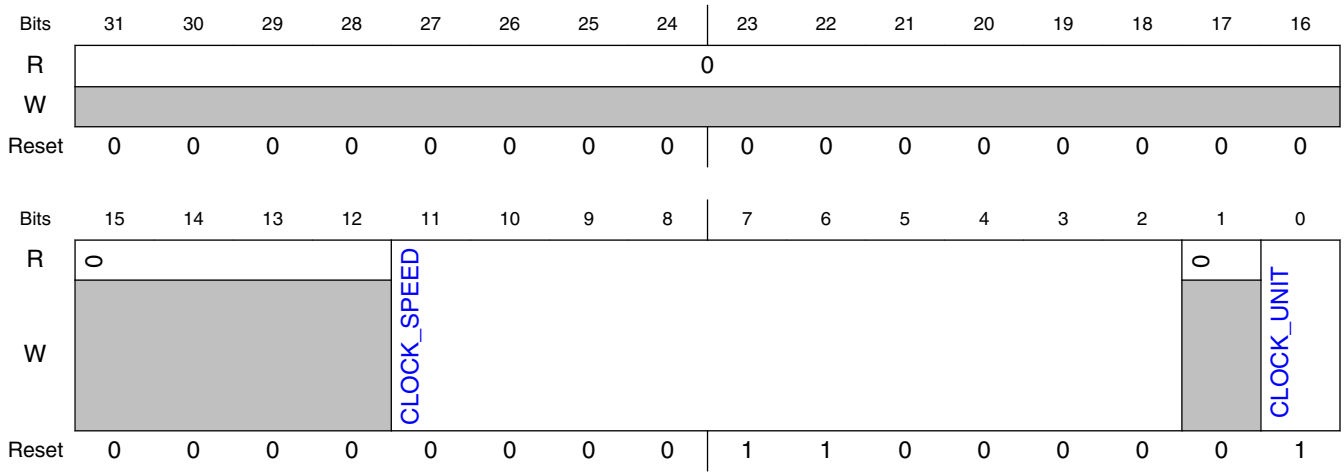
Field	Function
31-26 —	Reserved
25 SR	Software Reset Determines whether a software reset is performed. 0b - Do not perform a software reset. 1b - Perform a software reset.
24 START	Start Change Detection Sequence Determines whether the charger detection sequence is initiated. 0b - Do not start the sequence. Writes of this value have no effect. 1b - Initiate the charger detection sequence. If the sequence is already running, writes of this value have no effect.
23-18 —	Reserved
17 BC12	BC12 BC1.2 compatibility. This bit cannot be changed after start detection. 0b - Compatible with BC1.1 (default) 1b - Compatible with BC1.2
16 IE	Interrupt Enable Enables/disables interrupts to the system. 0b - Disable interrupts to the system. 1b - Enable interrupts to the system.
15-9 —	Reserved
8 IF	Interrupt Flag Determines whether an interrupt is pending. 0b - No interrupt is pending. 1b - An interrupt is pending.
7-1 —	Reserved
0 IACK	Interrupt Acknowledge Determines whether the interrupt is cleared. 0b - Do not clear the interrupt. 1b - Clear the IF bit (interrupt flag).

56.8.1.3 Clock register (CLOCK)

56.8.1.3.1 Offset

Register	Offset
CLOCK	4h

56.8.1.3.2 Diagram



56.8.1.3.3 Fields

Field	Function
31-12 —	Reserved
11-2 CLOCK_SPEED	Numerical Value of Clock Speed in Binary The unit of measure is programmed in CLOCK_UNIT. The valid range is from 1 to 1023 when clock unit is MHz and 4 to 1023 when clock unit is kHz. Examples with CLOCK_UNIT = 1: • For 48 MHz: 0b00_0011_0000 (48) (Default) • For 24 MHz: 0b00_0001_1000 (24) Examples with CLOCK_UNIT = 0: • For 100 kHz: 0b00_0110_0100 (100) • For 500 kHz: 0b01_1111_0100 (500)
1 —	Reserved
0 CLOCK_UNIT	Unit of Measurement Encoding for Clock Speed Specifies the unit of measure for the clock speed. 0b - kHz Speed (between 1 kHz and 1023 kHz) 1b - MHz Speed (between 1 MHz and 1023 MHz)

56.8.1.4 Status register (STATUS)

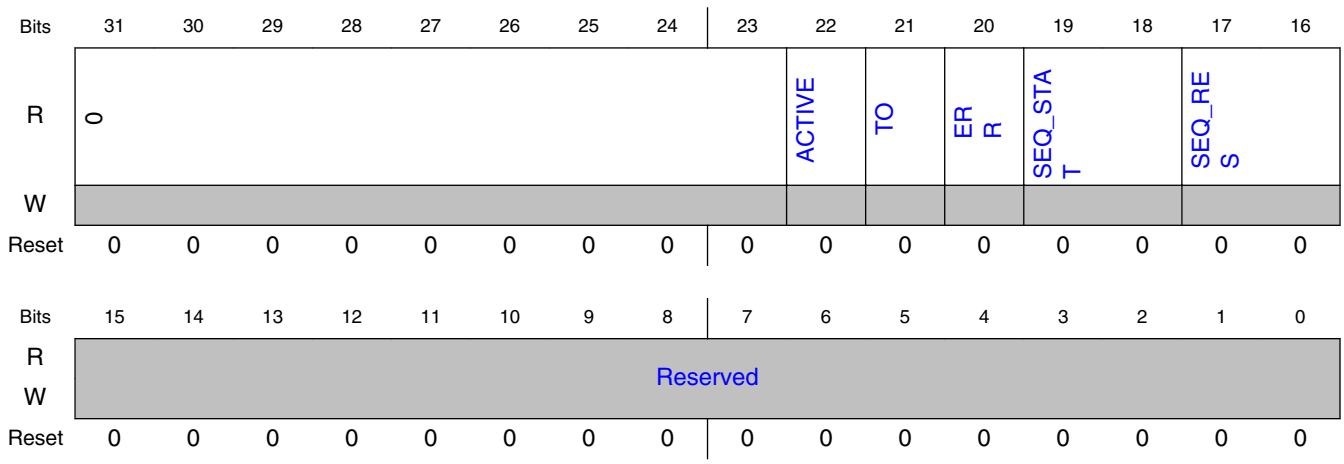
56.8.1.4.1 Offset

Register	Offset
STATUS	8h

56.8.1.4.2 Function

Provides the current state of the module for system software monitoring.

56.8.1.4.3 Diagram



56.8.1.4.4 Fields

Field	Function
31-23 —	Reserved
22 ACTIVE	Active Status Indicator Indicates whether the sequence is running. 0b - The sequence is not running. 1b - The sequence is running.
21 TO	Timeout Flag Indicates whether the detection sequence has passed the timeout threshhold. 0b - The detection sequence has not been running for over 1 s. 1b - It has been over 1 s since the data pin contact was detected and debounced.
20 ERR	Error Flag Indicates whether there is an error in the detection sequence. 0b - No sequence errors. 1b - Error in the detection sequence. See the SEQ_STAT field to determine the phase in which the error occurred.

Table continues on the next page...

Field	Function
19-18 SEQ_STAT	Charger Detection Sequence Status Indicates the status of the charger detection sequence. 00b - The module is either not enabled, or the module is enabled but the data pins have not yet been detected. 01b - Data pin contact detection is complete. 10b - Charging port detection is complete. 11b - Charger type detection is complete.
17-16 SEQ_RES	Charger Detection Sequence Results Reports how the charger detection is attached. 00b - No results to report. 01b - Attached to an SDP. Must comply with USB 2.0 by drawing only 2.5 mA (max) until connected. 10b - Attached to a charging port. The exact meaning depends on bit 18 (value 0: Attached to either a CDP or a DCP. The charger type detection has not completed. value 1: Attached to a CDP. The charger type detection has completed.) 11b - Attached to a DCP.
15-0 —	Reserved **NOTE:** Bits do not always read as 0.

56.8.1.5 Signal Override Register (SIGNAL_OVERRIDE)

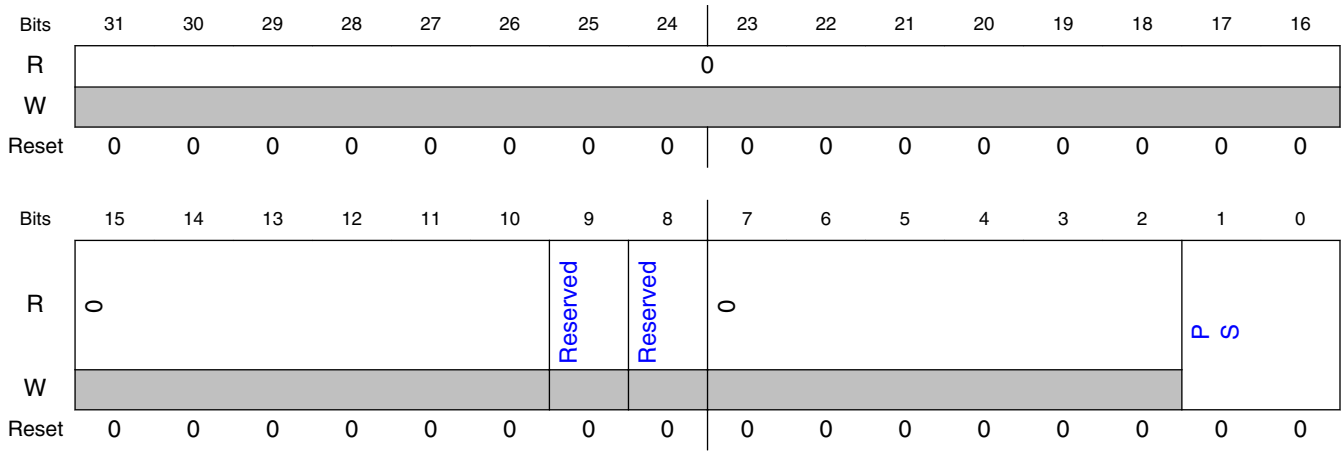
56.8.1.5.1 Offset

Register	Offset
SIGNAL_OVERRIDE	Ch

56.8.1.5.2 Function

The Signal Override register provides a way for the customer to enable signaling required by the USB BC v1.2 specification after the battery charger detection sequences have completed.

56.8.1.5.3 Diagram



56.8.1.5.4 Fields

Field	Function
31-10 —	Reserved
9 —	Reserved Reserved, not for customer use. The value of this bit field may change during module operation.
8 —	Reserved Reserved, not for customer use. The value of this bit field may change during module operation.
7-2 —	Reserved
1-0 PS	Phase Selection Used to enable specified voltage and current source circuits on the USB_DP and USB_DM pins. Customers may set this bit field to 2'b10 for required signaling if attached to a Dedicated Charging Port, or during operation under the Dead Battery Provision. 00b - No overrides. Bit field must remain at this value during normal USB data communication to prevent unexpected conditions on USB_DP and USB_DM pins. (Default) 01b - Reserved, not for customer use. 10b - Enables VDP_SRC voltage source for the USB_DP pin and IDM_SINK current source for the USB_DM pin. 11b - Reserved, not for customer use.

56.8.1.6 TIMER0 register (TIMER0)

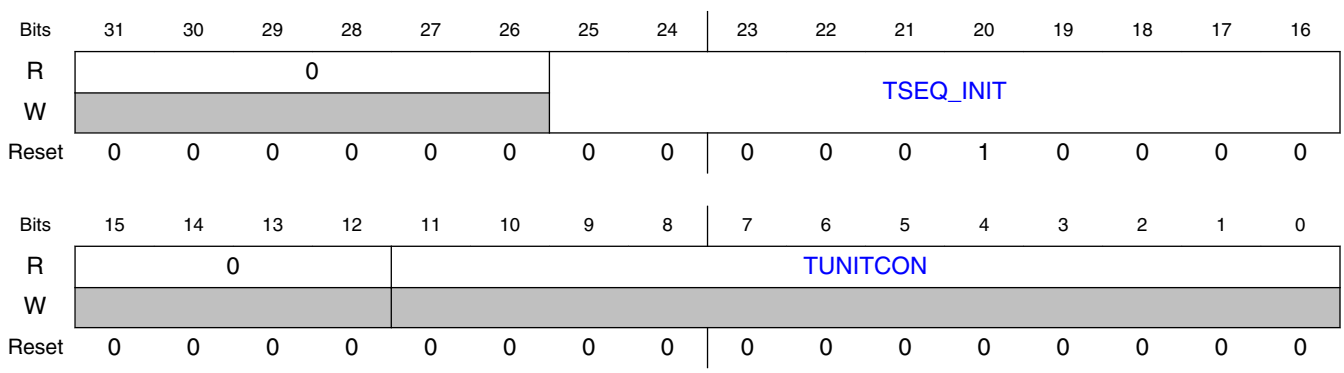
56.8.1.6.1 Offset

Register	Offset
TIMER0	10h

56.8.1.6.2 Function

TIMER0 has an TSEQ_INIT field that represents the system latency in ms. Latency is measured from the time when VBUS goes active until the time system software initiates charger detection sequence in USBDCD module. When software sets the CONTROL[START] bit, the Unit Connection Timer (TUNITCON) is initialized with the value of TSEQ_INIT. Valid values are 0-1023, however the USB Battery Charging Specification requires the entire sequence, including TSEQ_INIT, to be completed in 1s or less.

56.8.1.6.3 Diagram



56.8.1.6.4 Fields

Field	Function
31-26 —	Reserved
25-16 TSEQ_INIT	Sequence Initiation Time TSEQ_INIT represents the system latency (in ms) measured from the time VBUS goes active to the time system software initiates the charger detection sequence in the USBDCD module. When software sets the CONTROL[START] bit, the Unit Connection Timer (TUNITCON) is initialized with the value of TSEQ_INIT. Valid values are 0-1023, but the USB Battery Charging Specification requires the entire sequence, including TSEQ_INIT, to be completed in 1s or less.
15-12 —	Reserved
11-0	Unit Connection Timer Elapse (in ms)

Field	Function
TUNITCON	Displays the amount of elapsed time since the event of setting the START bit plus the value of TSEQ_INIT. The timer is automatically initialized with the value of TSEQ_INIT before starting to count. This timer enables compliance with the maximum time allowed to connect T_{UNIT_CON} under the USB Battery Charging Specification. If the timer reaches the one second limit, the module triggers an interrupt and sets the error flag STATUS[ERR]. The timer continues counting throughout the charger detection sequence, even when control has been passed to software. As long as the module is active, the timer continues to count until it reaches the maximum value of 0xFFF (4095 ms). The timer does not rollover to zero. A software reset clears the timer.

56.8.1.7 TIMER1 register (TIMER1)

56.8.1.7.1 Offset

Register	Offset
TIMER1	14h

56.8.1.7.2 Function

TIMER1 contains timing parameters. Note that register values can be written that are not compliant with the USB Battery Charging Specification, so care should be taken when overwriting the default values.

56.8.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						TDCD_DBNC									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						TVDPSRC_ON									
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

56.8.1.7.4 Fields

Field	Function
31-26 —	Reserved
25-16 TDCD_DBNC	Time Period to Debounce D+ Signal Sets the time period (ms) to debounce the D+ signal during the data pin contact detection phase. See "Debouncing the data pin contact" Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 10 ms.
15-10 —	Reserved
9-0 TVDPSRC_ON	Time Period Comparator Enabled This timing parameter is used after detection of the data pin. See "Charging Port Detection". Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 40 ms.

56.8.1.8 TIMER2_BC11 register (TIMER2_BC11)

56.8.1.8.1 Offset

Register	Offset
TIMER2_BC11	18h

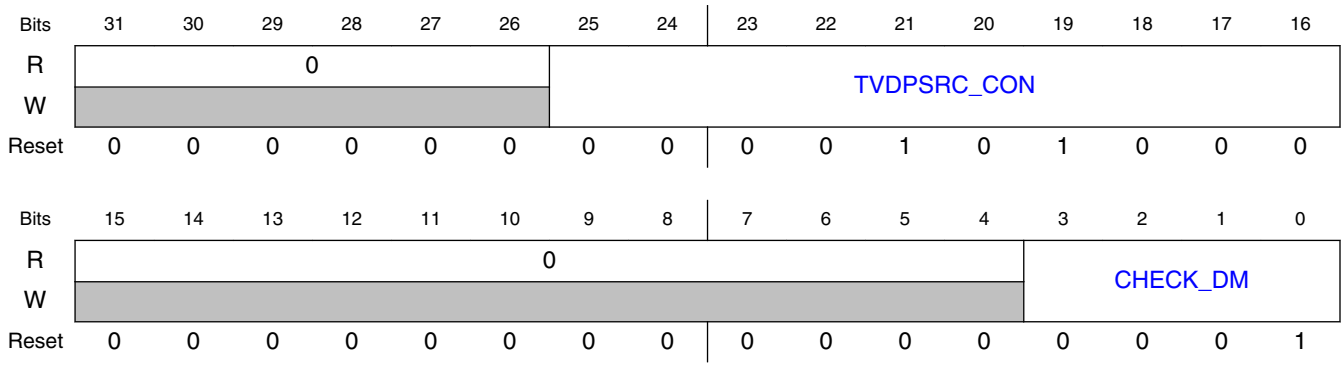
56.8.1.8.2 Function

TIMER2_BC11 contains timing parameters for *USB Battery Charging Specification, Rev 1.1*.

NOTE

Register values can be written that are not compliant with the USB Battery Charging Specification, so care should be taken when overwriting the default values.

56.8.1.8.3 Diagram



56.8.1.8.4 Fields

Field	Function
31-26 —	Reserved
25-16 TVDPSRC_CON	Time Period Before Enabling D+ Pullup Sets the time period (ms) that the module waits after charging port detection before system software must enable the D+ pullup to connect to the USB host. Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 40 ms.
15-4 —	Reserved
3-0 CHECK_DM	Time Before Check of D- Line Sets the amount of time (in ms) that the module waits after the device connects to the USB bus until checking the state of the D- line to determine the type of charging port. See "Charger Type Detection." Valid values are 1-15ms.

56.8.1.9 TIMER2_BC12 register (TIMER2_BC12)

56.8.1.9.1 Offset

Register	Offset
TIMER2_BC12	18h

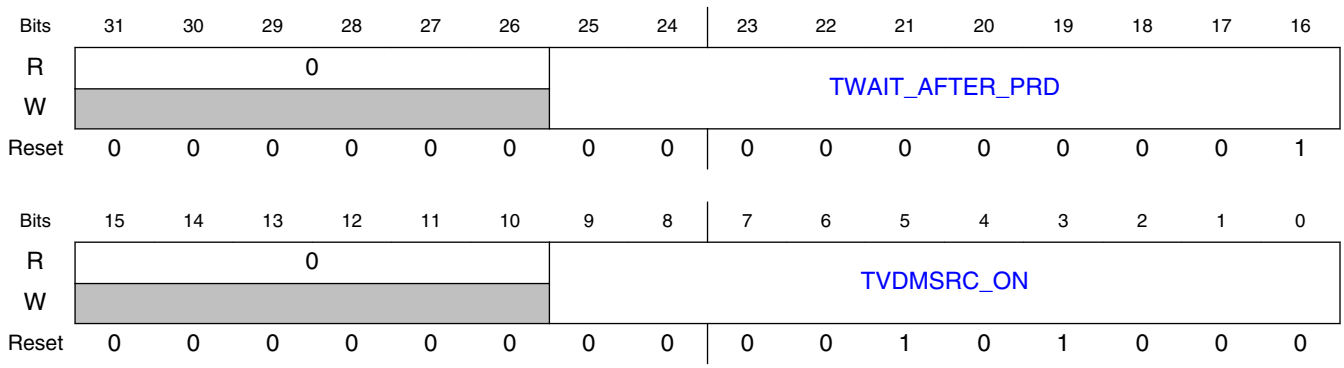
56.8.1.9.2 Function

TIMER2_BC12 contains timing parameters for *USB Battery Charging Specification, Rev 1.2*.

NOTE

Register values can be written that are not compliant with the USB Battery Charging Specification, so care should be taken when overwriting the default values.

56.8.1.9.3 Diagram



56.8.1.9.4 Fields

Field	Function
31-26 —	Reserved
25-16 TWAIT_AFTER_PRD	TWAIT_AFTER_PRD Sets the amount of time (in ms) that the module waits after primary detection before start to secondary detection. Valid values are 1-1023ms. Default is 1ms.
15-10 —	Reserved
9-0 TVDMSRC_ON	TVDMSRC_ON Sets the amount of time (in ms) that the module enables the V_{DM_SRC} . Valid values are 0-40ms.

56.9 Functional description

The sequence of detecting the presence of charging port and type of charging port involves several hardware components, coordinated by system software. This collection of interacting hardware and software is called the USB Battery Charging Subsystem. The following figure shows the USBDCD module as a component of the subsystem. The following table describes the components.

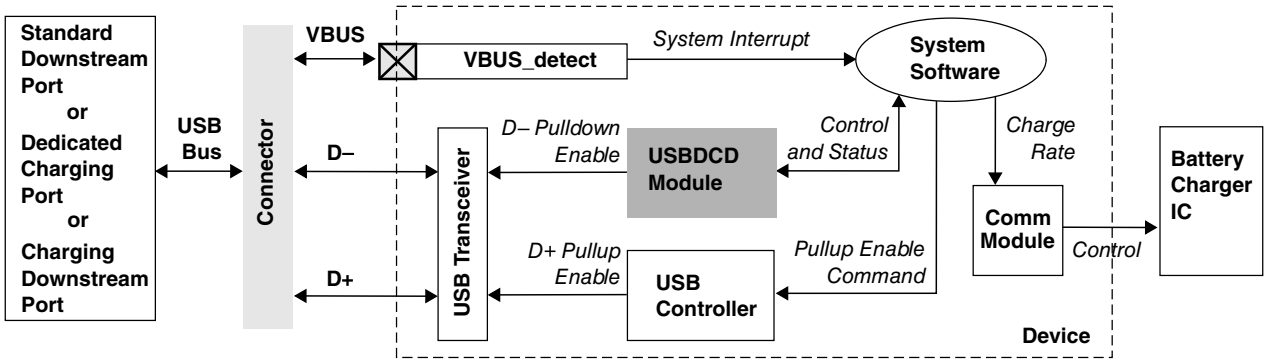


Figure 56-2. USB battery charging subsystem

Table 56-7. USB battery charger subsystem components

Component	Description
Battery Charger IC	The external battery charger IC regulates the charge rate to the rechargable battery. System software is responsible for communicating the appropriate charge rates. **Charger** / **Maximum current drawn** Standard Downstream Port (SDP) / up to 500 mA (I_{CFG_MAX}) Charging Downstream Port (CDP) / up to 1500 mA (I_{DEV_CHG}) Dedicated Charging Port (DCP) / up to 1500 mA (I_{DEV_CHG}) 1. If the USB host has suspended the USB device, system software must configure the system to limit the current drawn from the USB bus to 2.5 mA or less.
Comm Module	A communications module on the device can be used to control the charge rate of the battery charger IC.
System software	Coordinates the detection activities of the subsystem.
USB Controller	The D+ pullup enable control signal plays a role during the charger type detection phase. System software must issue a command to the USB controller to assert this signal. After this pullup is enabled, the device is considered to be connected to the USB bus. The host then attempts to enumerate it. **NOTE:** The USB controller must be used only for USB device applications when using the USBDCD module. For USB host applications, the USBDCD module must be disabled.
USB Transceiver	The USB transceiver contains the pullup resistor for the USB D+ signal and the pulldown resistors for the USB D+ and D– signals. The D+ pullup and the D– pulldown are both used during the charger detection sequence in BC1.1, but it is not used during charger detection in BC1.2. The USB transceiver also outputs the digital state of the D+ and D– signals from the USB bus. The pullup and pulldown enable signals are controlled by other modules during the charger detection sequence in BC1.1: The D+ pullup enable is output from the USB controller and is under software control. The USBDCD module controls the D– pulldown enable.
USBDCD Module	Detects whether the device has been plugged into either an SDP, a CDP, or a DCP.
VBUS_detect	This interrupt pin connected to the USB VBUS signal detects when the device has been plugged into or unplugged from the USB bus. If the system requires waking up from a low power mode on being plugged into the USB port, this interrupt should also be a low power wake up source. If this pin multiplexes other functions, such as GPIO, the pin can be configured as an interrupt so that the USB plug or unplug event can be detected.

1. If the USB host has suspended the USB device, system software must configure the system to limit the current drawn from the USB bus to 2.5 mA or less.

56.9.1 The charger detection sequence

The following figure illustrates the charger detection sequence in a simplified timing diagram based on the *USB Battery Charging Specification, Rev. 1.2*.

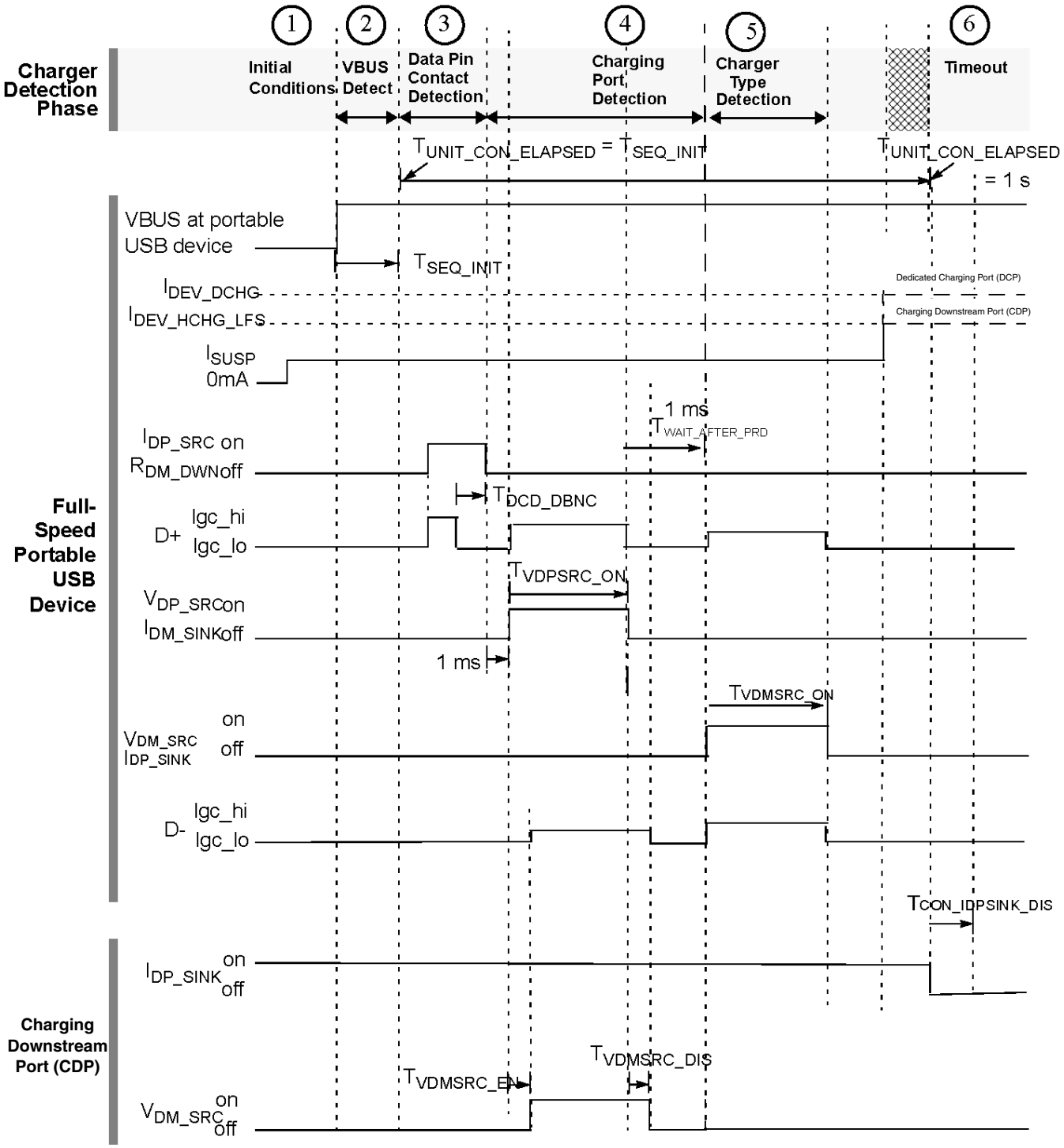


Figure 56-3. Full speed charger detection timing for BC1.2

Timing parameter values used in this module for BC1.2 are listed in the following table.

Table 56-8. Timing parameters for the charger detection sequence for BC1.2

Parameter	USB Battery Charging Spec	Module default	Module programmable range
$T_{DCD\_DBNC'}$ [1]	10 ms min (no max)	10 ms	0– 1023 ms
T_{VDPSRC_ON} [1]	40 ms min (no max)	40 ms	0 –1023 ms
$T_{WAIT\_AFTER\_PRD}$	N/A	1 ms	0– 1023 ms
T_{VDMSRC_ON}	40 ms	40 ms	0 –1023 ms
T_{SEQ_INIT}	N/A	16 ms	0 –1023 ms
T_{UNIT_CON} [1]	1 s	N/A	N/A
T_{VDMSRC_EN} [1]	1– 20 ms	From the USB host	N/A
T_{VDMSRC_DIS} [1]	0 – 20 ms	From the USB host	N/A
$T_{CON\_IDPSINK\_DIS}$ [1]	0 – 10 ms	From the USB host	N/A

1. This parameter is defined by the *USB Battery Charging Specification, Rev. 1.2*.

The following figure illustrates the charger detection sequence in a simplified timing diagram based on the *USB Battery Charging Specification, Rev. 1.1*.

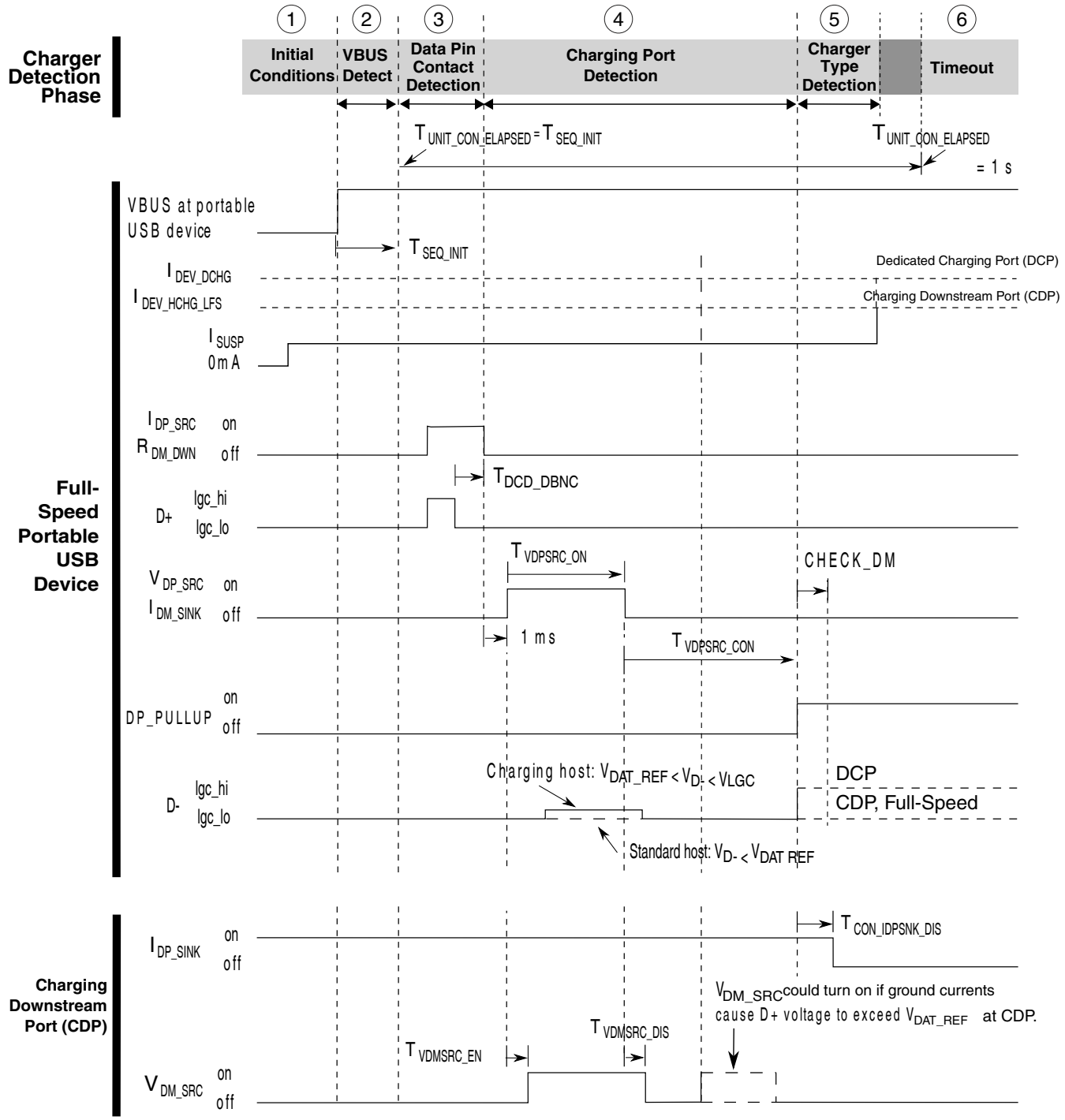


Figure 56-4. Full speed charger detection timing for BC1.1

Timing parameter values used in this module for BC1.1 are listed in the following table.

Table 56-9. Timing parameters for the charger detection sequence for BC1.1

Parameter	USB Battery Charging Spec	Module default	Module programmable range
$T_{DCD\_DBNC'}$ [1]	10 ms min (no max)	10 ms	0– 1023 ms
T_{VDPSRC_ON} [1]	40 ms min (no max)	40 ms	0 –1023 ms
T_{VDPSRC_CON} [1]	40 ms min (no max)	40 ms	0 –1023 ms
CHECK_DM	N/A	1 ms	0– 15 ms
T_{SEQ_INIT}	N/A	16 ms	0 –1023 ms
T_{UNIT_CON} [1]	1 s	N/A	N/A
T_{VDMSRC_EN} [1]	1– 20 ms	From the USB host	N/A
T_{VDMSRC_DIS} [1]	0 –20 ms	From the USB host	N/A
$T_{CON\_IDPSINK\_DIS}$ [1]	0– 20 ms	From the USB host	N/A

1. This parameter is defined by the *USB Battery Charging Specification, Rev. 1.1*.

The following table provides an overview description of the charger detection sequence shown in the preceding figure.

Table 56-10. Overview of the charger detection sequence

Phase		Overview description	Full description
1	Initial Conditions	Initial system conditions that need to be met before the detection sequence is initiated.	Initial System Conditions
2	VBUS Detection	System software detects contact of the VBUS signal with the system interrupt pin VBUS_detect.	VBUS contact detection
3	Data Pin Contact Detection	The USBDCD module detects that the USB data pins D+ and D– have made contact with the USB port.	Data pin contact detection
4	Charging Port Detection	The USBDCD module detects if the port is an SDP or either type of charging port, that is CDP or DCP.	Charging port detection
5	Charger Type Detection	The USBDCD module detects the type of charging port, if applicable.	Charger type detection
6	Sequence Timeout	The USBDCD module did not finish the detection sequence within the timeout interval. The sequence will continue until halted by software.	Charger detection sequence timeout

56.9.1.1 Initial System Conditions

The USBDCD module is intended for use with USB device applications using a rechargable battery. The module does not have support for interfacing with ACA or ACA-Dock equipment as defined in the *USB Battery Charging Specification, Revision 1.2*. It cannot be used with USB applications that are embedded host or OTG.

In addition, before the USBDCD module's charger detection sequence can be initiated, the system must be:

- Powered-up and in run mode. The USBDCD instantiation of this module for the High-Speed port does not directly depend on the VBUS voltage and can operate as long as the avdd33 supply is in a valid range.
- Recently plugged into a USB port.
- Drawing not more than 2.5 mA total system current from the USB bus, except as allowed by the *USB 2.0 Connect Timing Update ECN*.

Examples of allowable precursors to this set of initial conditions include:

- A powered-down device is subsequently powered-up upon being plugged into the USB bus.
- A device in a low power mode subsequently enters run mode upon being plugged into the USB bus.

56.9.1.2 VBUS contact detection

Once the device is plugged into a USB port, the VBUS_detect system interrupt is triggered. System software must do the following to initialize the module and start the charger detection sequence:

1. Restore power if the module is powered-off.
2. Set CONTROL[SR] to initiate a software reset.
3. Configure the USBDCD module by programming the CLOCK register and the timing parameters as needed.
4. Set CONTROL[IE] to enable interrupts, or clear the bit if software polling method is used.
5. Program CONTROL[BC12] based on which revision of the USB Battery Charging Specification is needed.
6. Set CONTROL[START] to start the charger detection sequence.

56.9.1.3 Data pin contact detection

The module must ensure that the data pins have made contact because the detection sequence depends upon the state of the USB D+ and D- signals. USB plugs and receptables are designed such that when the plug is inserted into the receptable, the power pins make contact before the data pins make contact. See the following figure.

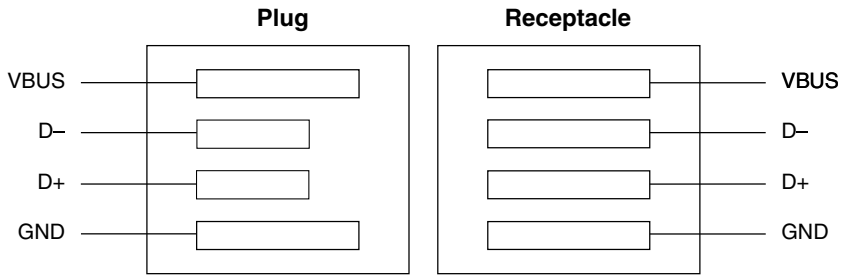


Figure 56-5. Relative pin positions in USB plugs and receptacles

As a result, when a portable USB device is attached to an upstream port, the portable USB device detects VBUS before the data pins have made contact. The time between power pins and data pins making contact depends on how fast the plug is inserted into the receptable. Delays of several hundred milliseconds are possible.

56.9.1.3.1 Debouncing the data pin contact

When system software has initiated the charger detection sequence, as described in Initial System Conditions, the USBDCD module turns on the I_{DP_SRC} current source and enables the R_{DM_DWN} pulldown resistor. If the data pins have not made contact, the D+ line remains high. After the data pins make contact, the D+ line goes low and debouncing begins.

After the D+ line goes low, the module continuously samples the D+ line over the duration of the T_{DCD_DBNC} debounce time interval.By deafult, T_{DCD_DBNC} is 10 ms, but it can be programmed in the TIMER0[TDCD_DBNC] field. See the description of the TIMER0 Register for register information.

When it has remained low for the entire interval, the debouncing is complete. However, if the D+ line returns high during the debounce interval, the module waits until the D+ line goes low again to restart the debouncing. This cycle repeats until either of the following happens:

- The data pin contact has been successfully debounced (see Success in detecting data pin contact (phase completion)).
- A timeout occurs (see Charger detection sequence timeout).

56.9.1.3.2 Success in detecting data pin contact (phase completion)

After successfully debouncing the D+ state, the module does the following:

- Updates the STATUS register to reflect phase completion (See Table 56-14 for field values.)
- Directly proceeds to the next step in the sequence: detection of a charging port (See Charging port detection.)

56.9.1.4 Charging port detection

After it detects that the data pins have made contact, the module waits for a fixed delay of 1 ms, and then attempts to detect whether it is plugged into a charging port. The module connects the following analog units to the USB D+ or D– lines during this phase:

- The voltage source V_{DP_SRC} connects to the D+ line
- The current sink I_{DM_SINK} connects to the D– line
- The voltage comparator connects to the USB D– line, comparing it to the voltage V_{DAT_REF}.

After a time of T_{VDPSRC_ON}, the module samples the D– line. The T_{VDPSRC_ON} parameter is programmable and defaults to 40 ms. After sampling the D– line, the module disconnects the voltage source, current sink, and comparator.

The next steps in the sequence depend on the voltage on the D– line as determined by the voltage comparator. See the following table.

Table 56-11. Sampling D– in the charging port detection phase

If the voltage on D- is...	Then...	See...
Below V_{DAT_REF}	The port is an SDP that does not support using charging currents above I_{CFG_MAX}.	Standard downstream port
Above V_{DAT_REF} but below V_{LGC}	The port is a charging port.	Charging port
Above V_{LGC}	This is an error condition.	Error in charging port detection

56.9.1.4.1 Standard downstream port

As part of the charger detection handshake with a standard USB host, the module does the following without waiting for the interval ($T_{WAIT\_AFTER\_PRD}$ or T_{VDPSRC_CON}) to elapse:

- Updates the STATUS register to reflect that a standard downstream port (SDP) has been detected with SEQ_RES = 01. See Table 56-14 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].

At this point, control has been passed to system software via the interrupt. The rest of the sequence, which detects the type of charging port, is not applicable, so software should perform the following steps:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Set CONTROL[SR] to issue a software reset to the module.
4. Disable the module.
5. Communicate the appropriate charge rate to the external battery charger IC; see Table 56-7.

56.9.1.4.2 Charging port

As part of the charger detection handshake with any type of USB host, the module waits until the interval ($T_{WAIT\_AFTER\_PRD}$ or T_{VDPSRC_CON}) has elapsed before it does the following:

- Enables V_{DM_SRC} (for *USB Battery Charging Specification, Rev1.2* only).
- Updates the STATUS register to reflect that a charging port has been detected with SEQ_RES = 10. See Table 56-14 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].

At this point, control has passed to system software via the interrupt. Software should:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Issue a command to the USB controller to pullup the USB D+ line.
4. Wait for the module to complete the final phase of the sequence. See Charger type detection.

56.9.1.4.3 Error in charging port detection

For this error condition, the module does the following:

- Updates the STATUS register to reflect the error with SEQ_RES = 00. See Table 56-14 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].

Note that in this case the module does not wait for the interval ($T_{WAIT\_AFTER\_PRD}$ or T_{VDPSRC_CON}) to elapse.

At this point, control has been passed to system software via the interrupt. The rest of the sequence (detecting the type of charging port) is not applicable, so software should:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Set CONTROL[SR] to issue a software reset to the module.
4. Disable the module.

56.9.1.5 Charger type detection

For *USB Battery Charging Specification, Rev. 1.2*:

After the USBDCD module enables the V_{DM_SRC}, the module starts the T_{VDMSRC_ON} timer counting down the time interval programmed into the TIMER2[T_{VDMSRC_ON}] field.

Once the T_{VDMSRC_ON} timer has elapsed, the module samples the USB D+ line to determine the type of charger. See the following table.

Table 56-12. Sampling D+ in the charger type detection phase (BC1.2)

If the voltage on D+ is...	Then...	See...
High (D+>V_{DAT_REF})	The port is a DCP. [1]	Dedicated charging port
Low (D+<V_{DAT_REF})	The port is a CDP. [2]	Charging downstream port

1. In a DCP, the D+ and D– lines are shorted together through a small resistor.
2. In a CDP, the D+ and D– lines are not shorted.

For *USB Battery Charging Specification, Rev. 1.1*:

After software enables the D+ pullup resistor, the module is notified automatically (via internal signaling) to start the CHECK_DM timer counting down the time interval programmed in the TIMER2[CHECK_DM] field.

After the CHECK_DM time has elapsed, the module samples the USB D– line to determine the type of charger. See the following table.

Table 56-13. Sampling D– in the charger type detection phase (BC1.1)

If the voltage on D– is...	Then...	See...
High	The port is a DCP. [1]	Dedicated charging port
Low	The port is a CDP. [2]	Charging downstream port

1. In a DCP, the D+ and D– lines are shorted together through a small resistor.
2. In a CDP, the D+ and D– lines are not shorted.

56.9.1.5.1 Dedicated charging port

For a dedicated charging port (DCP), the module does the following:

- Updates the STATUS register to reflect that a dedicated charging port has been detected with SEQ_RES = 11. See Table 56-14 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE] bit.

The *USB Battery Charging Specification, Rev. 1.2* indicates that additionally if the detection sequence determines an attachment to a DCP, then the USB device should signal on the D+ pin by either enabling the D+ pullup resistor or enabling V_{DP_SRC}.

At this point, control has been passed to system software via the interrupt. Software should:

1. Read the STATUS register.
2. Disable the USB controller to prevent transitions on the USB D+ or D– lines from causing spurious interrupt or wakeup events to the system.
3. Set CONTROL[IACK] to acknowledge the interrupt.
4. Set CONTROL[SR] to issue a software reset to the module.
5. Disable the module.

6. Enable signaling on the D+ pin. This can be done by enabling the Transceiver D+ pullup resistor through configuration of register bit fields in the USB controller instance for this USB port. On this product this signaling can alternatively be done by enabling V_{DP_SRC} through setting the SIGNAL_OVERRIDE[PS] bit field to a value of 2'b10.
7. Communicate the appropriate charge rate to the external battery charger IC; see Table 56-7.

56.9.1.5.2 Charging downstream port

For a charging downstream port (CDP), the module does the following:

- Updates the STATUS register to reflect that a CDP has been detected with SEQ_RES = 10. See Table 56-14 for field values.
- Sets CONTROL[IF].
- Generates an interrupt if enabled in CONTROL[IE].

At this point, control has been passed to system software via the interrupt. Software should:

1. Read the STATUS register.
2. Set CONTROL[IACK] to acknowledge the interrupt.
3. Set CONTROL[SR] to issue a software reset to the module.
4. Disable the module.
5. Communicate the appropriate charge rate to the external battery charger IC; see Table 56-7.

56.9.1.6 Charger detection sequence timeout

The maximum time allowed to connect according to the *USB Battery Charging Specification* is one second. If the Unit Connection Timer reaches the one second limit and the sequence is still running as indicated by the STATUS[ACTIVE] bit, the module does the following:

- Updates the STATUS register to reflect that a timeout error has occured. See Table 56-14 for field values.

- Sets the CONTROL[IF] bit.
- Generates an interrupt if enabled in CONTROL[IE].
- The detection sequence continues until explicitly halted by software setting the CONTROL[SR] bit.
- The Unit Connection Timer continues counting. See the description of the TIMER0 Register.

At this point, control has been passed to system software via the interrupt, which has two options: ignore the interrupt and allow more time for the sequence to complete, or halt the sequence. To halt the sequence, software should:

1. Read the STATUS register.
2. Set the CONTROL[IACK] bit to acknowledge the interrupt.
3. Set the CONTROL[SR] bit to issue a software reset to the module.
4. Disable the module.

This timeout function is also useful in case software does not realize that the USB device is unplugged from USB port during the charger detection sequence. If the interrupt occurs but the V_{BUS_DETECT} input is low, software can disable and reset the module.

System software might allow the sequence to run past the timeout interrupt under these conditions:

1. The USB Battery Charging Spec is amended to allow more time. In this case, software should poll TIMER0[$T_{UNITCON}$] periodically to track elapsed time after 1s; or
2. For debug purposes.

Note that the $T_{UNITCON}$ register field will stop incrementing when it reaches its maximum value so it will not rollover to zero and start counting up again.

56.9.1.7 Dead Battery Provision signaling

If the system needs to operate under the Dead Battery Provision of the *USB Battery Charging Specification, Revision 1.2*, the USBDCD module must be configured to signal V_{DP_SRC} on the USB_DP pin.

For information on Dead Battery Provision conditions and signaling, see the Dead or weak battery later in this chapter.

56.9.2 Interrupts and events

The USBDCD module has an interrupt to alert system software of certain events, which are listed in the following table. All events except the Phase Complete event for the Data Pin Detection phase can trigger an interrupt.

Table 56-14. Events triggering an interrupt by sequence phase

Sequence phase	Event	Event description	STATUS fields [1]	Phase description
Data Pin Detection	Phase Complete	The module has detected data pin contact. *No interrupt occurs*: CONTROL[IF] = 0.	ERR = 0 SEQ_STAT = 01 SEQ_RES = 00 TO = 0	VBUS contact detection
Charging Port Detection	Phase Complete	The module has completed the process of identifying if the USB port is a charging port or not. **NOTE:** This only applies to *USB Battery Charging Specification, Rev. 1.1*.	ERR = 0 SEQ_STAT = 10 SEQ_RES = 01 or 10 TO = 0	Charging port detection
	Error	The module cannot identify the type of port because the D– line is above the USB's VLGC threshold.	ERR = 1 SEQ_STAT = 10 SEQ_RES = 00 TO = 0	Error in charging port detection
Charger Type Detection	Phase Complete	The module has completed the process of identifying the charger type detection. **Note:** The ERR flag always reads zero because no known error conditions are checked during this phase.	ERR = 0 SEQ_STAT = 11 SEQ_RES = 11 or 10 TO = 0	Charger type detection
Sequence Timeout	Error	The timeout interval from the time the USB device attaches to a USB port until it connects has elapsed.	ERR = 1 SEQ_STAT = last value [2] SEQ_RES = last value [2] TO = 1	Charger detection sequence timeout.

1. See the description of the Status register for register information.
2. The SEQ_STAT and SEQ_RES fields retain the values held at the time of the timeout error.

56.9.2.1 Interrupt Handling

Software can read which event caused the interrupt from the STATUS register during the interrupt service routine.

An interrupt is generated only if CONTROL[IE] is set. The CONTROL[IF] bit is always set under interrupt conditions, even if CONTROL[IE] is cleared. In this case, software can poll CONTROL[IF] to determine if an interrupt condition is pending.

Writes to CONTROL[IF] are ignored. To reset CONTROL[IF], set CONTROL[IACK] to acknowledge the interrupt. Writing to CONTROL[IACK] when CONTROL[IF] is cleared has no effect.

56.9.3 Resets

There are two ways to reset various register contents in this module: hardware resets and a software reset.

56.9.3.1 Hardware resets

Hardware resets originate at the system or device level and propagate down to the individual module level. They include start up reset, low-voltage reset, and all other hardware reset sources.

Hardware resets cause the register contents to be restored to their default state as listed in the register descriptions.

56.9.3.2 Software reset

A software reset re-initializes the module's status information, but leaves configuration information unchanged. The software reset allows software to prepare the module without needing to reprogram the same configuration each time the USB device is plugged into a USB port.

Setting CONTROL[SR] initiates a software reset. The following table shows all register fields that are reset to their default values by a software reset.

Table 56-15. Software reset and register fields affected

Register	Fields affected	Fields not affected
CONTROL[1]	IF	IE, START
STATUS	All	None
CLOCK	None	All
TIMER*n*	TUNITCON	All other

1. CONTROL[SR] and CONTROL[IACK] are self-clearing.

A software reset also returns all internal logic, timers, and counters to their reset states. If the module is already active (STATUS[ACTIVE] = 1), a software reset stops the sequence.

Note

Software must always initiate a software reset before starting the sequence to ensure the module is in a known state.

56.10 Initialization information

This module has been designed for minimal configuration while retaining significant programmability. The CLOCK register needs to be initialized to the actual system clock frequency, unless the default value already matches the system requirements.

The proper operation of the USBHSDCD instantiation of this module requires that the USBPHY_ANACTRL[DEV_PULLDOWN] bit field be reset to value 1'b0 during the period when this module is in use.

The other registers generally do not need to be modified, because they default to values that comply with the USB Battery Charging Specification. However, several timing parameters can be changed for a great deal of flexibility if a particular system requires it.

All module configuration must occur *before* initiating the charger detection sequence. Configuration changes made *after* setting CONTROL[START] result in undefined behavior.

56.11 Application information

This section provides application information.

56.11.1 External pullups

Any external pullups applied to the USB D+ or D- data lines must be capable of being disabled to prevent incorrect pullup values or incorrect operation of the USB subsystem.

56.11.2 Dead or weak battery

According to the *USB Battery Charging Specification, Revision 1.2*, a USB device with a dead or weak battery that is attached to a downstream port can draw charging current from VBUS without connecting for several minutes, until the battery is charged to the point that the USB device can connect. The conditions for this operation are referred to as the Dead Battery Provision.

The *USB Battery Charging Specification, Revision 1.2* limits this charging condition to 100mA for a duration of 45 minutes. The later *USB 2.0 Connect Timing Update ECN* modifies this charging condition to 500mA for a duration of 2 minutes. Customers designing systems to take advantage of the Dead Battery Provision should study both specifications closely.

The USBDCD module is compatible with systems that do not check the strength of the battery. Therefore, this module assumes that the battery is good, so the USB device must immediately connect to the USB bus by pulling the USB_DP pin high after the USBDCD module has determined that the device is attached to an SDP or CDP. (By definition, a DCP does not support connection but still requires a signaling event, see Dedicated charging port.)

The module is also compatible with systems that have other circuitry to check the strength of the battery. In these systems, if it is known that the battery is weak or dead, software can delay connecting to the USB while charging using the Dead Battery Provision. Once the battery is charged to the good battery threshold, software must then connect to the USB host by pulling the USB_DP pin high.

While using the Dead Battery Provision, the USB_DP pin must signal by enabling V_{DP_SRC}. On this product that signaling can be done most simply by setting the USBDCD_SIGNAL_OVERRIDE[PS] bit field to value 2'b10.

56.11.3 Handling unplug events

If the device is unplugged from the USB bus during the charger detection sequence, the contents of the STATUS register must be ignored and the USBDCD module must get a Software Reset, as described in Software reset.

Chapter 57
Cyclic Redundancy Check (CRC)

57.1 Chip-specific CRC information

Table 57-1. Reference links to related information

Topic	Related module	Reference
Full description	CRC	CRC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

57.1.1 Cyclic Redundancy Check (CRC)

The CRC module is a hardware CRC generator circuit using 16/32-bit shift register. The CRC module supports error detection for all single, double, odd, and most multi-bits errors, programmable initial seed value, and optional feature to transpose input data and CRC result via transpose register. The following table shows the configuration of the CRC.

Table 57-2. CRC configuration

Parameter	Description
Name	Cyclic Redundancy Check (CRC)
Instances	1
Configurable features	NA
Interface speed	NA
External I/O pins	See the attached IOMUXC spreadsheet for pin details

57.2 Introduction

The cyclic redundancy check (CRC) module generates 16/32-bit CRC code for error detection.

The CRC module provides a programmable polynomial and other parameters required to implement a 16-bit or 32-bit CRC standard.

The 16/32-bit code is calculated for 32 bits of data at a time.

57.2.1 Features

Features of the CRC module include:

- Hardware CRC generator circuit using a 16-bit or 32-bit programmable shift register
- Programmable initial seed value and polynomial
- Option to transpose input data or output data (the CRC result) bitwise or bytewise. This option is required for certain CRC standards. A bytewise transpose operation is not possible when accessing the CRC data register via 8-bit accesses. In this case, the user's software must perform the bytewise transpose function.
- Option for inversion of final CRC result
- 32-bit CPU register programming interface

57.2.2 Block diagram

The following is a block diagram of the CRC.

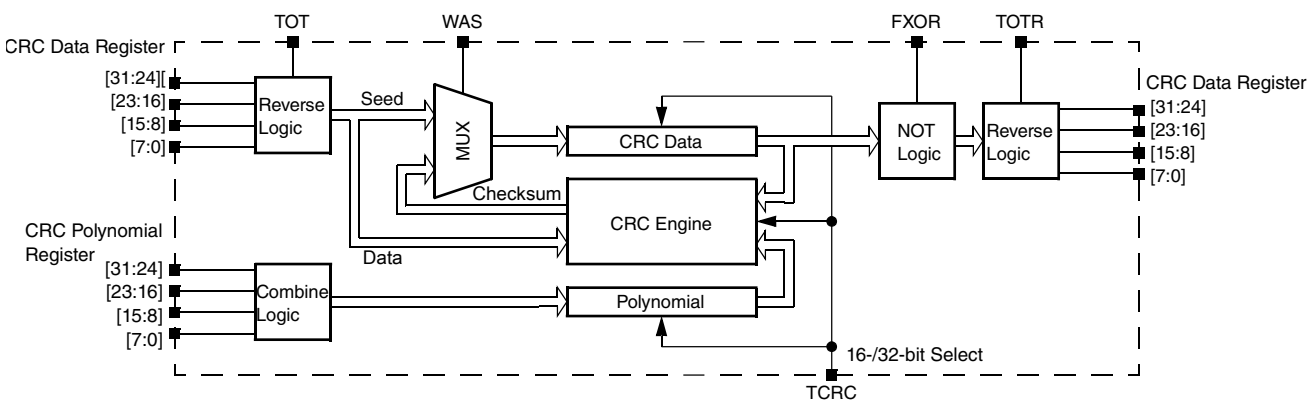


Figure 57-1. Programmable cyclic redundancy check (CRC) block diagram

57.2.3 Modes of operation

Various MCU modes affect the CRC module's functionality.

57.2.3.1 Run mode

This is the basic mode of operation.

57.2.3.2 Low-power modes (Wait or Stop)

Any CRC calculation in progress stops when the MCU enters a low-power mode that disables the module clock. It resumes after the clock is enabled or via the system reset for exiting the low-power mode. Clock gating for this module is dependent on the MCU.

57.3 Memory map and register descriptions

57.3.1 CRC register descriptions

57.3.1.1 CRC Memory map

CRC0 base address: 4102_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	CRC Data register (DATA)	32	RW	FFFF_FFFFh
4h	CRC Polynomial register (GPOLY)	32	RW	0000_1021h
8h	CRC Control register (CTRL)	32	RW	0000_0000h

57.3.1.2 CRC Data register (DATA)

57.3.1.2.1 Offset

Register	Offset
DATA	0h

57.3.1.2.2 Function

The CRC Data register contains the value of the seed, data, and checksum. When CTRL[WAS] is set, any write to the data register is regarded as the seed value. When CTRL[WAS] is cleared, any write to the data register is regarded as data for general CRC computation.

In 16-bit CRC mode, the HU and HL fields are not used for programming the seed value, and reads of these fields return an indeterminate value. In 32-bit CRC mode, all fields are used for programming the seed value.

When programming data values, the values can be written 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous; with MSB of data value written first.

After all data values are written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is available in the LU and LL fields. In 32-bit CRC mode, all fields contain the result. Reads of this register at any time return the intermediate CRC value, provided the CRC module is configured.

57.3.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HU								HL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LU								LL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

57.3.1.2.4 Fields

Field	Function
31-24 HU	CRC High Upper Byte

Table continues on the next page...

Field	Function
	In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
23-16 HL	CRC High Lower Byte In 16-bit CRC mode (CTRL[TCRC] is 0), this field is not used for programming a seed value. In 32-bit CRC mode (CTRL[TCRC] is 1), values written to this field are part of the seed value when CTRL[WAS] is 1. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.
15-8 LU	CRC Low Upper Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.
7-0 LL	CRC Low Lower Byte When CTRL[WAS] is 1, values written to this field are part of the seed value. When CTRL[WAS] is 0, data written to this field is used for CRC checksum generation.

57.3.1.3 CRC Polynomial register (GPOLY)

57.3.1.3.1 Offset

Register	Offset
GPOLY	4h

57.3.1.3.2 Function

This register contains the value of the polynomial for the CRC calculation. The HIGH field contains the upper 16 bits of the CRC polynomial, which are used only in 32-bit CRC mode. Writes to the HIGH field are ignored in 16-bit CRC mode. The LOW field contains the lower 16 bits of the CRC polynomial, which are used in both 16- and 32-bit CRC modes.

57.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HIGH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LOW															
Reset	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1

57.3.1.3.4 Fields

Field	Function
31-16 HIGH	High Polynominal Half-word Writable and readable in 32-bit CRC mode (CTRL[TCRC] is 1). This field is not writable in 16-bit CRC mode (CTRL[TCRC] is 0).
15-0 LOW	Low Polynominal Half-word Writable and readable in both 32-bit and 16-bit CRC modes.

57.3.1.4 CRC Control register (CTRL)

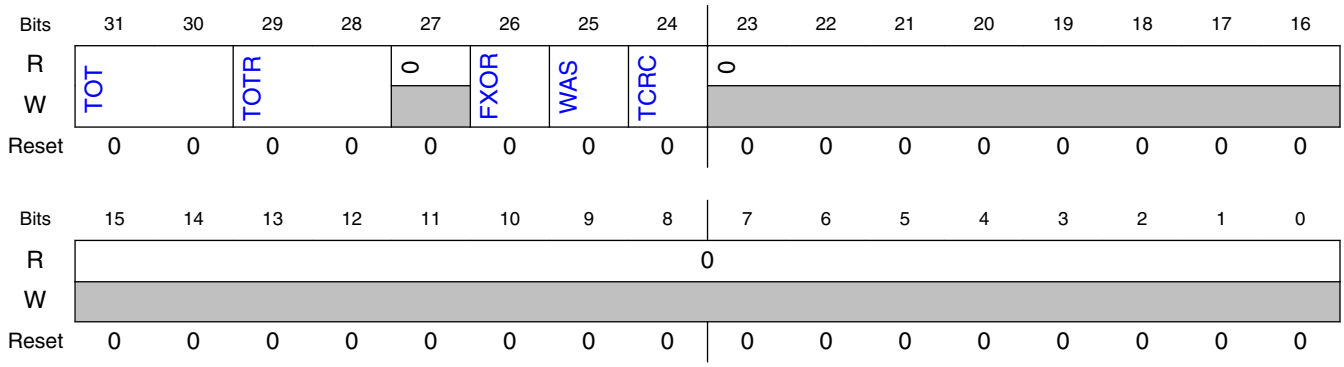
57.3.1.4.1 Offset

Register	Offset
CTRL	8h

57.3.1.4.2 Function

This register controls the configuration and working of the CRC module. Appropriate bits must be set before starting a new CRC calculation. A new CRC calculation is initialized by asserting CTRL[WAS] and then writing the seed into the CRC data register.

57.3.1.4.3 Diagram



57.3.1.4.4 Fields

Field	Function
31-30 TOT	Type Of Transpose For Writes Defines the transpose configuration of the data written to the CRC data register. See the description of the transpose feature for the available transpose options. 00b - No transposition. 01b - Bits in bytes are transposed; bytes are not transposed. 10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed; no bits in a byte are transposed.
29-28 TOTR	Type Of Transpose For Read Identifies the transpose configuration of the value read from the CRC Data register. See the description of the transpose feature for the available transpose options. 00b - No transposition. 01b - Bits in bytes are transposed; bytes are not transposed. 10b - Both bits in bytes and bytes are transposed. 11b - Only bytes are transposed; no bits in a byte are transposed.
27 —	Reserved
26 FXOR	Complement Read Of CRC Data Register Some CRC protocols require the final checksum to be XORed with 0xFFFFFFFF or 0xFFFF. Asserting this bit enables on the fly complementing of read data. 0b - No XOR on reading. 1b - Invert or complement the read value of the CRC Data register.
25 WAS	Write CRC Data Register As Seed When asserted, a value written to the CRC data register is considered a seed value. When deasserted, a value written to the CRC data register is taken as data for CRC computation. 0b - Writes to the CRC data register are data values. 1b - Writes to the CRC data register are seed values.
24 TCRC	TCRC Width of CRC protocol. 0b - 16-bit CRC protocol. 1b - 32-bit CRC protocol.
23-0 —	Reserved

57.4 Functional description

57.4.1 CRC initialization/reinitialization

To enable the CRC calculation, the user must program CRC_CTRL[WAS], CRC_GPOLY,necessary parameters for transposition and CRC result inversion in the applicable registers. Asserting CRC_CTRL[WAS] enables the programming of the seed value into the CRC_DATA register.

After a completed CRC calculation, the module can be reinitialized for a new CRC computation by reasserting CRC_CTRL[WAS] and programming a new, or previously used, seed value. All other parameters must be set before programming the seed value and subsequent data values.

57.4.2 CRC calculations

In 16-bit and 32-bit CRC modes, data values can be programmed 8 bits, 16 bits, or 32 bits at a time, provided all bytes are contiguous. Noncontiguous bytes can lead to an incorrect CRC computation.

57.4.2.1 16-bit CRC

To compute a 16-bit CRC:

1. Clear CRC_CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 16-bit polynomial to the CRC_GPOLY[LOW] field. The CRC_GPOLY[HIGH] field is not usable in 16-bit CRC mode.
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to CRC_DATA[LU:LL]. CRC_DATA[HU:HL] are not used.
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[LU:LL].

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

57.4.2.2 32-bit CRC

To compute a 32-bit CRC:

1. Set CRC_CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 32-bit polynomial to CRC_GPOLY[HIGH:LOW].
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to CRC_DATA[HU:HL:LU:LL].
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DATA[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DATA[HU:HL:LU:LL].
8. When all values have been written, read the final CRC result from CRC_DATA[HU:HL:LU:LL]. The CRC is calculated bytewise, and two clocks are required to complete one CRC calculation.

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

57.4.3 Transpose feature

By default, the transpose feature is not enabled. However, some CRC standards require the input data and/or the final checksum to be transposed. The user software has the option to configure each transpose operation separately, as desired by the CRC standard. The data is transposed on the fly while being read or written.

Some protocols use little endian format for the data stream to calculate a CRC. In this case, the transpose feature usefully flips the bits. This transpose option is one of the types supported by the CRC module.

57.4.3.1 Types of transpose

The CRC module provides several types of transpose functions to flip the bits and/or bytes, for both writing input data and reading the CRC result, separately using the CTRL[TOT] or CTRL[TOTR] fields, according to the CRC calculation being used.

The following types of transpose functions are available for writing to and reading from the CRC data register:

1. CTRL[TOT] or CTRL[TOTR] is 00.

   No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 01

   Bits in a byte are transposed, while bytes are not transposed.

   reg[31:0] becomes {reg[24:31], reg[16:23], reg[8:15], reg[0:7]}

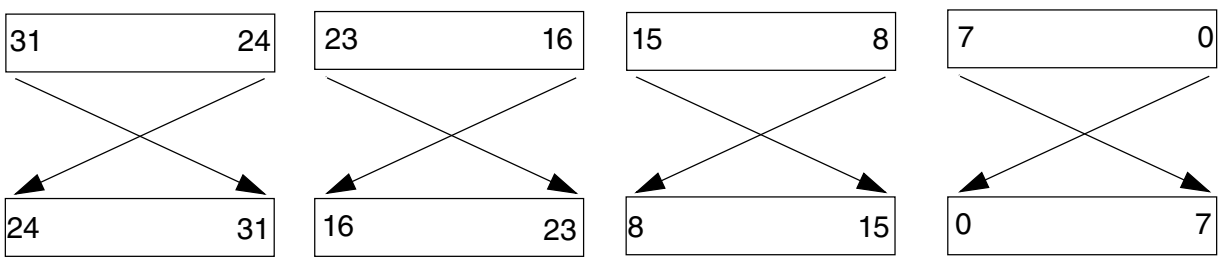


Figure 57-2. Transpose type 01

3. CTRL[TOT] or CTRL[TOTR] is 10.

   Both bits in bytes and bytes are transposed.

   reg[31:0] becomes = {reg[0:7], reg[8:15],reg[16:23], reg[24:31]}

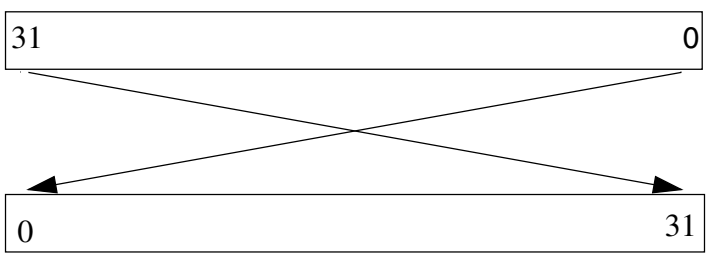


Figure 57-3. Transpose type 10

4. CTRL[TOT] or CTRL[TOTR] is 11.

   Bytes are transposed, but bits are not transposed.

   reg[31:0] becomes {reg[7:0], reg[15:8], reg[23:16], reg[31:24]}

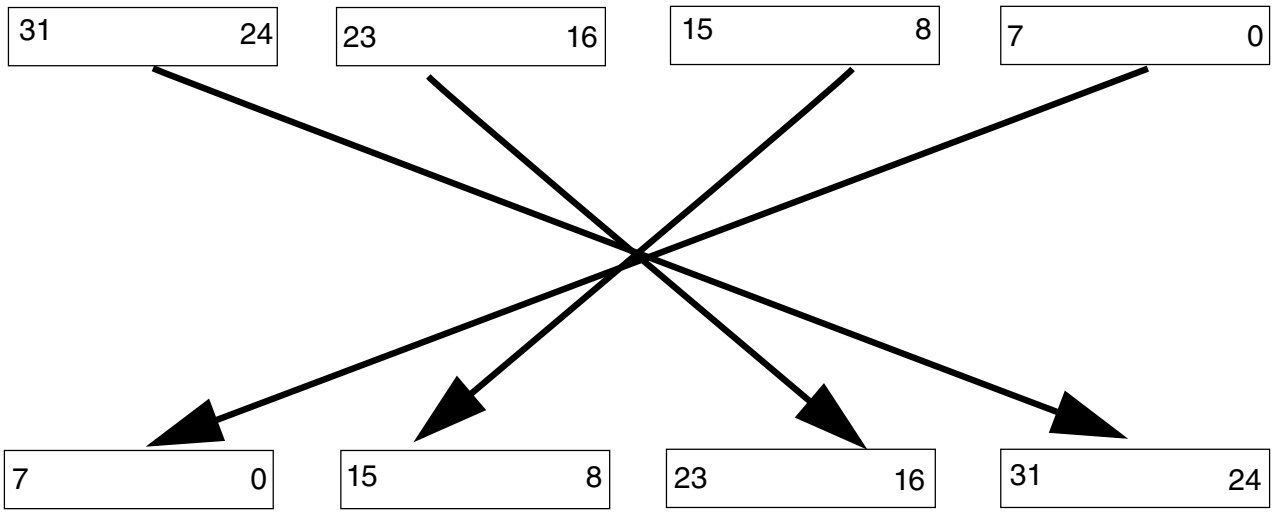


Figure 57-4. Transpose type 11

NOTE

For 8-bit and 16-bit write accesses to the CRC data register, the data is transposed with zeros on the unused byte or bytes (taking 32 bits as a whole), but the CRC is calculated on the valid byte(s) only. When reading the CRC data register for a 16-bit CRC result and using transpose options 10 and 11, the resulting value after transposition resides in the CRC[HU:HL] fields. The user software must account for this situation when reading the 16-bit CRC result, so reading 32 bits is preferred.

57.4.4 CRC result complement

When CTRL[FXOR] is set, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in the CRC data register every time the CRC data register is read. When CTRL[FXOR] is cleared, reading the CRC data register accesses the raw checksum value.

Chapter 58
Low Power Timer (LPTMR)

58.1 Chip-specific LPTMR information

Table 58-1. Reference links to related information

Topic	Related module	Reference
Full description	LPTMR	LPTMR
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

58.1.1 LPTMR

The Low Power Timer (LPTMR) can operate as real-time interrupt or pulse accumulator. It includes a 2^N prescaler (real-time interrupt mode) or glitch filter (pulse accumulator mode). An interrupt is generated when the counter equals the value in the the 16-bit compare register. The counter may reset when the interrupt is generated. This LPTMR module can remain functional when the chip is in low power modes, provided the reference clock to this timer is active.

Table 58-2. LPTMR configuration

Parameter	Description
Name	LPTMR
Instances	2
Configurable features	• LPTMR0-1: 16-bit timer, 1 channel • LPTMR0-1 input sources: See Table 58-3 and Table 58-4
Interface speed	-
External I/O pins	LPTMRx: ALT0-3. See the attached IOMUXC spreadsheet to know the pin details.

58.1.2 LPTMR input sources

The LPTMR_CSR[TPS] bitfield configures the input source used in pulse counter mode. The following tables shows the chip-specific input assignments for this bitfield.

Table 58-3. LPTMR0 pulse counter input options

LPTMR0_CSR[TPS]	Pulse counter input number	Chip input
00	0	CMP0_OUT
01	1	LPTMR0_ALT1 pin
10	2	LPTMR0_ALT2 pin
11	3	LPTMR0_ALT3 pin

Table 58-4. LPTMR1 pulse counter input options

LPTMR1_CSR[TPS]	Pulse counter input number	Chip input
00	0	CMP0_OUT
01	1	LPTMR1 Trigger
10	2	LPTMR1_ALT2 pin
11	3	LPTMR1_ALT3 pin

58.1.3 LPTMR glitch filter/prescaler clocking options

The prescaler and glitch filter of the LPTMR can be clocked from miscellaneous sources determined by LPTMR0_PSR[PCS] and LPTMR0_CSR[TPS] bitfields for LPTMR0 instance (and similarly for LPTMR1). The supported clock sources on this device are shown in LPTMR clocking.

58.2 Introduction

The low-power timer (LPTMR) can be configured to operate as a time counter with optional prescaler, or as a pulse counter with optional glitch filter, across all power modes, including the low-leakage modes. It can also continue operating through most system reset events, allowing it to be used as a time of day counter.

58.2.1 Features

The features of the LPTMR module include:

- 16-bit time counter or pulse counter with compare
  - Optional interrupt can generate asynchronous wakeup from any low-power mode
  - Hardware trigger output
  - Counter supports free-running mode or reset on compare
- Configurable clock source for prescaler/glitch filter
- Configurable input source for pulse counter
  - Rising-edge or falling-edge

58.2.2 Modes of operation

The following table describes the operation of the LPTMR module in various modes.

Table 58-5. Modes of operation

Modes	Description
Run	The LPTMR operates normally.
Wait	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Stop	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Low-Leakage	The LPTMR continues to operate normally and can be configured to exit the low-power mode by generating an interrupt request.
Debug	The LPTMR operates normally in Pulse Counter mode, but the counter does not increment in Time Counter mode.

58.3 LPTMR signal descriptions

Table 58-6. LPTMR signal descriptions

Signal	I/O	Description
LPTMR_ALT*n*	I	Pulse Counter Input pin

58.3.1 Detailed signal descriptions

Table 58-7. LPTMR interface—detailed signal descriptions

Signal	I/O	Description	
LPTMR_ALT*n*	I	Pulse Counter Input The LPTMR can select one of the input pins to be used in Pulse Counter mode.	
		State meaning	Assertion—If configured for pulse counter mode with active-high input, then assertion causes the CNR to increment. Deassertion—If configured for pulse counter mode with active-low input, then deassertion causes the CNR to increment.
		Timing	Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.

58.4 Memory map and register definition

NOTE

The LPTMR registers are reset only on a POR or LVD event.
See LPTMR power and reset for more details.

58.4.1 LPTMR register descriptions

58.4.1.1 LPTMR Memory map

LPTMR0 base address: 4102_E000h

LPTMR1 base address: 4102_F000h

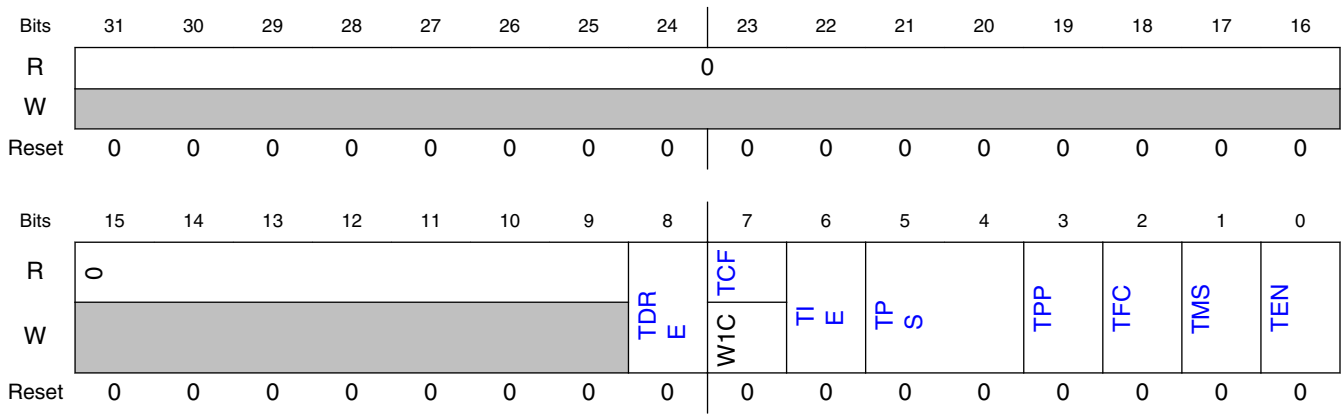
Offset	Register	Width (In bits)	Access	Reset value
0h	Low Power Timer Control Status Register (CSR)	32	RW	0000_0000h
4h	Low Power Timer Prescale Register (PSR)	32	RW	0000_0000h
8h	Low Power Timer Compare Register (CMR)	32	RW	0000_0000h
Ch	Low Power Timer Counter Register (CNR)	32	RW	0000_0000h

58.4.1.2 Low Power Timer Control Status Register (CSR)

58.4.1.2.1 Offset

Register	Offset
CSR	0h

58.4.1.2.2 Diagram



58.4.1.2.3 Fields

Field	Function
31-9 —	Reserved
8 TDRE	Timer DMA Request Enable When TDRE is set, the LPTMR DMA Request is generated whenever TCF is also set and the TCF is cleared when the DMA Controller is done. 0b - Timer DMA Request disabled. 1b - Timer DMA Request enabled.
7 TCF	Timer Compare Flag TCF is set when the LPTMR is enabled and the CNR equals the CMR and increments. TCF is cleared when the LPTMR is disabled or a logic 1 is written to it. 0b - The value of CNR is not equal to CMR and increments. 1b - The value of CNR is equal to CMR and increments.
6 TIE	Timer Interrupt Enable When TIE is set, the LPTMR Interrupt is generated whenever TCF is also set. 0b - Timer interrupt disabled. 1b - Timer interrupt enabled.
5-4 TPS	Timer Pin Select

Table continues on the next page...

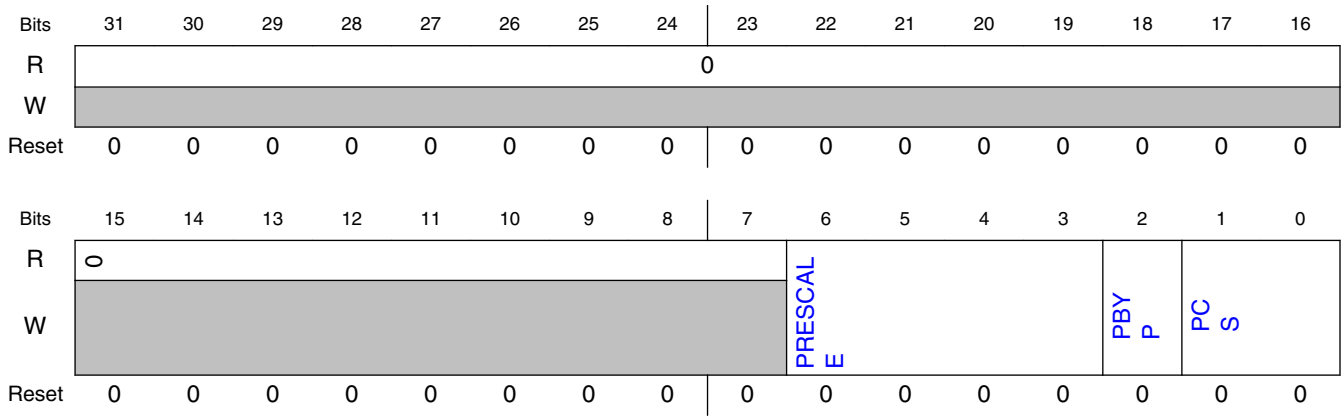
Field	Function
	Configures the input source to be used in Pulse Counter mode. TPS must be altered only when the LPTMR is disabled. The input connections vary by device. See the chip configuration information about connections to these inputs. 00b - Pulse counter input 0 is selected. 01b - Pulse counter input 1 is selected. 10b - Pulse counter input 2 is selected. 11b - Pulse counter input 3 is selected.
3 TPP	Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. TPP must be changed only when the LPTMR is disabled. 0b - Pulse Counter input source is active-high, and the CNR will increment on the rising-edge. 1b - Pulse Counter input source is active-low, and the CNR will increment on the falling-edge.
2 TFC	Timer Free-Running Counter When clear, TFC configures the CNR to reset whenever TCF is set. When set, TFC configures the CNR to reset on overflow. TFC must be altered only when the LPTMR is disabled. 0b - CNR is reset whenever TCF is set. 1b - CNR is reset on overflow.
1 TMS	Timer Mode Select Configures the mode of the LPTMR. TMS must be altered only when the LPTMR is disabled. 0b - Time Counter mode. 1b - Pulse Counter mode.
0 TEN	Timer Enable When TEN is clear, it resets the LPTMR internal logic, including the CNR and TCF. When TEN is set, the LPTMR is enabled. While writing 1 to this field, CSR[5:1] must not be altered. 0b - LPTMR is disabled and internal logic is reset. 1b - LPTMR is enabled.

58.4.1.3 Low Power Timer Prescale Register (PSR)

58.4.1.3.1 Offset

Register	Offset
PSR	4h

58.4.1.3.2 Diagram



58.4.1.3.3 Fields

Field	Function
31-7 —	Reserved
6-3 PRESCALE	Prescale Value Configures the size of the Prescaler in Time Counter mode or width of the glitch filter in Pulse Counter mode. The width of the glitch filter can vary by 1 cycle due to synchronization of the pulse counter input. PRESCALE must be altered only when the LPTMR is disabled. 0000b - Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration. 0001b - Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after 2 rising clock edges. 0010b - Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after 4 rising clock edges. 0011b - Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after 8 rising clock edges. 0100b - Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges. 0101b - Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges. 0110b - Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges. 0111b - Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges. 1000b - Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges. 1001b - Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges. 1010b - Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges. 1011b - Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges. 1100b - Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges. 1101b - Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges.

Table continues on the next page...

Field	Function
	1110b - Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges. 1111b - Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges.
2 PBYP	Prescaler Bypass When PBYP is set, the selected prescaler clock in Time Counter mode or selected input source in Pulse Counter mode directly clocks the CNR. When PBYP is clear, the CNR is clocked by the output of the prescaler/glitch filter. PBYP must be altered only when the LPTMR is disabled. 0b - Prescaler/glitch filter is enabled. 1b - Prescaler/glitch filter is bypassed.
1-0 PCS	Prescaler Clock Select Selects the clock to be used by the LPTMR prescaler/glitch filter. In time counter mode, this field selects the input clock to the prescaler. In pulse counter mode, this field selects the input clock to the glitch filter. PCS must be altered only when the LPTMR is disabled. The clock connections vary by device. **NOTE:** See the chip configuration details for information on the connections to these inputs. 00b - Prescaler/glitch filter clock 0 selected. 01b - Prescaler/glitch filter clock 1 selected. 10b - Prescaler/glitch filter clock 2 selected. 11b - Prescaler/glitch filter clock 3 selected.

58.4.1.4 Low Power Timer Compare Register (CMR)

58.4.1.4.1 Offset

Register	Offset
CMR	8h

58.4.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMPARE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

58.4.1.4.3 Fields

Field	Function
31-16 —	Reserved
15-0 COMPARE	Compare Value When the LPTMR is enabled and the CNR equals the value in the CMR and increments, TCF is set and the hardware trigger asserts until the next time the CNR increments. If the CMR is 0, the hardware trigger will remain asserted until the LPTMR is disabled. If the LPTMR is enabled, the CMR must be altered only when TCF is set.

58.4.1.5 Low Power Timer Counter Register (CNR)

58.4.1.5.1 Offset

Register	Offset
CNR	Ch

58.4.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNTER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

58.4.1.5.3 Fields

Field	Function
31-16 —	Reserved
15-0 COUNTER	Counter Value The CNR returns the current value of the LPTMR counter at the time this register was last written.

58.5 Functional description

58.5.1 LPTMR power and reset

The LPTMR remains powered in all power modes, including low-leakage modes. If the LPTMR is not required to remain operating during a low-power mode, then it must be disabled before entering the mode.

The LPTMR is reset only on global Power On Reset (POR) or Low Voltage Detect (LVD). When configuring the LPTMR registers, the CSR must be initially written with the timer disabled, before configuring the PSR and CMR. Then, CSR[TIE] must be set as the last step in the initialization. This ensures the LPTMR is configured correctly and the LPTMR counter is reset to zero following a warm reset.

58.5.2 LPTMR clocking

The LPTMR prescaler/glitch filter can be clocked by one of the four clocks. The clock source must be enabled before the LPTMR is enabled.

NOTE

The clock source selected by PSR[PCS] may need to be configured to remain enabled in low-power modes, otherwise the LPTMR will not operate during low-power modes.

In Pulse Counter mode with the prescaler/glitch filter bypassed, the selected input source directly clocks the CNR and no other clock source is required. To minimize power in this case, configure the prescaler clock source for a clock that is not toggling.

NOTE

The clock source or pulse input source selected for the LPTMR should not exceed the frequency f_{LPTMR} defined in the device datasheet.

58.5.3 LPTMR prescaler/glitch filter

The LPTMR prescaler and glitch filter share the same logic which operates as a prescaler in Time Counter mode and as a glitch filter in Pulse Counter mode.

NOTE

The prescaler/glitch filter configuration must not be altered when the LPTMR is enabled.

58.5.3.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks the CNR. When the LPTMR is enabled, the CNR will increment every 2^2 to 2^{16} prescaler clock cycles. After the LPTMR is enabled, the first increment of the CNR will take an additional one or two prescaler clock cycles due to synchronization logic.

58.5.3.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments the CNR on every clock cycle. When the LPTMR is enabled, the first increment will take an additional one or two prescaler clock cycles due to synchronization logic.

58.5.3.3 Glitch filter

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks the CNR. When the LPTMR is first enabled, the output of the glitch filter is asserted, that is, logic 1 for active-high and logic 0 for active-low. The following table shows the change in glitch filter output with the selected input source.

If	Then
The selected input source remains deasserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges	The glitch filter output will also deassert.
The selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising-edges	The glitch filter output will also assert.

NOTE

The input is only sampled on the rising clock edge.

The CNR will increment each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which the CNR can increment is once every 2^2 to 2^{16} prescaler clock edges. When first enabled, the glitch filter will wait an additional one or two prescaler clock edges due to synchronization logic.

58.5.3.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the CNR every time it asserts. Before the LPTMR is first enabled, the selected input source is forced to be asserted. This prevents the CNR from incrementing if the selected input source is already asserted when the LPTMR is first enabled.

58.5.4 LPTMR counter

The CNR increments by one on every:

- Prescaler clock in Time Counter mode with prescaler bypassed
- Prescaler output in Time Counter mode with prescaler enabled
- Input source assertion in Pulse Counter mode with glitch filter bypassed
- Glitch filter output in Pulse Counter mode with glitch filter enabled

The CNR is reset when the LPTMR is disabled or if the counter register overflows. If CSR[TFC] is cleared, then the CNR is also reset whenever CSR[TCF] is set.

When the core is halted in Debug mode:

- If configured for Pulse Counter mode, the CNR continues incrementing.
- If configured for Time Counter mode, the CNR stops incrementing.

The CNR cannot be initialized, but can be read at any time. On each read of the CNR, software must first write to the CNR with any value. This will synchronize and register the current value of the CNR into a temporary register. The contents of the temporary register are returned on each read of the CNR.

When reading the CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing, otherwise incorrect data may be returned.

58.5.5 LPTMR compare

When the CNR equals the value of the CMR and increments, the following events occur:

- CSR[TCF] is set.
- LPTMR interrupt is generated if CSR[TIE] is also set.
- LPTMR hardware trigger is generated.
- CNR is reset if CSR[TFC] is clear.

When the LPTMR is enabled, the CMR can be altered only when CSR[TCF] is set. When updating the CMR, the CMR must be written and CSR[TCF] must be cleared before the LPTMR counter has incremented past the new LPTMR compare value.

NOTE

When the LPTMR is enabled in Time Counter mode, the first increment will take an additional one or two clock cycles due to synchronization logic. This will result in the first compare (and therefore interrupt and hardware trigger) occuring slightly later. A faster prescaler clock or larger prescaler value will minimize this impact.

58.5.6 LPTMR interrupt

The LPTMR interrupt is generated whenever CSR[TIE] and CSR[TCF] are set. CSR[TCF] is cleared by disabling the LPTMR or by writing a logic 1 to it.

CSR[TIE] can be altered and CSR[TCF] can be cleared while the LPTMR is enabled.

The LPTMR interrupt is generated asynchronously to the system clock and can be used to generate a wakeup from any low-power mode, including the low-leakage modes, provided the LPTMR is enabled as a wakeup source.

58.5.7 LPTMR hardware trigger

The LPTMR hardware trigger asserts at the same time the CSR[TCF] is set and can be used to trigger hardware events in other peripherals without software intervention. The hardware trigger is always enabled.

When	Then
The CMR is set to 0 with CSR[TFC] clear	The LPTMR hardware trigger will assert on the first compare and does not deassert.
The CMR is set to a nonzero value, or, if CSR[TFC] is set	The LPTMR hardware trigger will assert on each compare and deassert on the following increment of the CNR.

Chapter 59
Watchdog Timer (WDOG)

59.1 Chip-specific WDOG information

Table 59-1. Reference links to related information

Topic	Related module	Reference
Full description	WDOG	WDOG
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

59.1.1 WDOG (Watchdog Timer)

The Watchdog Timer (WDOG) module keeps a watch on the system functioning and resets it in case of its failure. Reasons for failure include run-away software code and the stoppage of the system clock that in a safety critical system can lead to serious consequences. In such cases, the WDOG brings the system into a safe state of operation. The WDOG monitors the operation of the system by expecting periodic communication from the software, generally known as servicing or refreshing the WDOG. If this periodic refreshing does not occur, the WDOG resets the system. There is a WDOG module associated with each processor, and the third WDOG instance is for security software use.

Table 59-2. WDOG configuration

Parameter	Description
Name	WDOG
Instances	3
Configurable features	NA
Interface speed	NA
External I/O pins	NA

59.1.2 WDOG clock sources

i.MX 7ULP uses the following clock sources to the WDOG module:

- 1 kHz LPO clock from SCG/SIRC
- 32 kHz RTCOSC internal clock from SCG/SNVS
- PCC asyn clk slow peripheral external clock

NOTE

To configure the external reference clock (ext_clk) of the WDG1 (A7) and WDG2 (A7) in the A7 core, first, clear the CGC bit of the PCC_WDOG1 and PCC_WDOG2 registers using the CM4 core, otherwise the WDG does not receive the clock.

59.2 Introduction

The Watchdog Timer (WDOG) module is an independent timer that is available for system use. It provides a safety feature to ensure that software is executing as planned and that the CPU is not stuck in an infinite loop or executing unintended code. If the WDOG module is not serviced (refreshed) within a certain period, it resets the MCU.

59.2.1 Features

Features of the WDOG module include:

- Configurable clock source inputs independent from the bus clock
  - Bus clock
  - LPO clock
  - INTCLK (internal clock)
  - ERCLK (external reference clock)
- Programmable timeout period
  - Programmable 16-bit timeout value
  - Optional fixed 256 clock prescaler when longer timeout periods are needed

- Robust write sequence for counter refresh
  - Refresh sequence of writing 0xA602 and then 0xB480
- Window mode option for the refresh mechanism
  - Programmable 16-bit window value
  - Provides robust check that program flow is faster than expected
  - Early refresh attempts trigger a reset.
- Optional timeout interrupt to allow post-processing diagnostics
  - Interrupt request to CPU with interrupt vector for an interrupt service routine (ISR)
  - Forced reset occurs 128 bus clocks after the interrupt vector fetch.
- Configuration bits are write-once-after-reset to ensure watchdog configuration cannot be mistakenly altered.
- Robust write sequence for unlocking write-once configuration bits
  - Unlock sequence of writing 0xC520 and then 0xD928 for allowing updates to write-once configuration bits
  - Software must make updates within 128 bus clocks after unlocking and before WDOG closing unlock window.

59.2.2 Block diagram

The following figure shows a block diagram of the WDOG module.

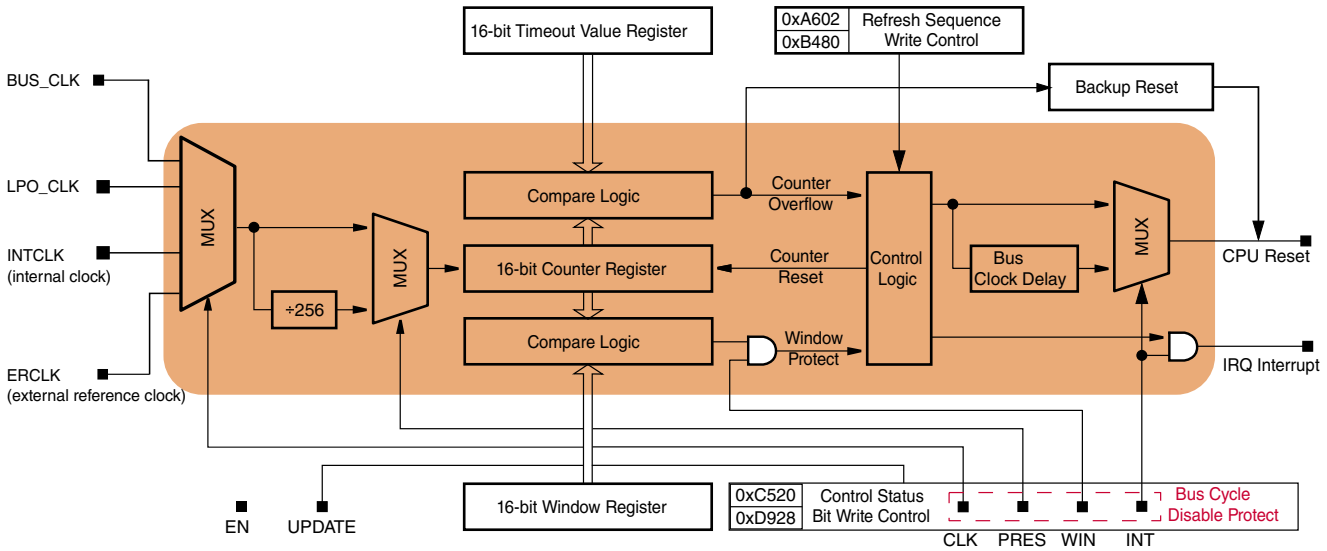


Figure 59-1. WDOG block diagram

59.3 Memory map and register definition

59.3.1 WDOG register descriptions

59.3.1.1 WDOG Memory map

WDOG0 base address: 4102_5000h

WDOG1 base address: 403D_0000h

WDOG2 base address: 4043_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Watchdog Control and Status Register (CS)	32	RW	0000_0980h
4h	Watchdog Counter Register (CNT)	32	RW	0000_0000h
8h	Watchdog Timeout Value Register (TOVAL)	32	RW	0000_0400h
Ch	Watchdog Window Register (WIN)	32	RW	0000_0000h

59.3.1.2 Watchdog Control and Status Register (CS)

59.3.1.2.1 Offset

Register	Offset
CS	0h

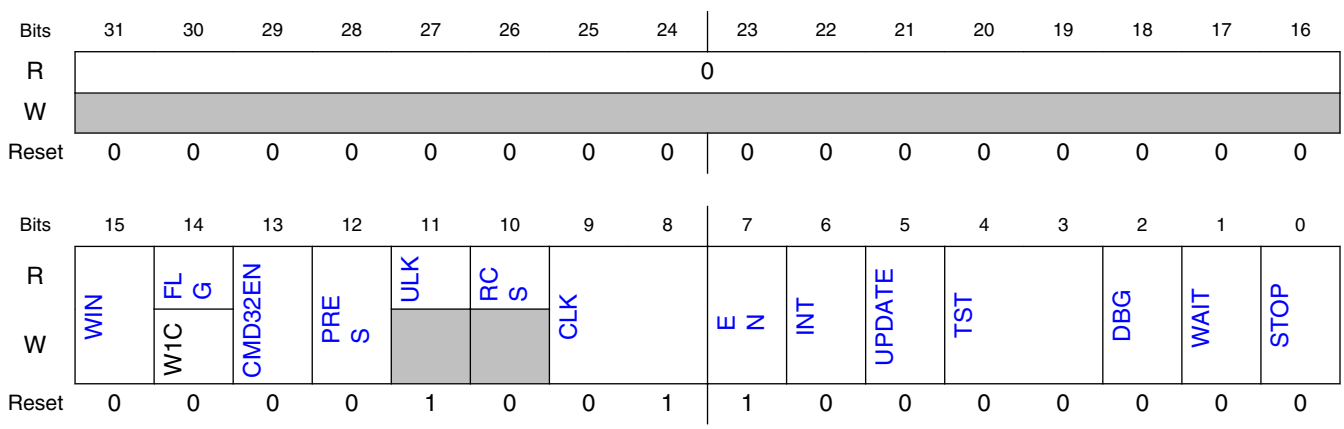
59.3.1.2.2 Function

This section describes the function of Watchdog Control and Status Register.

NOTE

TST is cleared (0:0) on POR only. Any other reset does not affect the value of this field.

59.3.1.2.3 Diagram



59.3.1.2.4 Fields

Field	Function
31-16 —	Reserved
15 WIN	Watchdog Window This write-once bit enables window mode. See the Window mode section. 0b - Window mode disabled. 1b - Window mode enabled.
14 FLG	Watchdog Interrupt Flag This bit is an interrupt indicator when INT is set in control and status register 1. Write 1 to clear it. 0b - No interrupt occurred.

Table continues on the next page...

Field	Function
	1b - An interrupt occurred.
13 CMD32EN	Enables or disables WDOG support for 32-bit (otherwise 16-bit or 8-bit) refresh/unlock command write words This is write-once field, and the user needs to unlock WDOG after writing this field for reconfiguration. 0b - Disables support for 32-bit refresh/unlock command write words. Only 16-bit or 8-bit is supported. 1b - Enables support for 32-bit refresh/unlock command write words. 16-bit or 8-bit is NOT supported.
12 PRES	Watchdog prescaler This write-once bit enables a fixed 256 pre-scaling of watchdog counter reference clock. (The block diagram shows this clock divider option.) 0b - 256 prescaler disabled. 1b - 256 prescaler enabled.
11 ULK	Unlock status This read-only bit indicates whether WDOG is unlocked or not. 0b - WDOG is locked. 1b - WDOG is unlocked.
10 RCS	Reconfiguration Success This read-only bit indicates whether the reconfiguration is successful or not. Default reset value is 0.This bit is set when new configuration takes effect, and is cleared by successful unlock command. 0b - Reconfiguring WDOG. 1b - Reconfiguration is successful.
9-8 CLK	Watchdog Clock This write-once field indicates the clock source that feeds the watchdog counter. See the Clock source section. 00b - Bus clock 01b - LPO clock 10b - INTCLK (internal clock) 11b - ERCLK (external reference clock)
7 EN	Watchdog Enable This write-once bit enables the watchdog counter to start counting. 0b - Watchdog disabled. 1b - Watchdog enabled.
6 INT	Watchdog Interrupt This write-once bit configures the watchdog to immediately generate an interrupt request upon a reset-triggering event (timeout or illegal write to the watchdog), before forcing a reset. After the interrupt vector fetch (which comes after the reset-triggering event), the reset occurs after a delay of 128 bus clocks. 0b - Watchdog interrupts are disabled. Watchdog resets are not delayed. 1b - Watchdog interrupts are enabled. Watchdog resets are delayed by 128 bus clocks from the interrupt vector fetch.
5 UPDATE	Allow updates This write-once bit allows software to reconfigure the watchdog without a reset. 0b - Updates not allowed. After the initial configuration, the watchdog cannot be later modified without forcing a reset. 1b - Updates allowed. Software can modify the watchdog configuration registers within 128 bus clocks after performing the unlock write sequence.
4-3 TST	Watchdog Test Enables the fast test mode. The test mode allows software to exercise all bits of the counter to demonstrate that the watchdog is functioning properly. See the Fast testing of the watchdog section.

Table continues on the next page...

Field	Function
	This write-once field is cleared (0:0) on POR only. Any other reset does not affect the value of this field. 00b - Watchdog test mode disabled. 01b - Watchdog user mode enabled. (Watchdog test mode disabled.) After testing the watchdog, software should use this setting to indicate that the watchdog is functioning normally in user mode. 10b - Watchdog test mode enabled, only the low byte is used. CNT[CNTLOW] is compared with TOVAL[TOVALLOW]. 11b - Watchdog test mode enabled, only the high byte is used. CNT[CNTHIGH] is compared with TOVAL[TOVALHIGH].
2 DBG	Debug Enable This write-once bit enables the watchdog to operate when the chip is in debug mode. 0b - Watchdog disabled in chip debug mode. 1b - Watchdog enabled in chip debug mode.
1 WAIT	Wait Enable This write-once bit enables the watchdog to operate when the chip is in wait mode. 0b - Watchdog disabled in chip wait mode. 1b - Watchdog enabled in chip wait mode.
0 STOP	Stop Enable This write-once bit enables the watchdog to operate when the chip is in stop mode. 0b - Watchdog disabled in chip stop mode. 1b - Watchdog enabled in chip stop mode.

59.3.1.3 Watchdog Counter Register (CNT)

59.3.1.3.1 Offset

Register	Offset
CNT	4h

59.3.1.3.2 Function

This section describes the watchdog counter register.

The watchdog counter register provides access to the value of the free-running watchdog counter. Software can read the counter register at any time.

Software cannot write directly to the watchdog counter; however, two write sequences to these registers have special functions:

1. The *refresh sequence* resets the watchdog counter to 0x0000. See the "Refreshing the Watchdog" section.
2. The *unlock sequence* allows the watchdog to be reconfigured without forcing a reset (when CS[UPDATE] = 1). See the "Configure for reconfigurable" section.

NOTE

All other writes to this register are illegal and force a reset.

59.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CNTHIGH								CNTLOW							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

59.3.1.3.4 Fields

Field	Function
31-16 —	Reserved
15-8 CNTHIGH	High byte of the Watchdog Counter
7-0 CNTLOW	Low byte of the Watchdog Counter

59.3.1.4 Watchdog Timeout Value Register (TOVAL)

59.3.1.4.1 Offset

Register	Offset
TOVAL	8h

59.3.1.4.2 Function

This section describes the watchdog timeout value register. TOVAL contains the 16-bit value used to set the timeout period of the watchdog.

The watchdog counter (CNT) is continuously compared with the timeout value (TOVAL). If the counter reaches the timeout value, the watchdog forces a reset triggering event.

NOTE

Do not write 0 to the TOVAL register (if CS[TST]=11b, then TOVALHIGH cannot be written as 0; if CS[TST]=10b, then TOVALLOW cannot be 0); otherwise, the watchdog always generates a reset.

59.3.1.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOVALHIGH								TOVALLOW							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

59.3.1.4.4 Fields

Field	Function
31-16 —	Reserved
15-8 TOVALHIGH	High byte of the timeout value
7-0 TOVALLOW	Low byte of the timeout value

59.3.1.5 Watchdog Window Register (WIN)

59.3.1.5.1 Offset

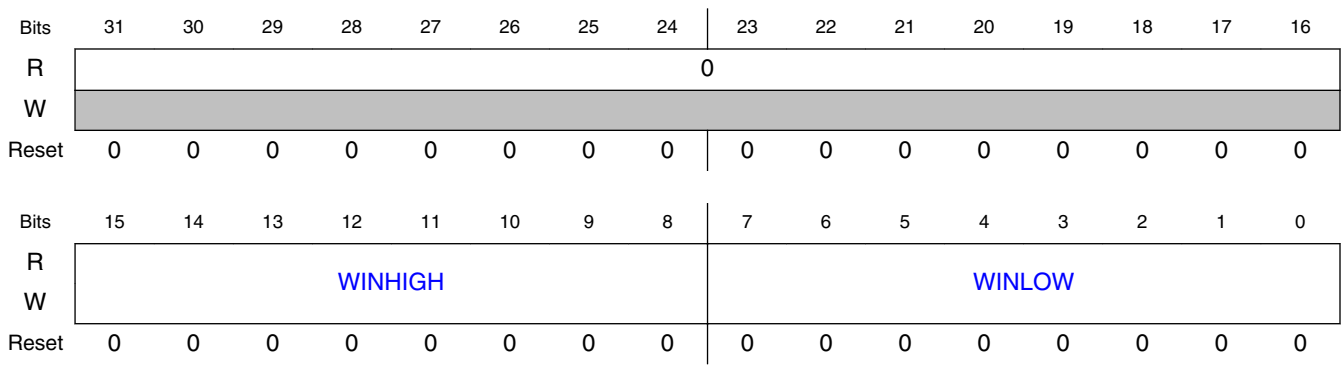
Register	Offset
WIN	Ch

59.3.1.5.2 Function

This section describes the watchdog window register. When window mode is enabled (CS[WIN] is set), The WIN register determines the earliest time that a refresh sequence is considered valid. See the Watchdog refresh mechanism section.

The WIN register value must be less than the TOVAL register value.

59.3.1.5.3 Diagram



59.3.1.5.4 Fields

Field	Function
31-16 —	Reserved
15-8 WINHIGH	High byte of Watchdog Window
7-0 WINLOW	Low byte of Watchdog Window

59.4 Functional description

The WDOG module provides a fail safe mechanism to ensure the system can be reset to a known state of operation in case of system failure, such as the CPU clock stopping or there being a run away condition in the software code. The watchdog counter runs continuously off a selectable clock source and expects to be serviced (refreshed) periodically. If it is not, it generates a reset triggering event.

59.4.1 Clock source

The watchdog counter has the following clock source options selected by programming CS[CLK]:

- bus clock
- Low-Power Oscillator clock (LPO_CLK)
- internal clock
- external clock

The options allow software to select a clock source independent of the bus clock for applications that need to meet more robust safety requirements. Using a clock source other than the bus clock ensures that the watchdog counter continues to run if the bus clock is somehow halted; see Backup reset.

NOTE

The *default* clock source of WDOG should be enabled after its funcitonal reset is deasserted, for the WDOG module to function properly.

An optional fixed prescaler for all clock sources allows for longer timeout periods. When CS[PRES] is set, the clock source is prescaled by 256 before clocking the watchdog counter.

The following table summarizes the different watchdog timeout periods which could be available, as an example.

Table 59-3. Watchdog timeout availability

Reference clock	Prescaler	Watchdog time-out availability
REF_CLK	Pass through	1×RCP to 65535×RCP [1]
	Enable	256×RCP to 16776960×RCP

1. RCP means Reference Clock Period.

NOTE

When the programmer switches clock sources during reconfiguration, the watchdog hardware holds the counter at zero for 2.5 periods of the previous clock source and 2.5 periods of the new clock source after the configuration time period (128 bus clocks) ends. This delay ensures a smooth transition before restarting the counter with the new configuration.

59.4.2 Watchdog refresh mechanism

The watchdog resets the MCU if the watchdog counter is not refreshed. A robust refresh mechanism makes it very unlikely that the watchdog can be refreshed by runaway code.

To refresh the watchdog counter, software must execute a refresh write sequence before the timeout period expires. In addition, if window mode is used, software must not start the refresh sequence until after the time value set in the WIN register. See the following figure.

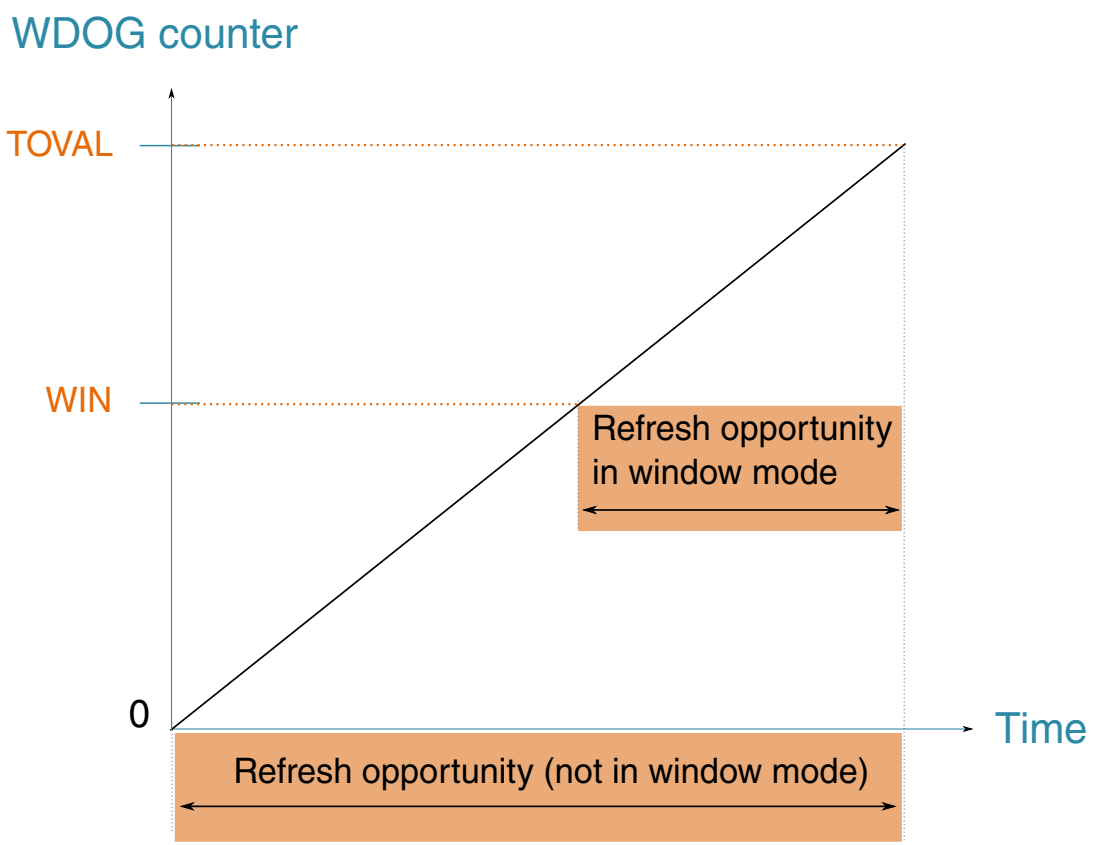


Figure 59-2. Refresh opportunity for the Watchdog counter

59.4.2.1 Window mode

Software finishing its main control loop faster than expected could be an indication of a problem. Depending on the requirements of the application, the WDOG can be programmed to force a reset when refresh attempts are early.

When Window mode is enabled, the watchdog must be refreshed after the counter has reached a minimum expected time value; otherwise, the watchdog resets the MCU. The minimum expected time value is specified in the WIN register. Setting CS[WIN] enables Window mode.

59.4.2.2 Refreshing the Watchdog

The refresh write sequence can be

- either two 16-bit writes (0xA602, 0xB480) or four 8-bit writes (0xA6, 0x02, 0xB4, 0x80) if WDOG_CS[CMD32EN] is 0;
- one 32-bit write (0xB480_A602) if WDOG_CS[CMD32EN] is 1.

to the CNT register. Both methods must occur before the WDG timeout; otherwise, the watchdog resets the MCU.

Note

Before starting the refresh sequence, disable the global interrupts. Otherwise, an interrupt could effectively invalidate the refresh sequence, if the interrupt occurs before the refresh writes finish. After the sequence finishes, restore the global interrupt control state.

The example codes can be found at the end of this chapter.

59.4.3 Configuring the Watchdog

59.4.3.1 Configuring the Watchdog Once

All watchdog control bits, timeout value, and window value are write-once after reset . This means that after a write has occurred they cannot be changed unless a reset occurs. This is guaranteed by the user configuring the window and timeout value first, followed by the other control bits, and ensuring that CS[UPDATE] is also set to 0.

This provides a robust mechanism to configure the watchdog and ensure that a runaway condition cannot mistakenly disable or modify the watchdog configuration after configured.

The new configuration takes effect only after all registers except CNT are written after reset. Otherwise, the WDOG uses the reset values by default. If window mode is not used (CS[WIN] is 0), writing to WIN is not required to make the new configuration take effect.

59.4.3.2 Reconfiguring the Watchdog

In some cases (like when supporting a bootloader function), you may want to reconfigure or disable the watchdog, *without forcing a reset first*.

- By setting CS[UPDATE] to 1 on the initial configuration of the watchdog after a reset, you can reconfigure the watchdog at any time by executing an unlock sequence.
- Conversely, if CS[UPDATE] remains 0, the only way to reconfigure the watchdog is by initiating a reset.

The unlock sequence is similar to the refresh sequence but uses different values.

59.4.3.2.1 Unlocking the Watchdog

The unlock sequence is a write to the CNT register of 0xC520 followed by 0xD928 within 16 bus clocks at any time after the watchdog has been configured. On completing the unlock sequence, the user must reconfigure the watchdog within 128 bus clocks; otherwise, the watchdog closes the unlock window.

NOTE

Due to the 128 bus clocks requirement for reconfiguring the watchdog, some delays must be inserted before executing STOP or WAIT instructions after reconfiguring the watchdog. This ensures that the watchdog's new configuration takes effect before the MCU enters low power mode. Otherwise, the MCU may not be waken up from low power mode.

The example codes can be found at end of this chapter.

59.4.4 Using interrupts to delay resets

- **When interrupts are enabled (CS[INT] = 1):** After a reset-triggering event (like a counter timeout or invalid refresh attempt), the watchdog first generates an interrupt request. Next, the watchdog delays 128 bus clocks (from the interrupt vector fetch, not the reset-triggering event) before forcing a reset, to allow the interrupt service routine (ISR) to perform tasks (like analyzing the stack to debug code).
- **When interrupts are disabled (CS[INT] = 0):** the watchdog does not delay the forcing a reset.

59.4.5 Backup reset

NOTE

A clock source other than the bus clock must be used as the reference clock for the counter; otherwise, the backup reset function is not available.

The backup reset function is a safeguard feature that independently generates a reset in case the main WDOG logic loses its clock (the bus clock) and can no longer monitor the counter. If the watchdog counter overflows twice in succession (without an intervening reset), the backup reset function takes effect and generates a reset.

Backup reset becomes valid when interrupt is enabled and the watchdog clock is from non bus clock. If interrupt is enabled, once the bus clock is cut off before exiting interrupt routine, the normal watchdog reset will be blocked. Under this case, the second overflow will cause backup reset directly.

59.4.6 Functionality in debug and low-power modes

By default, the watchdog is not functional in Debug mode, Wait mode, or Stop mode. However, the watchdog can remain functional in these modes as follows:

- **For Debug mode,** set CS[DBG]. (This way the watchdog is functional in Debug mode even when the CPU is held by the Debug module.)
- **For Wait mode,** set CS[WAIT].
- **For Stop mode,** set CS[STOP], CS[WAIT], and ensure the clock source is active in Stop mode.

NOTE

The watchdog can generate interrupt in Stop mode.

For Debug mode and Stop mode, in addition to the above configurations, a clock source other than the bus clock must be used as the reference clock for the counter; otherwise, the watchdog cannot function.

59.4.7 Fast testing of the watchdog

Before executing application code in safety critical applications, users are required to test that the watchdog works as expected and resets the MCU. Testing every bit of a 16-bit counter by letting it run to the overflow value takes a relatively long time (64 k clocks).

To help minimize the startup delay for application code after reset, the watchdog has a feature to test the watchdog more quickly by splitting the counter into its constituent byte-wide stages. The low and high bytes are run independently and tested for timeout against the corresponding byte of the timeout value register. (For complete coverage when testing the high byte of the counter, the test feature feeds the input clock via the 8th bit of the low byte, thus ensuring that the overflow connection from the low byte to the high byte is tested.)

Using this test feature reduces the test time to 512 clocks (not including overhead, such as user configuration and reset vector fetches). To further speed testing, use a faster clock (such as the bus clock) for the counter reference.

On a power-on reset, the POR bit in the system reset register is set, indicating the user should perform the WDOG fast test.

59.4.7.1 Testing each byte of the counter

The test procedure follows these steps:

1. Program the preferred watchdog timeout value in the TOVAL register during the watchdog configuration time period.
2. Select a byte of the counter to test using the CS[TST] = 10b for the low byte; CS[TST] = 11b for the high byte.
3. Wait for the watchdog to timeout. Optionally, in the idle loop, increment RAM locations as a parallel software counter for later comparison. Because the RAM is not affected by a watchdog reset, the timeout period of the watchdog counter can be compared with the software counter to verify the timeout period has occurred as expected.
4. The watchdog counter times out and forces a reset.
5. Confirm the WDOG flag in the system reset register is set, indicating that the watchdog caused the reset. (The POR flag remains clear.)
6. Confirm that CS[TST] shows a test (10b or 11b) was performed.

If confirmed, the count and compare functions work for the selected byte. Repeat the procedure, selecting the other byte in step 2.

NOTE

CS[TST] is cleared by a POR only and not affected by other resets.

59.4.7.2 Entering user mode

After successfully testing the low and high bytes of the watchdog counter, the user can configure CS[TST] to 01b to indicate the watchdog is ready for use in application user mode. Thus if a reset occurs again, software can recognize the reset trigger as a real watchdog reset caused by runaway or faulty application code.

As an ongoing test when using the default clock source, software can periodically read the CNT register to ensure the counter is being incremented.

59.5 Application Information

The watchdog is enabled by default after reset. To disable or reconfigure the watchdog, it is better to be done before the first watchdog timeout. It is suggested to disable or reconfigure the watchdog at the very beginning of the software code, e.g. beginning of the startup or main function.

NOTE

When the watchdog is configured by user, it needs at least 2.5 periods of watchdog clock to take effect. This means interval between two configures by user must be larger than 2.5 clocks.

NOTE

When Chip startup from BOOT ROM then jump to flash, the watchdog would be disabled in the beginning of bootloader, and enabled when bootloader exits. If there is any code in the flash program want to reconfigure the watchdog, it must be run 2.5 watchdog clocks later after the bootloader exits.

To disable or reconfigure the watchdog without forcing a reset, WDOG_CS[UPDATE] bit must be set during the initial configuration of the WDOG module. Then, the unlock sequence can be used at any time within the timeout limit to reconfigure the watchdog.

59.5.1 Disable Watchdog

To disable the watchdog, first do unlock sequence, then unset the WDOG_CS[EN] bit. The code snippet below shows an example for 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
WDOG_CS &= ~WDOG_CS_EN_MASK; //disable watchdog
EnableInterrupts; //enable global interrupt
```

59.5.2 Disable Watchdog after Reset

All of watchdog registers are unlocked by reset. Therefore, unlock sequence is unnecessary, but it needs to write all of watchdog registers to make the new configuration take effect. The code snippet below shows an example of disabling watchdog after reset.

```
DisableInterrupts; // disable global interrupt
WDOG_CS &= ~WDOG_CS_EN_MASK; // disable watchdog
WDOG_TOVAL= 0xFFFF;
while(WDOG_CS[ULK]);  // waiting for lock
while(~WDOG_CS[RCS]); // waiting for new configuration to take effect
EnableInterrupts; // enable global interrupt
```

59.5.3 Configure Watchdog

The watchdog can be configured once by set the WDOG_CS[UPDATE]=0. After that, the watchdog cannot be reconfigured until a reset. If set WDOG_CS[UPDATE]=1 when configuring the watchdog, the watchdog can be reconfigured without forcing a reset. The following example code shows how to configure the watchdog without window mode, clock source as LPO, interrupt enabled and timeout value to 256 clocks. The code snippet below shows an example for 32-bit write.

Configure once

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(0);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

Configure for reconfigurable

```
DisableInterrupts; //disable global interrupt
WDOG_CNT = 0xD928C520; //unlock watchdog
while(WDOG_CS[ULK]==0);  //wait until registers are unlocked
WDOG_TOVAL = 256; //set timeout value
WDOG_CS = WDOG_CS_EN(1) | WDOG_CS_CLK(1) | WDOG_CS_INT(1) |
          WDOG_CS_WIN(0) | WDOG_CS_UPDATE(1);
while(WDOG_CS[RCS]==0); //wait until new configuration takes effect
EnableInterrupts; //enable global interrupt
```

59.5.4 Refreshing the Watchdog

To refresh the watchdog and reset the watchdog counter to zero, a refresh sequence is required. The code snippet below shows an example for 32-bit write.

```
DisableInterrupts; // disable global interrupt
WDOG_CNT = 0xB480A602; // refresh watchdog
EnableInterrupts; // enable global interrupt
```

Chapter 60
Time Stamp Timer (TSTMR)

60.1 Chip-specific TSTMR information

Table 60-1. Reference links to related information

Topic	Related module	Reference
Full description	TSTMR	TSTMR
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

60.1.1 Time stamp timer module (TSTMR)

The TSTMR module is a free running incremental counter that starts running after system reset deassertion and can be read at any time by the software for determining the software ticks. The TSTMR is a 64-bit clock cycle counter. It runs off the 1 MHz clock and resets on every system reset. The counter only stops when the clock to the TSTMR is disabled.

M4 and A7 TSTMRs are instantiated inside SIM module and timer values are mapped in SIM address spaces.

Table 60-2. TSTMR configuration

Parameter	Description
Name	Time Stamp Timer Module (TSTMR)
Instances	2 instances (one in A7 and other in M4)
Configurable features	NA
Interface speed	NA
External I/O pins	NA

60.2 Introduction

The Time Stamp Timer (TSTMR) is a 64-bit clock cycle counter, reset by system reset.

60.2.1 Features

- Free-running Time Stamp Timer

60.3 TSTMR A Memory map and register definition

The TSTMR A Memory Map/Register Definition can be found here.

The TSTMR A registers can be accessed from the A-Side Processor.

NOTE

TSTMR registers can be read with 32-bit accesses only.

60.3.1 TSTMRA register descriptions

This section contains the detailed register descriptions for the TSTMRA registers.

60.3.1.1 TSTMRA Memory map

TSTMRA base address: 410A_3C00h

Offset	Register	Width (In bits)	Access	Reset value
0h	Time Stamp Timer Register Low (LOW)	32	RO	0000_0000h
4h	Time Stamp Timer Register High (HIGH)	32	RO	0000_0000h

60.3.1.2 Time Stamp Timer Register Low (LOW)

60.3.1.2.1 Offset

Register	Offset
LOW	0h

60.3.1.2.2 Function

The Time Stamp Timer is a 64-bit clock cycle counter, reset by system reset. It is readable by way of the TSTMR HIGH and LOW registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Note the following:

- A complete read operation should include both TSTMR LOW and HIGH reads. If a HIGH read does not follow a LOW read, then any other Time Stamp value read will be locked at a fixed value.
- The TSTMR LOW read should occur first, followed by the TSTMR HIGH read.

60.3.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

60.3.1.2.4 Fields

Field	Function
31-0 VALUE	Time Stamp Timer Low Lower 32 bits of the 64-bit time stamp value. The software must read the LOW first, followed by a read to HIGH to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

60.3.1.3 Time Stamp Timer Register High (HIGH)

60.3.1.3.1 Offset

Register	Offset
HIGH	4h

60.3.1.3.2 Function

The Time Stamp Timer is a 64-bit counters clock cycle counter, reset by system reset. It is readable by way of the TSTMR HIGH and LOW registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Note the following:

- A complete read operation should include both TSTMR LOW and HIGH reads. If a HIGH read does not follow a LOW read, then any other Time Stamp value read will be locked at a fixed value.
- The TSTMR LOW read should occur first, followed by the TSTMR HIGH read.

60.3.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

60.3.1.3.4 Fields

Field	Function
31-0 VALUE	Time Stamp Timer High Upper 32 bits of the 64-bit time stamp value. The software must read the LOW first, followed by a read to HIGH to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

60.4 TSTMR B Memory map and register definition

The TSTMR B Memory Map/Register Definition can be found here.

The TSTMR B registers can be accessed from the B-Side Processor.

NOTE

TSTMR registers can be read with 32-bit accesses only.

60.4.1 TSTMRB register descriptions

This section contains the detailed register descriptions for the TSTMRB registers.

60.4.1.1 TSTMRB Memory map

TSTMRB base address: 410A_3C08h

Offset	Register	Width (In bits)	Access	Reset value
0h	Time Stamp Timer Register Low (LOW)	32	RO	0000_0000h
4h	Time Stamp Timer Register High (HIGH)	32	RO	0000_0000h

60.4.1.2 Time Stamp Timer Register Low (LOW)

60.4.1.2.1 Offset

Register	Offset
LOW	0h

60.4.1.2.2 Function

The Time Stamp Timer is a 64-bit clock cycle counter, reset by system reset. It is readable by way of the TSTMR HIGH and LOW registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Note the following:

- A complete read operation should include both TSTMR LOW and HIGH reads. If a HIGH read does not follow a LOW read, then any other Time Stamp value read will be locked at a fixed value.
- The TSTMR LOW read should occur first, followed by the TSTMR HIGH read.

60.4.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

60.4.1.2.4 Fields

Field	Function
31-0 VALUE	Time Stamp Timer Low Lower 32 bits of the 64-bit time stamp value. The software must read the LOW first, followed by a read to HIGH to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

60.4.1.3 Time Stamp Timer Register High (HIGH)

60.4.1.3.1 Offset

Register	Offset
HIGH	4h

60.4.1.3.2 Function

The Time Stamp Timer is a 64-bit counters clock cycle counter, reset by system reset. It is readable by way of the TSTMR HIGH and LOW registers. Only 32-bit read accesses are allowed. Any other access will generate a transfer error.

Note the following:

- A complete read operation should include both TSTMR LOW and HIGH reads. If a HIGH read does not follow a LOW read, then any other Time Stamp value read will be locked at a fixed value.
- The TSTMR LOW read should occur first, followed by the TSTMR HIGH read.

60.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VALUE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

60.4.1.3.4 Fields

Field	Function
31-0 VALUE	Time Stamp Timer High Upper 32 bits of the 64-bit time stamp value. The software must read the LOW first, followed by a read to HIGH to retrieve the complete value. The all-zero reset value may not be readable because the value will increment before it can be read by the software.

60.5 Functional description

The TSTMR module is a free running incrementing counter that starts running after system reset de-assertion and can be read at any time by the software for determining the software ticks. However, the software must follow the read sequence as mentioned in the TSTMR register descriptions for correctly reading the TSTMR value. The TSTMR is a

64-bit counter and hence requires two 32-bit reads to read the full value. The TSTMR runs off the 1 MHz clock and resets on every system reset. The counter only stops when the clock to the TSTMR is disabled.

See the chip-specific TSTMR information for implementation details of this module's instances.

Chapter 61
External Watchdog Monitor (EWM)

61.1 Chip-specific EWM information

Table 61-1. Reference links to related information

Topic	Related module	Reference
Full description	EWM	EWM
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

61.1.1 External Watchdog Monitor

The External Watchdog Monitor (EWM) is designed to monitor external circuits, as well as the MCU software flow. This provides a back-up mechanism to the internal WDOG that can reset the system. The EWM has 1 kHz LPO clock source.

The EWM differs from the internal WDOG in that it does not reset the system. The EWM, if allowed to time-out, provides an independent trigger pin that when asserted resets or places an external circuit into a safe mode. The processor resets the EWM counter that is logically ANDed with an external digital input pin. This pin allows an external circuit to influence the reset_out signal.

Table 61-2. EWM configuration

Parameter	Description
Name	EWM
Instances	1
Configurable features	NA
Interface speed	NA

Table continues on the next page...

Table 61-2. EWM configuration (continued)

Parameter	Description
External I/O pins	EWM_OUT_b, EWM_IN. See the attached IOMUXC spreadsheet for pin details.

61.1.2 CLKPRESCALER register implementation

The CLKPRESCALER register is used for prescaling the clock frequency of low power clock source (~1 kHz LPO for i.MX 7ULP). Also, the number of bits of this register depends upon clock frequency of this clock source and the desired prescaled value.

61.2 Introduction

For safety, a redundant watchdog system, External Watchdog Monitor (EWM), is designed to monitor external circuits, as well as the MCU software flow. This provides a back-up mechanism to the internal watchdog that resets the MCU's CPU and peripherals.

The watchdog is generally used to monitor the flow and execution of embedded software within an MCU. The watchdog consists of a counter that if allowed to overflow, forces an internal reset (asynchronous) to all on-chip peripherals and optionally assert the RESET_B pin to reset external devices/circuits. The overflow of the watchdog counter must not occur if the software code works well and services the watchdog to re-start the actual counter.

The EWM differs from the internal watchdog in that it does not reset the MCU's CPU and peripherals. The EWM provides an independent EWM_OUT_b signal that when asserted resets or places an external circuit into a safe mode. The EWM_OUT_b signal is asserted upon the EWM counter time-out. An optional external input EWM_in is provided to allow additional control of the assertion of EWM_OUT_b signal.

61.2.1 Features

Features of EWM module include:

- Independent LPO_CLK clock source
- Programmable time-out period specified in terms of number of EWM LPO_CLK clock cycles.
- Windowed refresh option

    - Provides robust check that program flow is faster than expected.
    - Programmable window.
    - Refresh outside window leads to assertion of EWM_OUT_b.
- Robust refresh mechanism
    - Write values of 0xB4 and 0x2C to EWM Refresh Register within 23 peripheral bus clock cycles.
- One output port, EWM_OUT_b, when asserted is used to reset or place the external circuit into safe mode.
- One Input port, EWM_in, allows an external circuit to control the assertion of the EWM_OUT_b signal.

61.2.2 Modes of Operation

This section describes the module's operating modes.

61.2.2.1 Stop Mode

When the EWM is in stop mode, the CPU refreshes to the EWM cannot occur. On entry to stop mode, the EWM's counter freezes.

There are two possible ways to exit from Stop mode:

- On exit from stop mode through a reset, the EWM remains disabled.
- On exit from stop mode by an interrupt, the EWM is re-enabled, and the counter continues to be clocked from the same value prior to entry to stop mode.

Note the following if the EWM enters the stop mode during CPU refresh mechanism: At the exit from stop mode by an interrupt, refresh mechanism state machine starts from the previous state which means, if first refresh command is written correctly and EWM enters the stop mode immediately, the next command has to be written within the next *23* peripheral bus clocks after exiting from stop mode. User must mask all interrupts prior to executing EWM refresh instructions.

61.2.2.2 Debug Mode

Entry to debug mode has no effect on the EWM.

- If the EWM is enabled prior to entry of debug mode, it remains enabled.
- If the EWM is disabled prior to entry of debug mode, it remains disabled.

61.2.3 Block Diagram

This figure shows the EWM block diagram.

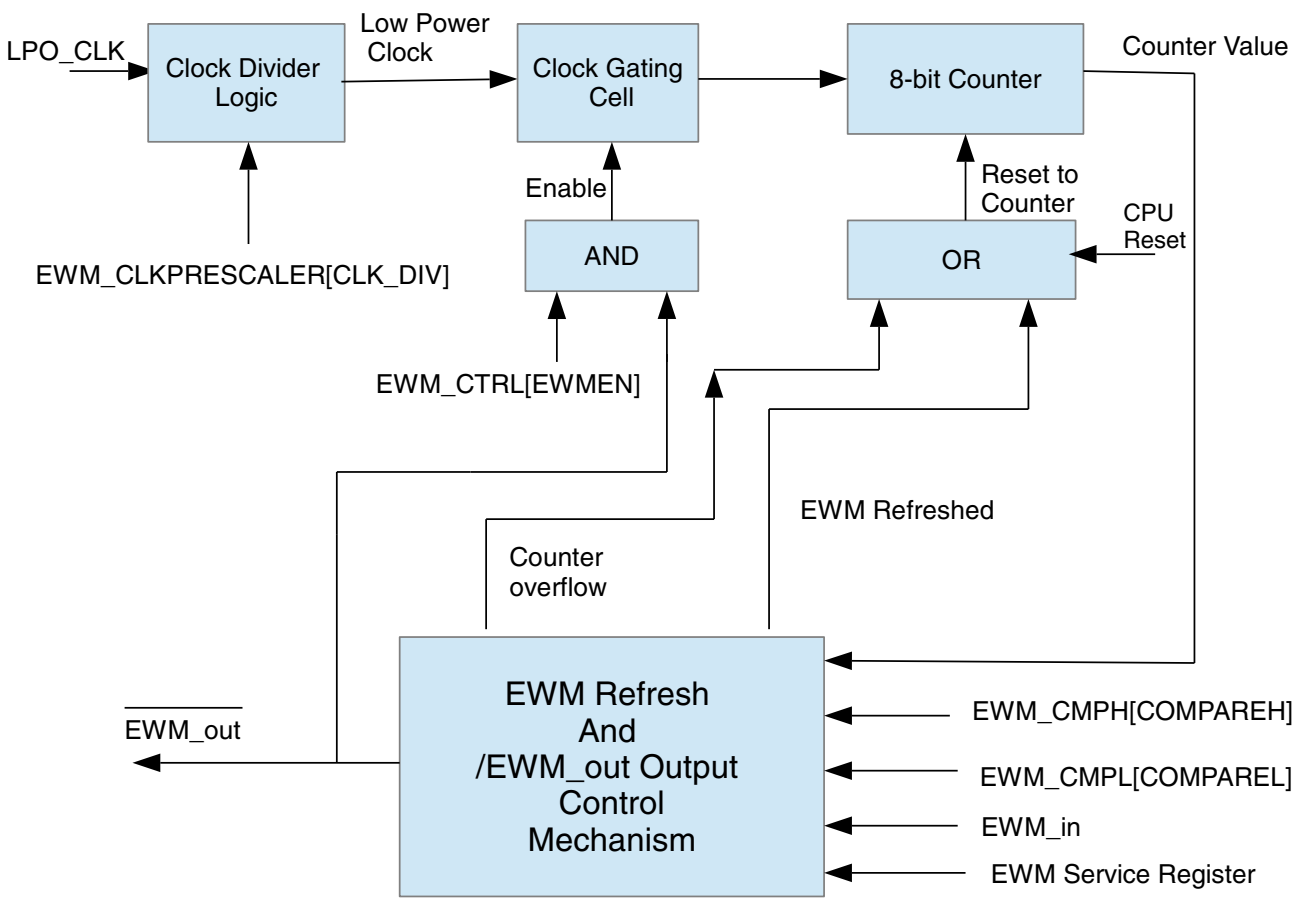


Figure 61-1. EWM Block Diagram

61.3 EWM Signal Descriptions

The EWM has two external signals and internal options for the counter clock sources, as shown in the following table.

NOTE

All active-low signals are now represented with the suffix "_b" throughout the chapter.

Table 61-3. EWM Signal Descriptions

Signal	Description	I/O
EWM_in	EWM input for safety status of external safety circuits. The polarity of EWM_in is programmable using the EWM_CTRL[ASSIN] bit. The default polarity is active low.	I
EWM_OUT_b	EWM reset out signal	O
lpo_clk[3:0]	Low power clock sources for running counter	I

61.4 Memory Map/Register Definition

This section contains the module memory map and registers.

NOTE

EWM only supports 8-bit register access. 16-bit and 32-bit access are not possible.

61.4.1 EWM register descriptions

61.4.1.1 EWM Memory map

EWM base address: 410A_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Register (CTRL)	8	RW	00h
1h	Service Register (SERV)	8	WORZ	00h
2h	Compare Low Register (CMPL)	8	RWONCE	00h
3h	Compare High Register (CMPH)	8	RWONCE	FFh
4h	Clock Control Register (CLKCTRL)	8	RWONCE	00h
5h	Clock Prescaler Register (CLKPRESCALER)	8	RWONCE	00h

61.4.1.2 Control Register (CTRL)

61.4.1.2.1 Offset

Register	Offset
CTRL	0h

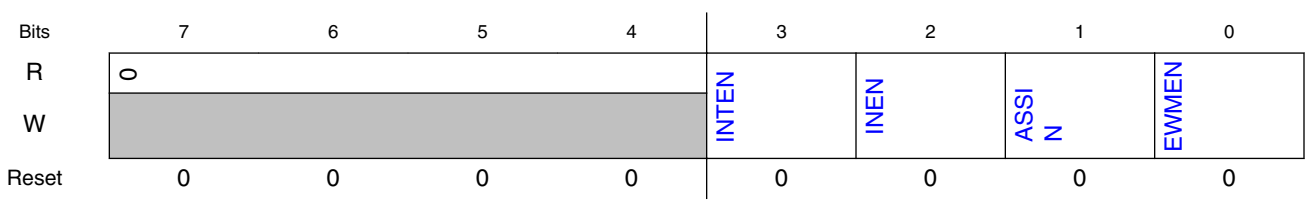
61.4.1.2.2 Function

The CTRL register is cleared by any reset.

NOTE

INEN, ASSIN and EWMEN bits can be written once after a CPU reset. Modifying these bits more than once, generates a bus transfer error.

61.4.1.2.3 Diagram



61.4.1.2.4 Fields

Field	Function
7-4 —	Reserved
3 INTEN	Interrupt Enable. This bit when set and EWM_OUT_b is asserted, an interrupt request is generated. To de-assert interrupt request, user should clear this bit by writing 0.
2 INEN	Input Enable. This bit when set, enables the EWM_in port.
1 ASSIN	EWM_in's Assertion State Select. Default assert state of the EWM_in signal is logic zero. Setting the ASSIN bit inverts the assert state of EWM_in signal to a logic one.
0	EWM enable.

Field	Function
EWMEN	This bit when set, enables the EWM module. This resets the EWM counter to zero and deasserts the EWM_OUT_b signal. This bit when unset, keeps the EWM module disabled. It cannot be re-enabled until a next reset, due to the write-once nature of this bit.

61.4.1.3 Service Register (SERV)

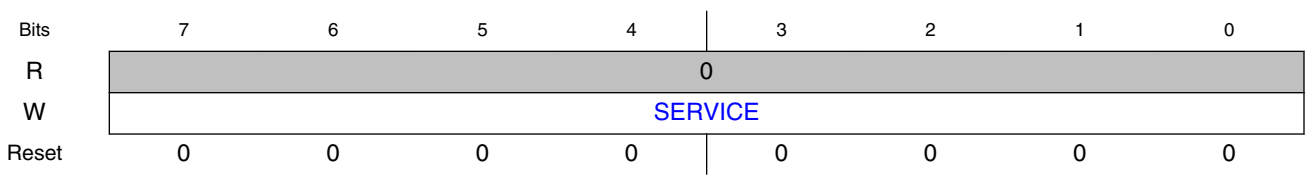
61.4.1.3.1 Offset

Register	Offset
SERV	1h

61.4.1.3.2 Function

The SERV register provides the interface from the CPU to the EWM module. It is write-only and reads of this register return zero.

61.4.1.3.3 Diagram



61.4.1.3.4 Fields

Field	Function
7-0 SERVICE	SERVICE The EWM refresh mechanism requires the CPU to write two values to the SERV register: a first data byte of 0xB4, followed by a second data byte of 0x2C. The EWM refresh is invalid if either of the following conditions is true. • The first or second data byte is not written correctly. • The second data byte is not written within a fixed number of peripheral bus cycles of the first data byte. This fixed number of cycles is called *EWM_refresh_time*. 23 peripheral bus clock cycles are required for *EWM_refresh_time*

61.4.1.4 Compare Low Register (CMPL)

61.4.1.4.1 Offset

Register	Offset
CMPL	2h

61.4.1.4.2 Function

The CMPL register is reset to zero after a CPU reset. This provides no minimum time for the CPU to refresh the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

61.4.1.4.3 Diagram

Bits	7	6	5	4	3	2	1	0
R W	COMPAREL							
Reset	0	0	0	0	0	0	0	0

61.4.1.4.4 Fields

Field	Function
7-0 COMPAREL	COMPAREL To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) minimum refresh time is required.

61.4.1.5 Compare High Register (CMPH)

61.4.1.5.1 Offset

Register	Offset
CMPH	3h

61.4.1.5.2 Function

The CMPH register is reset to 0xFF after a CPU reset. This provides a maximum of 256 clocks time, for the CPU to refresh the EWM counter.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

The valid values for CMPH are up to 0xFE because the EWM counter never expires when CMPH = 0xFF. The expiration happens only if EWM counter is greater than CMPH.

61.4.1.5.3 Diagram

Bits	7	6	5	4	3	2	1	0
R W	COMPAREH							
Reset	1	1	1	1	1	1	1	1

61.4.1.5.4 Fields

Field	Function
7-0 COMPAREH	COMPAREH To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) maximum refresh time is required.

61.4.1.6 Clock Control Register (CLKCTRL)

61.4.1.6.1 Offset

Register	Offset
CLKCTRL	4h

61.4.1.6.2 Function

This CLKCTRL register is reset to 0x00 after a CPU reset.

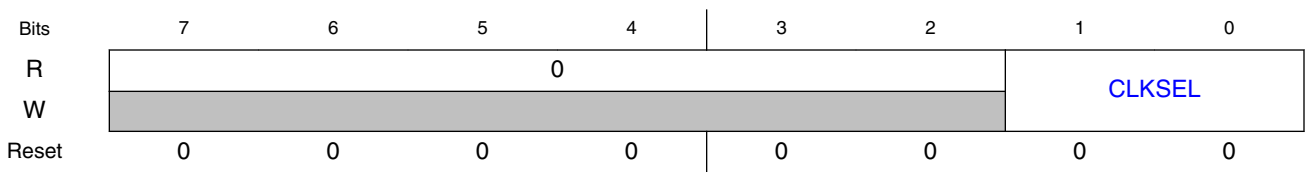
NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

User should select the required low power clock before enabling the EWM.

61.4.1.6.3 Diagram



61.4.1.6.4 Fields

Field	Function
7-2 —	reserved
1-0 CLKSEL	CLKSEL EWM has 4 possible low power clock sources for running EWM counter. One of the clock source can be selected by writing into this field. • 00 - lpo_clk[0] will be selected for running EWM counter. • 01 - lpo_clk[1] will be selected for running EWM counter. • 10 - lpo_clk[2] will be selected for running EWM counter. • 11 - lpo_clk[3] will be selected for running EWM counter.

61.4.1.7 Clock Prescaler Register (CLKPRESCALER)

61.4.1.7.1 Offset

Register	Offset
CLKPRESCALER	5h

61.4.1.7.2 Function

This CLKPRESCALER register is reset to 0x00 after a CPU reset.

NOTE

This register can be written only once after a CPU reset. Writing this register more than once generates a bus transfer error.

NOTE

Write the required prescaler value before enabling the EWM.

NOTE

The implementation of this register is chip-specific. See the Chip Configuration details.

61.4.1.7.3 Diagram

Bits	7	6	5	4	3	2	1	0
R W	CLK_DIV							
Reset	0	0	0	0	0	0	0	0

61.4.1.7.4 Fields

Field	Function
7-0 CLK_DIV	CLK_DIV Selected low power clock source for running the EWM counter can be prescaled as below. • Prescaled clock frequency = low power clock source frequency / (1 + CLK_DIV)

61.5 Functional Description

The following sections describe functional details of the EWM module.

NOTE

When the BUS_CLK is lost, then EWM module doesn't generate the EWM_OUT_b signal and no refresh operation is possible

61.5.1 The EWM_OUT_b Signal

The EWM_OUT_b is a digital output signal used to gate an external circuit (application specific) that controls critical safety functions.

The EWM_OUT_b signal remains deasserted when the EWM is being regularly refreshed by the CPU within the programmable refresh window, indicating that the application code is executed as expected.

The EWM_OUT_b signal is asserted in any of the following conditions:

- The EWM refresh occurs when the counter value is less than CMPL value.
- The EWM counter value reaches the CMPH value, and no EWM refresh has occurred.
- If functionality of EWM_in pin is enabled and EWM_in pin is asserted while refreshing the EWM.
- After any reset (by the virtue of the external pull-down mechanism on the EWM_OUT_b pin)

The EWM_OUT_b is asserted after any reset by the virtue of the external pull-down mechanism on the EWM_OUT_b signal. Then, to deassert the EWM_OUT_b signal, set EWMEN bit in the CTRL register to enable the EWM.

If the EWM_OUT_b signal shares its pad with a digital I/O pin, on reset this actual pad defers to being an input signal. The pad state is controlled by the EWM_OUT_b signal only after the EWM is enabled by the EWMEN bit in the CTRL register.

Note

EWM_OUT_b pad must be in pull down state when EWM functionality is used and when EWM is under Reset.

61.5.2 EWM_OUT_b pin state in low power modes

During Wait, Stop and Power Down modes the EWM_OUT_b pin enters a high-impedance state. A user has the option to control the logic state of the pin using an external pull device or by configuring the internal pull device. When the CPU enters a Run mode from Wait or Stop recovery, the pin resumes its previous state before entering Wait or Stop mode. When the CPU enters Run mode from Power Down, the pin returns to its reset state.

61.5.3 The EWM_in Signal

The EWM_in is a digital input signal for safety status of external safety circuits, that allows an external circuit to control the assertion of the EWM_OUT_b signal. For example, in the application, an external circuit monitors a critical safety function, and if there is fault with safety function, the external circuit can then actively initiate the EWM_OUT_b signal that controls the gating circuit.

The EWM_in signal is ignored if the EWM is disabled, or if INEN bit of CTRL register is cleared, as after any reset.

On enabling the EWM (setting the CTRL[EWMEN] bit) and enabling EWM_in functionality (setting the CTRL[INEN] bit), the EWM_in signal must be in the deasserted state prior to the CPU start refreshing the EWM. This ensures that the EWM_OUT_b stays in the deasserted state; otherwise, the EWM_OUT_b output signal is asserted.

Note

The user must update the CMPH and CMPL registers prior to enabling the EWM. After enabling the EWM, the counter resets to zero, therefore the user shall provide a reasonable time after a power-on reset for the external monitoring circuit to stabilize. The user shall also ensure that the EWM_in pin is deasserted.

61.5.4 EWM Counter

It is an 8-bit ripple counter fed from a clock source that is independent of the peripheral bus clock source. As the preferred time-out is between 1 ms and 100 ms the actual clock source should be in the kHz range.

The counter is reset to zero after the CPU reset, or when EWM refresh action completes, or at counter overflow. The counter value is not accessible to the CPU.

61.5.5 EWM Compare Registers

The compare registers CMPL and CMPH are write-once after a CPU reset and cannot be modified until another CPU reset occurs.

The EWM compare registers are used to create a refresh window to refresh the EWM module.

It is illegal to program CMPL and CMPH with same value. In this case, as soon as counter reaches (CMPL + 1), EWM_OUT_b is asserted.

61.5.6 EWM Refresh Mechanism

Other than the initial configuration of the EWM, the CPU can only access the EWM by the EWM Service Register. The CPU must access the EWM service register with correct write of unique data within the windowed time frame as determined by the CMPL and CMPH registers for correct EWM refresh operation. Therefore, three possible conditions can occur:

Table 61-4. EWM Refresh Mechanisms

Condition	Mechanism
An EWM refresh action completes when: CMPL < Counter < CMPH.	The software behaves as expected and the EWM counter is reset to zero. The EWM_OUT_b output signal remains in the deasserted state if, during the EWM refresh action, the EWM_in input has been in deasserted state..
An EWM refresh action completes when Counter < CMPL	The software refreshes the EWM before the windowed time frame, the counter is reset to zero and the EWM_OUT_b output signal is asserted irrespective of the input EWM_in signal.
Counter value reaches CMPH prior to completion of EWM refresh action.	Software has not refreshed the EWM. The EWM counter is reset to zero and the EWM_OUT_b output signal is asserted irrespective of the input EWM_in signal.

61.5.7 EWM Interrupt

When EWM_OUT_b is asserted, an interrupt request is generated to indicate the assertion of the EWM reset out signal. This interrupt is enabled when CTRL[INTEN] is set. Clearing this bit clears the interrupt request but does not affect EWM_OUT_b. The EWM_OUT_b signal can be deasserted only by forcing a system reset.

61.5.8 Selecting the EWM counter clock

There are four possible low power clock sources for the EWM counter. Select one of the available clock sources by programming CLKCTRL[CLKSEL].

61.5.9 Counter clock prescaler

The EWM counter clock source can be prescaled by a clock divider, by programming CLKPRESCALER[CLK_DIV]. This divided clock is used to run the EWM counter.

NOTE

The divided clock used to run the EWM counter must be no more than half the frequency of the bus clock.

Chapter 62
Low Power Timer/Pulse Width Modulation Module (TPM)

62.1 Chip-specific LPTPM information

Table 62-1. Reference links to related information

Topic	Related module	Reference
Full description	LPTPM	LPTPM
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

62.1.1 TPM

The Timer/PWM Module (TPM) supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The counter, compare and capture registers are clocked by an asynchronous clock that can remain enabled in low power modes. TPM can support global counter bus where one TPM drives the counter bus for the others, provided bit width is the same.

Table 62-2. TPM configuration

Parameter	Description
Name	TPM
Instances	8
Configurable features	• TPM0, 3, 4, 7: 16-bit timer, 6 channels • TPM1, 2, 5, 6: 32-bit timer, 2 channels • TPM0-7: Filter, Combine, Quadrature

Table continues on the next page...

Table 62-2. TPM configuration (continued)

Parameter	Description
	• Global triggers: TPM3 clocked from TPM0, TPM2 clocked from TPM1, TPM7 clocked from TPM4, and TPM6 clocked from TPM5. • TPM Trigger Options will be provided in the TRGMUX channel assignments. See Trigger connections
Interface speed	-
External I/O pins	TPMx: CHx, CLKIN, EXTRG_IN. See the attached IOMUXC spreadsheet for details.

62.1.2 Trigger connections

On i.MX 7ULP, TPM trigger sources get routed through the TRIGMUX module. See the Table 38-2 and Table 38-4 for external/internal trigger connections to this module.

62.2 Introduction

The TPM (Timer/PWM Module) is a 2- to 8-channel timer which supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications.

The counter, compare and capture registers are clocked by an asynchronous clock that can remain enabled in low power modes. An example of using the TPM with the asynchronous DMA is described in AN4631:Using the Asynchronous DMA features of the Kinetis L Series .

62.2.1 Features

The TPM features include:

- TPM clock mode is selectable
  - Can increment on every edge of the asynchronous counter clock
  - Can increment on rising edge of an external clock input synchronized to the asynchronous counter clock
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- TPM includes a 16-bit counter

    - It can be a free-running counter or modulo counter
    - The counting can be up or up-down
- Includes 6 channels that can be configured for input capture, output compare, edge-aligned PWM mode, or center-aligned PWM mode
    - In input capture mode the capture can occur on rising edges, falling edges or both edges
    - In output compare mode the output signal can be set, cleared, pulsed, or toggled on match
    - All channels can be configured for edge-aligned PWM mode or center-aligned PWM mode
- Support the generation of an interrupt and/or DMA request per channel
- Support the generation of an interrupt and/or DMA request when the counter overflows
- Support selectable trigger input to optionally reset or cause the counter to start incrementing.
    - The counter can also optionally stop incrementing on counter overflow
- Support the generation of hardware triggers when the counter overflows and per channel

62.2.2 Modes of operation

During debug mode, the TPM can can be configured to temporarily pause all counting until the core returns to normal user operating mode or to operate normally. When the counter is paused, trigger inputs and input capture events are ignored, and PWM outputs are forced to their initial state.

During doze mode, the TPM can be configured to operate normally or to pause all counting for the duration of doze mode. When the counter is paused, trigger inputs and input capture events are ignored, and PWM outputs are forced to their initial state.

During stop mode, the TPM counter clock can remain functional and the TPM can generate an asynchronous interrupt to exit the MCU from stop mode.

62.2.3 Block diagram

The TPM uses one input/output (I/O) pin per channel, CHn (TPM channel (n)) where n is the channel number.

The following figure shows the TPM structure. The central component of the TPM is the 16-bit counter with programmable final value and its counting can be up or up-down.

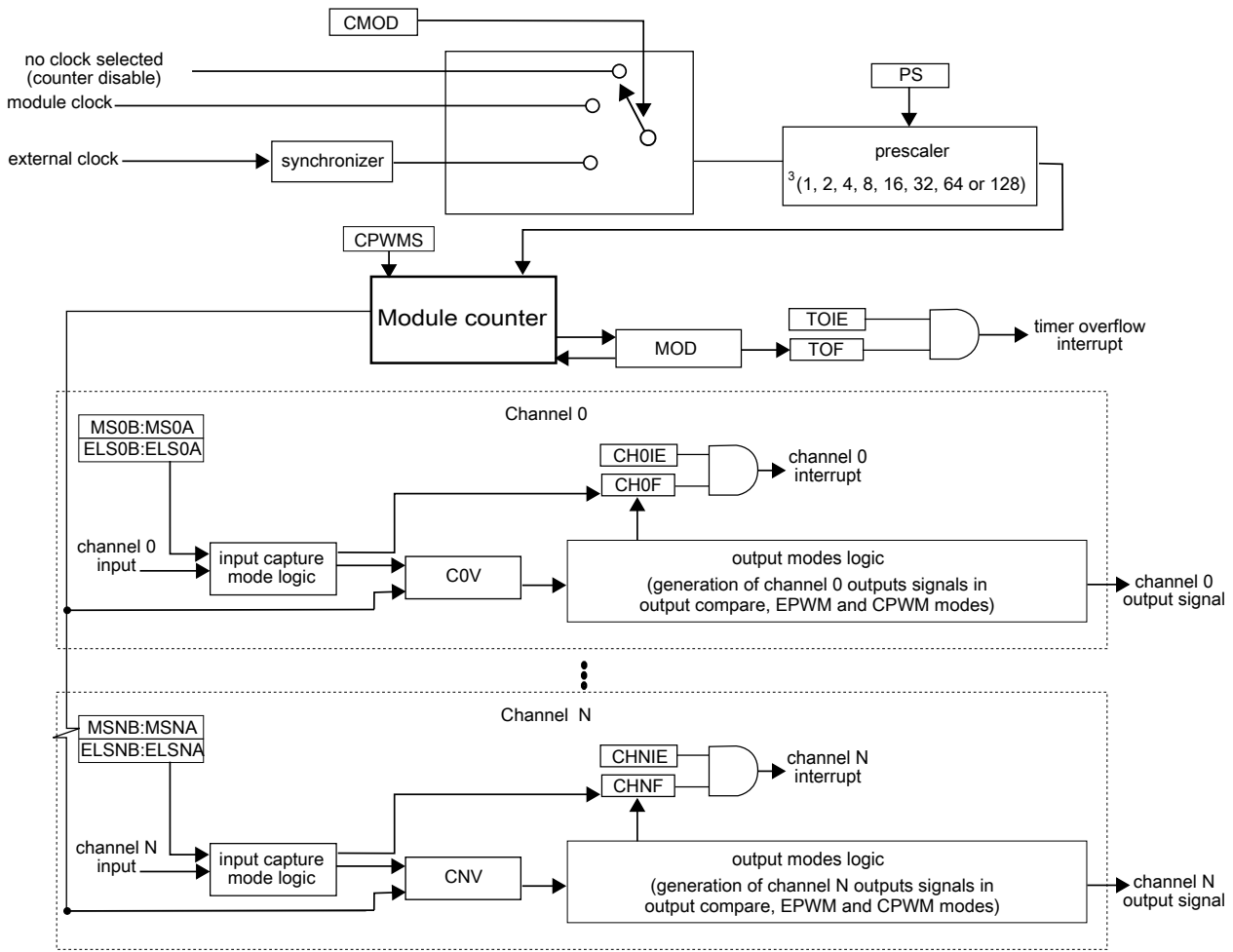


Figure 62-1. TPM block diagram

62.3 TPM Signal Descriptions

Table 62-3 shows the user-accessible signals for the TPM.

Table 62-3. TPM signal descriptions

Signal	Description	I/O
TPM_EXTCLK	External clock. TPM external clock can be selected to increment the counter on every rising edge synchronized to the counter clock.	I

Table continues on the next page...

Table 62-3. TPM signal descriptions (continued)

Signal	Description	I/O
TPM_CHn	TPM channel (n = 5 to 0). A TPM channel pin is configured as output when configured in an output compare or PWM mode and the TPM counter is enabled, otherwise the TPM channel pin is an input.	I/O

62.3.1 TPM_EXTCLK — TPM External Clock

The rising edge of the external input signal is used to increment the TPM counter if selected by CMOD[1:0] bits in the SC register. This input signal must be less than half of the TPM counter clock frequency. The TPM counter prescaler selection and settings are also used when an external input is selected.

62.3.2 TPM_CHn — TPM Channel (n) I/O Pin

Each TPM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.

62.4 Memory Map and Register Definition

This section provides a detailed description of all TPM registers.

Attempting to access a reserved register location in the TPM memory map will generate a bus error.

62.4.1 TPM register descriptions

62.4.1.1 TPM Memory map

TPM0 base address: 4103_0000h

TPM3 base address: 410A_9000h

TPM4 base address: 4025_0000h

TPM7 base address: 40A2_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0500_0007h
4h	Parameter Register (PARAM)	32	RO	0010_0406h
8h	TPM Global Register (GLOBAL)	32	RW	0000_0000h
10h	Status and Control (SC)	32	RW	0000_0000h
14h	Counter (CNT)	32	RW	0000_0000h
18h	Modulo (MOD)	32	RW	0000_FFFFh
1Ch	Capture and Compare Status (STATUS)	32	W1C	0000_0000h
20h	Channel (n) Status and Control (C0SC)	32	RW	0000_0000h
24h	Channel (n) Value (C0V)	32	RW	0000_0000h
28h	Channel (n) Status and Control (C1SC)	32	RW	0000_0000h
2Ch	Channel (n) Value (C1V)	32	RW	0000_0000h
30h	Channel (n) Status and Control (C2SC)	32	RW	0000_0000h
34h	Channel (n) Value (C2V)	32	RW	0000_0000h
38h	Channel (n) Status and Control (C3SC)	32	RW	0000_0000h
3Ch	Channel (n) Value (C3V)	32	RW	0000_0000h
40h	Channel (n) Status and Control (C4SC)	32	RW	0000_0000h
44h	Channel (n) Value (C4V)	32	RW	0000_0000h
48h	Channel (n) Status and Control (C5SC)	32	RW	0000_0000h
4Ch	Channel (n) Value (C5V)	32	RW	0000_0000h
64h	Combine Channel Register (COMBINE)	32	RW	0000_0000h
6Ch	Channel Trigger (TRIG)	32	RW	0000_0000h
70h	Channel Polarity (POL)	32	RW	0000_0000h
78h	Filter Control (FILTER)	32	RW	0000_0000h
80h	Quadrature Decoder Control and Status (QDCTRL)	32	RW	0000_0000h
84h	Configuration (CONF)	32	RW	0000_0000h

62.4.1.2 Version ID Register (VERID)

62.4.1.2.1 Offset

Register	Offset
VERID	0h

62.4.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

62.4.1.2.3 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number 0000000000000001b - Standard feature set. 0000000000000011b - Standard feature set with Filter and Combine registers implemented. 0000000000000111b - Standard feature set with Filter, Combine and Quadrature registers implemented.

62.4.1.3 Parameter Register (PARAM)

62.4.1.3.1 Offset

Register	Offset
PARAM	4h

62.4.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								WIDTH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRIG								CHAN							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0

62.4.1.3.3 Fields

Field	Function
31-24 —	Reserved
23-16 WIDTH	Counter Width Width of the counter and timer channels.
15-8 TRIG	Trigger Count Number of trigger inputs implemented.
7-0 CHAN	Channel Count Number of timer channels implemented.

62.4.1.4 TPM Global Register (GLOBAL)

62.4.1.4.1 Offset

Register	Offset
GLOBAL	8h

62.4.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														RST	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

62.4.1.4.3 Fields

Field	Function
31-2 —	Reserved
1 RST	Software Reset Reset all internal logic and registers, except the Global Register. Remains set until cleared by software. 0b - Module is not reset. 1b - Module is reset.
0 —	Reserved

62.4.1.5 Status and Control (SC)

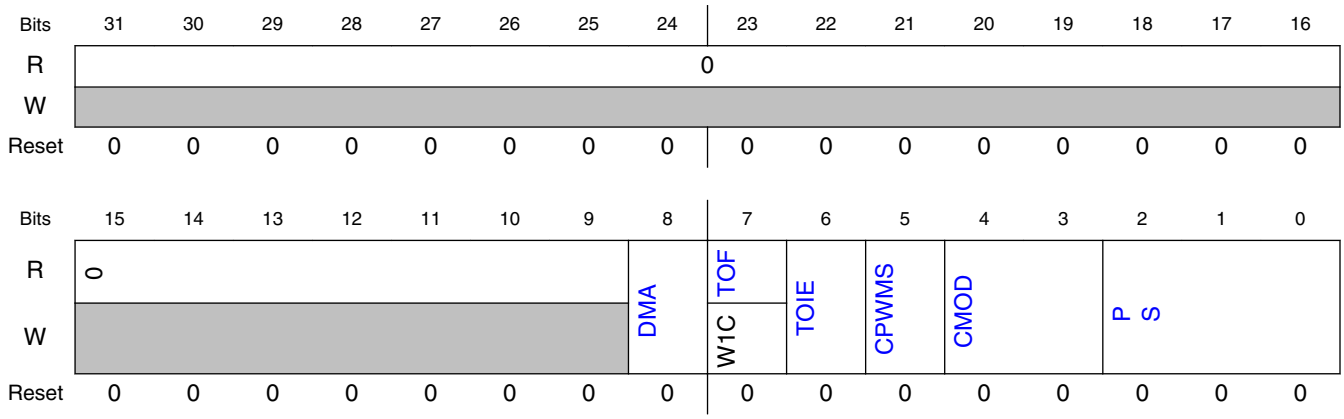
62.4.1.5.1 Offset

Register	Offset
SC	10h

62.4.1.5.2 Function

SC contains the overflow status flag and control bits used to configure the interrupt enable, module configuration and prescaler factor. These controls relate to all channels within this module.

62.4.1.5.3 Diagram



62.4.1.5.4 Fields

Field	Function
31-9 —	Reserved
8 DMA	DMA Enable Enables DMA transfers for the overflow flag. 0b - Disables DMA transfers. 1b - Enables DMA transfers.
7 TOF	Timer Overflow Flag Set by hardware when the TPM counter equals the value in the MOD register and increments. Writing a 1 to TOF clears it. Writing a 0 to TOF has no effect. If another TPM overflow occurs between the flag setting and the flag clearing, the write operation has no effect; therefore, TOF remains set indicating another overflow has occurred. In this case a TOF interrupt request is not lost due to a delay in clearing the previous TOF. 0b - TPM counter has not overflowed. 1b - TPM counter has overflowed.
6 TOIE	Timer Overflow Interrupt Enable Enables TPM overflow interrupts. 0b - Disable TOF interrupts. Use software polling or DMA request. 1b - Enable TOF interrupts. An interrupt is generated when TOF equals one.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the TPM to operate in up-down counting mode. This field is write protected. It can be written only when the counter is disabled. 0b - TPM counter operates in up counting mode. 1b - TPM counter operates in up-down counting mode.
4-3 CMOD	Clock Mode Selection Selects the TPM counter clock modes. When disabling the counter, this field remain set until acknolwedged in the TPM clock domain. 00b - TPM counter is disabled

Table continues on the next page...

Field	Function
	01b - TPM counter increments on every TPM counter clock 10b - TPM counter increments on rising edge of TPM_EXTCLK synchronized to the TPM counter clock 11b - TPM counter increments on rising edge of the selected external input trigger.
2-0 PS	Prescale Factor Selection Selects one of 8 division factors for the clock mode selected by CMOD. This field is write protected. It can be written only when the counter is disabled. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

62.4.1.6 Counter (CNT)

62.4.1.6.1 Offset

Register	Offset
CNT	14h

62.4.1.6.2 Function

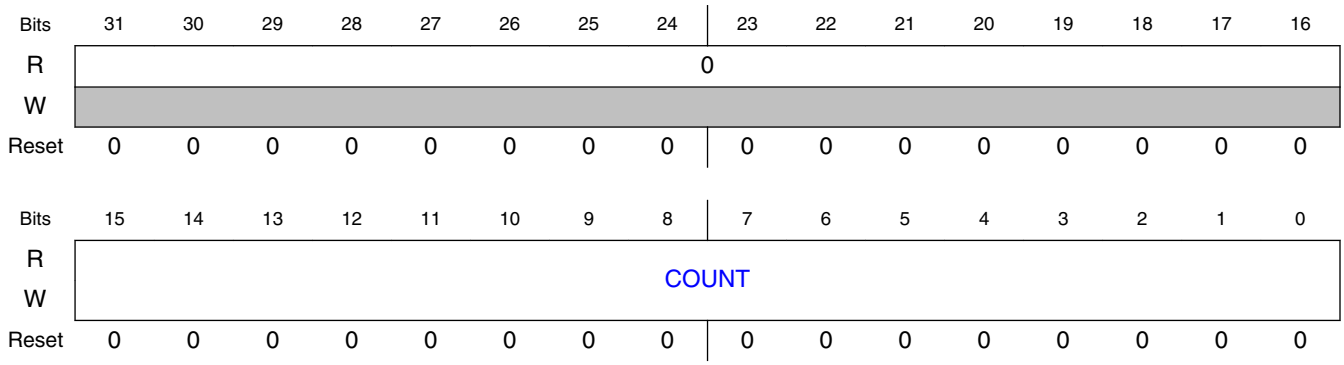
The CNT register contains the TPM counter value.

Reset clears the CNT register. Writing any value to COUNT will trigger a reload of the counter, this will force PWM outputs to their reload state..

When debug is active, the TPM counter does not increment unless configured otherwise.

Reading the CNT register adds two wait states to the register access due to synchronization delays.

62.4.1.6.3 Diagram



62.4.1.6.4 Fields

Field	Function
31-16 —	Reserved
15-0 COUNT	Counter value

62.4.1.7 Modulo (MOD)

62.4.1.7.1 Offset

Register	Offset
MOD	18h

62.4.1.7.2 Function

The Modulo register contains the modulo value for the TPM counter. When the TPM counter reaches the modulo value and increments, the overflow flag (TOF) is set and the next value of TPM counter depends on the selected counting method (see Counter).

Writing to the MOD register latches the value into a buffer. The MOD register is updated with the value of its write buffer according to MOD Register Update . Additional writes to the MOD write buffer are ignored until the register has been updated.

It is recommended to initialize the TPM counter (write to CNT) before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

62.4.1.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MOD															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

62.4.1.7.4 Fields

Field	Function
31-16 —	Reserved
15-0 MOD	Modulo value This field must be written with single 16-bit or 32-bit access.

62.4.1.8 Capture and Compare Status (STATUS)

62.4.1.8.1 Offset

Register	Offset
STATUS	1Ch

62.4.1.8.2 Function

The STATUS register contains a copy of the status flag, CnSC[CHnF] for each TPM channel, as well as SC[TOF], for software convenience.

Each CHnF bit in STATUS is a mirror of CHnF bit in CnSC. All CHnF bits can be checked using only one read of STATUS. All CHnF bits can be cleared by writing all ones to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. Writing a 1 to CHF clears it. Writing a 0 to CHF has no effect.

If another event occurs between the flag setting and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

62.4.1.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							TOF	0		CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W								W1C			W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

62.4.1.8.4 Fields

Field	Function
31-9 —	Reserved
8 TOF	Timer Overflow Flag See register description 0b - TPM counter has not overflowed. 1b - TPM counter has overflowed.
7-6 —	Reserved
5 CH5F	Channel 5 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
4	Channel 4 Flag

Table continues on the next page...

Field	Function
CH4F	See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
3 CH3F	Channel 3 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
2 CH2F	Channel 2 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
1 CH1F	Channel 1 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
0 CH0F	Channel 0 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.

62.4.1.9 Channel (n) Status and Control (C0SC - C5SC)

62.4.1.9.1 Offset

Register	Offset
C0SC	20h
C1SC	28h
C2SC	30h
C3SC	38h
C4SC	40h
C5SC	48h

62.4.1.9.2 Function

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

When switching from one channel mode to a different channel mode, the channel must first be disabled and this must be acknowledged in the TPM counter clock domain.

Table 62-4. Mode, Edge, and Level Selection

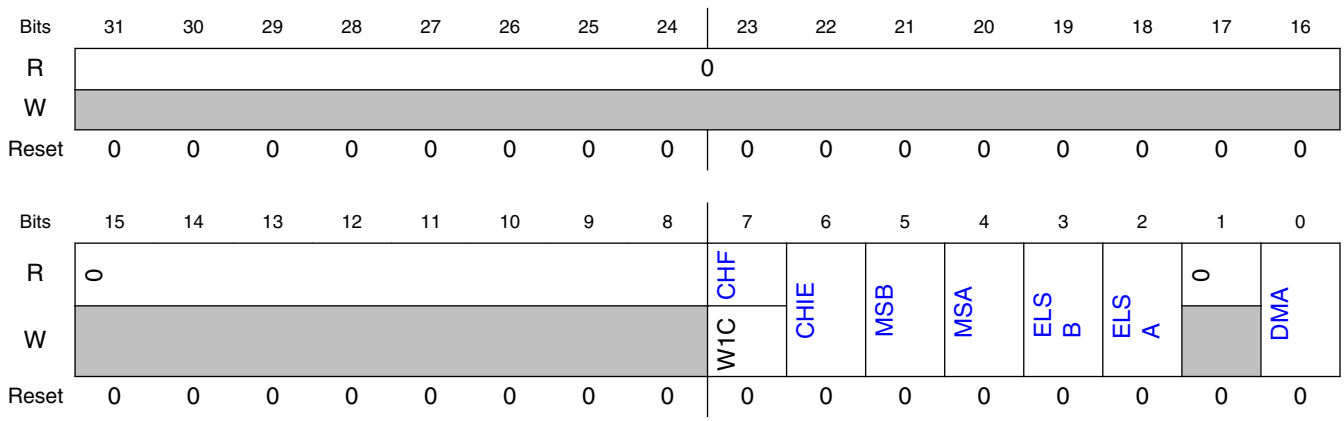
CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
0	00	00	None	Channel disabled
0	01	00	Software compare	Pin not used for TPM
0	00	01	Input capture	Capture on Rising Edge Only
		10		Capture on Falling Edge Only
		11		Capture on Rising or Falling Edge
	01	01	Output compare	Toggle Output on match
		10		Clear Output on match
		11		Set Output on match
	10	10	Edge-aligned PWM	High-true pulses (clear Output on counter match, set Output on counter reload, set Output when counter first enabled or paused)
		00		Reserved
		01		Low-true pulses (set Output on counter match, clear Output on counter reload, clear Output when counter first enabled or paused)
		11		Reserved
	11	10	Output compare	Pulse Output low on match
		01		Pulse Output high on match
1	10	10	Center-aligned PWM	High-true pulses (clear Output on counter match-up, set Output on counter match-down, set Output on counter reload or when counter first enabled or paused)
		00		Reserved
		01		Low-true pulses (set Output on match-up, clear Output on match-down, clear Output on

Table continues on the next page...

Table 62-4. Mode, Edge, and Level Selection (continued)

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
				counter reload or when counter first enabled or paused))
		11		Reserved

62.4.1.9.3 Diagram



62.4.1.9.4 Fields

Field	Function
31-8 —	Reserved
7 CHF	Channel Flag Set by hardware when an event occurs on the channel. CHF is cleared by writing a 1 to the CHF bit. Writing a 0 to CHF has no effect. If another event occurs between the CHF sets and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the delay in clearing the previous CHF. 0b - No channel event has occurred. 1b - A channel event has occurred.
6 CHIE	Channel Interrupt Enable Enables channel interrupts. 0b - Disable channel interrupts. 1b - Enable channel interrupts.
5 MSB	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.

Table continues on the next page...

Field	Function
4 MSA	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
3 ELSB	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.ph>
2 ELSA	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
1 —	Reserved
0 DMA	DMA Enable Enables DMA transfers for the channel. 0b - Disable DMA transfers. 1b - Enable DMA transfers.

62.4.1.10 Channel (n) Value (C0V - C5V)

62.4.1.10.1 Offset

Register	Offset
C0V	24h
C1V	2Ch
C2V	34h
C3V	3Ch
C4V	44h
C5V	4Ch

62.4.1.10.2 Function

These registers contain the captured TPM counter value for the input modes or the match value for the output modes.

In input capture mode, any write to a CnV register is ignored.

In compare modes, writing to a CnV register latches the value into a buffer. A CnV register is updated with the value of its write buffer according to CnV Register Update . Additional writes to the CnV write buffer are ignored until the register has been updated.

62.4.1.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

62.4.1.10.4 Fields

Field	Function
31-16 —	Reserved
15-0 VAL	Channel Value Captured TPM counter value of the input modes or the match value for the output modes. This field must be written with single 16-bit or 32-bit access.

62.4.1.11 Combine Channel Register (COMBINE)

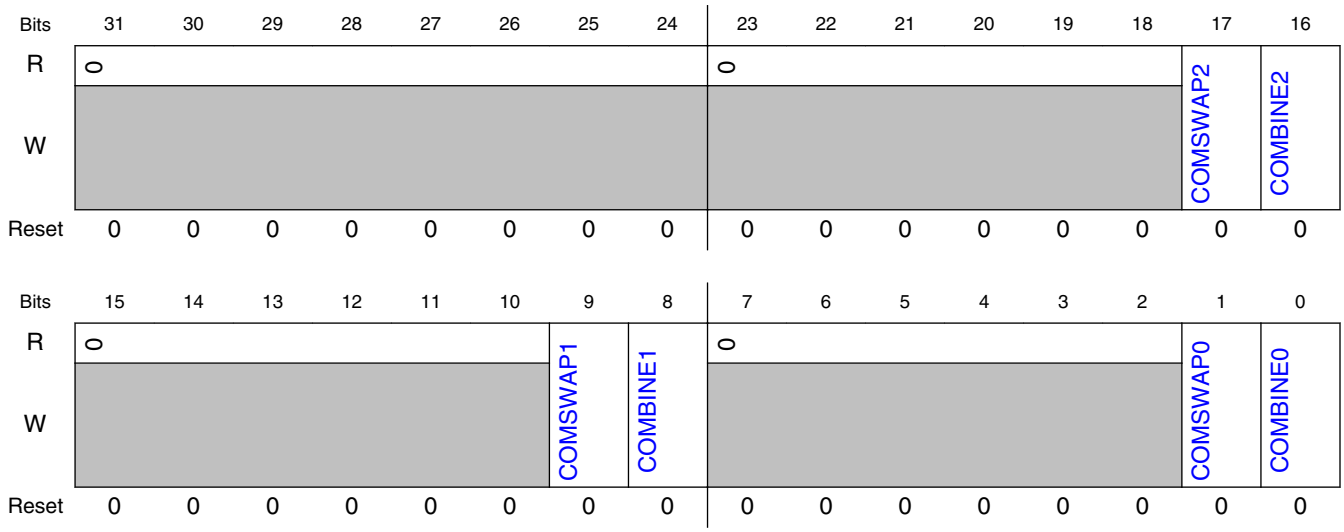
62.4.1.11.1 Offset

Register	Offset
COMBINE	64h

62.4.1.11.2 Function

This register contains the control bits used to configure the combine channel modes for each pair of channels (n) and (n+1), where n is all the even numbered channels.

62.4.1.11.3 Diagram



62.4.1.11.4 Fields

Field	Function
31-24 —	Reserved
23-18 —	Reserved
17 COMSWAP2	Combine Channels 4 and 5 Swap When set in combine mode, the even channel is used for the input capture and 1st compare, the odd channel is used for the 2nd compare. 0b - Even channel is used for input capture and 1st compare. 1b - Odd channel is used for input capture and 1st compare.
16 COMBINE2	Combine Channels 4 and 5 Enables the combine feature for channels 2 and 3. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0b - Channels 4 and 5 are independent. 1b - Channels 4 and 5 are combined.
15-10 —	Reserved
9 COMSWAP1	Combine Channels 2 and 3 Swap When set in combine mode, the odd channel is used for the input capture and 1st compare, the even channel is used for the 2nd compare. 0b - Even channel is used for input capture and 1st compare. 1b - Odd channel is used for input capture and 1st compare.
8 COMBINE1	Combine Channels 2 and 3

Table continues on the next page...

Field	Function
	Enables the combine feature for channels 2 and 3. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0b - Channels 2 and 3 are independent. 1b - Channels 2 and 3 are combined.
7-2 —	Reserved
1 COMSWAP0	Combine Channel 0 and 1 Swap When set in combine mode, the even channel is used for the input capture and 1st compare, the odd channel is used for the 2nd compare. 0b - Even channel is used for input capture and 1st compare. 1b - Odd channel is used for input capture and 1st compare.
0 COMBINE0	Combine Channels 0 and 1 Enables the combine feature for channels 0 and 1. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0b - Channels 0 and 1 are independent. 1b - Channels 0 and 1 are combined.

62.4.1.12 Channel Trigger (TRIG)

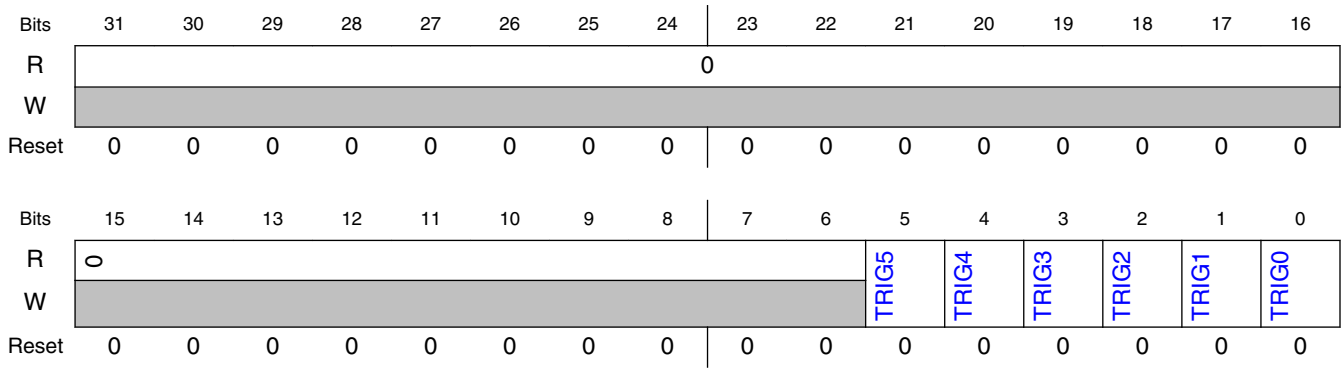
62.4.1.12.1 Offset

Register	Offset
TRIG	6Ch

62.4.1.12.2 Function

In input capture mode, configures the trigger input that is used by the channel to capture the counter value. In output compare or PWM mode, configures the trigger input used to modulate the channel output. When modulating the output, the output is forced to zero whenever the trigger is not asserted. Note that the even numbered channels share the first input trigger source and the odd numbered channels share the second input trigger source.

62.4.1.12.3 Diagram



62.4.1.12.4 Fields

Field	Function
31-6 —	Reserved
5 TRIG5	Channel 5 Trigger 0b - No effect. 1b - Configures trigger input 1 to be used by channel 5.
4 TRIG4	Channel 4 Trigger 0b - No effect. 1b - Configures trigger input 0 to be used by channel 4.
3 TRIG3	Channel 3 Trigger 0b - No effect. 1b - Configures trigger input 1 to be used by channel 3.
2 TRIG2	Channel 2 Trigger 0b - No effect. 1b - Configures trigger input 0 to be used by channel 2.
1 TRIG1	Channel 1 Trigger 0b - No effect. 1b - Configures trigger input 1 to be used by channel 1.
0 TRIG0	Channel 0 Trigger 0b - No effect. 1b - Configures trigger input 0 to be used by channel 0.

62.4.1.13 Channel Polarity (POL)

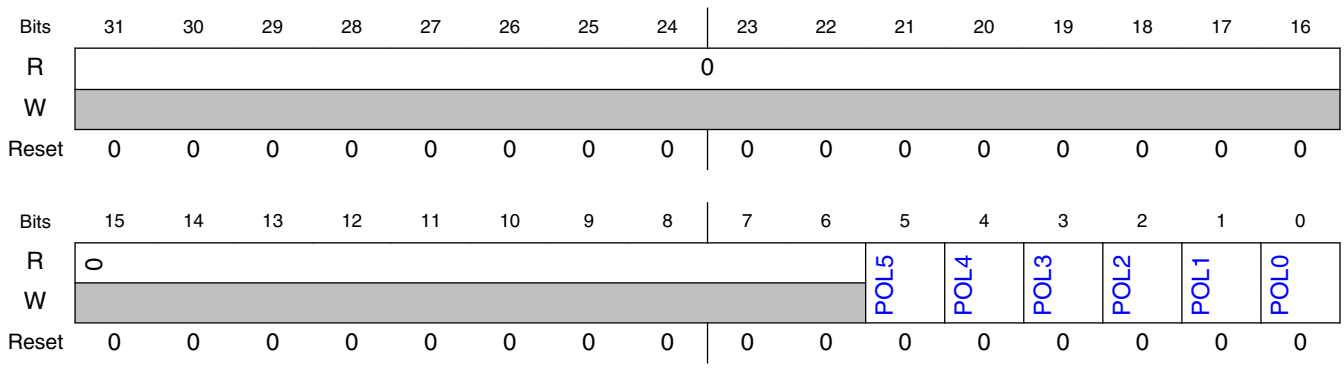
62.4.1.13.1 Offset

Register	Offset
POL	70h

62.4.1.13.2 Function

This register defines the input and output polarity of each of the channels.

62.4.1.13.3 Diagram



62.4.1.13.4 Fields

Field	Function
31-6 —	Reserved
5 POL5	Channel 5 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
4 POL4	Channel 4 Polarity 0b - The channel polarity is active high 1b - The channel polarity is active low.
3 POL3	Channel 3 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
2 POL2	Channel 2 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
1 POL1	Channel 1 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
0 POL0	Channel 0 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.

62.4.1.14 Filter Control (FILTER)

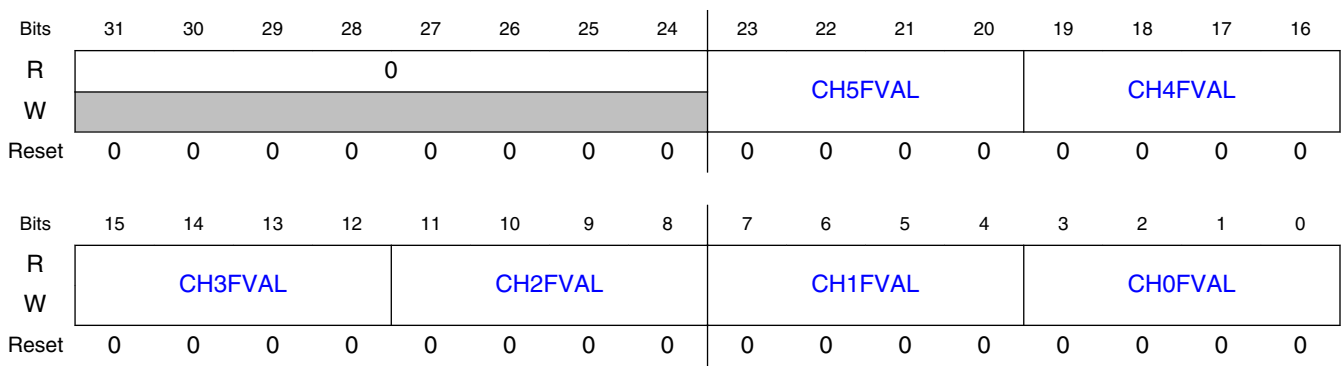
62.4.1.14.1 Offset

Register	Offset
FILTER	78h

62.4.1.14.2 Function

This register selects the filter value of the channel inputs, and an additional output delay value for the channel outputs. In PWM combine modes, the filter can be used to implement deadtime insertion.

62.4.1.14.3 Diagram



62.4.1.14.4 Fields

Field	Function
31-24 —	Reserved
23-20 CH5FVAL	Channel 5 Filter Value Sets the filter value for the channel 5 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH5FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 5 in PWM modes. Deadtime insertion is disabled when CH5FVAL is zero, otherwise deadtime insertion for channel 5 is configured as (CH5FVAL * 4) clock cycles.
19-16 CH4FVAL	Channel 4 Filter Value Sets the filter value for the channel 4 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH4FVAL * 4) clock cycles.

Table continues on the next page...

Field	Function
	Sets the deadtime insertion time for channel 4 in PWM modes. Deadtime insertion is disabled when CH4FVAL is zero, otherwise deadtime insertion for channel 4 is configured as (CH4FVAL * 4) clock cycles.
15-12 CH3FVAL	Channel 3 Filter Value Sets the filter value for the channel 3 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH3FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 3 in PWM modes. Deadtime insertion is disabled when CH3FVAL is zero, otherwise deadtime insertion for channel 3 is configured as (CH3FVAL * 4) clock cycles.
11-8 CH2FVAL	Channel 2 Filter Value Sets the filter value for the channel 2 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH2FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 2 in PWM modes. Deadtime insertion is disabled when CH2FVAL is zero, otherwise deadtime insertion for channel 2 is configured as (CH2FVAL * 4) clock cycles.
7-4 CH1FVAL	Channel 1 Filter Value Sets the filter value for the channel 1 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH1FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 1 in PWM modes. Deadtime insertion is disabled when CH1FVAL is zero, otherwise deadtime insertion for channel 1 is configured as (CH1FVAL * 4) clock cycles.
3-0 CH0FVAL	Channel 0 Filter Value Sets the filter value for the channel 0 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH0FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 0 in PWM modes. Deadtime insertion is disabled when CH0FVAL is zero, otherwise deadtime insertion for channel 0 is configured as (CH0FVAL * 4) clock cycles.

62.4.1.15 Quadrature Decoder Control and Status (QDCTRL)

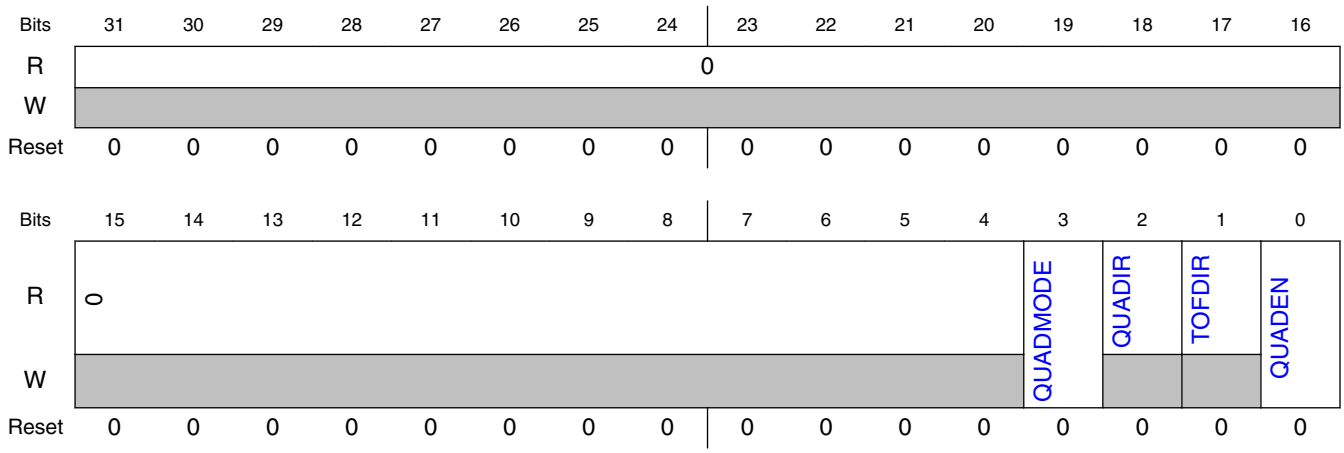
62.4.1.15.1 Offset

Register	Offset
QDCTRL	80h

62.4.1.15.2 Function

This register has the control and status bits for the quadrature decoder mode.

62.4.1.15.3 Diagram



62.4.1.15.4 Fields

Field	Function
31-4 —	Reserved
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the quadrature decoder mode. 0b - Phase encoding mode. 1b - Count and direction encoding mode.
2 QUADIR	Counter Direction in Quadrature Decode Mode Indicates the counting direction. 0b - Counter direction is decreasing (counter decrement). 1b - Counter direction is increasing (counter increment).
1 TOFDIR	TOFDIR Indicates if the TOF bit was set on the top or the bottom of counting. 0b - TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (zero) to its maximum value (MOD register). 1b - TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (zero).
0 QUADEN	QUADEN Enables the quadrature decoder mode. In this mode, the channel 0 and channel 1 inputs control the TPM counter direction and can only be used for software compare. The quadrature decoder mode has precedence over the other modes. 0b - Quadrature decoder mode is disabled. 1b - Quadrature decoder mode is enabled.

62.4.1.16 Configuration (CONF)

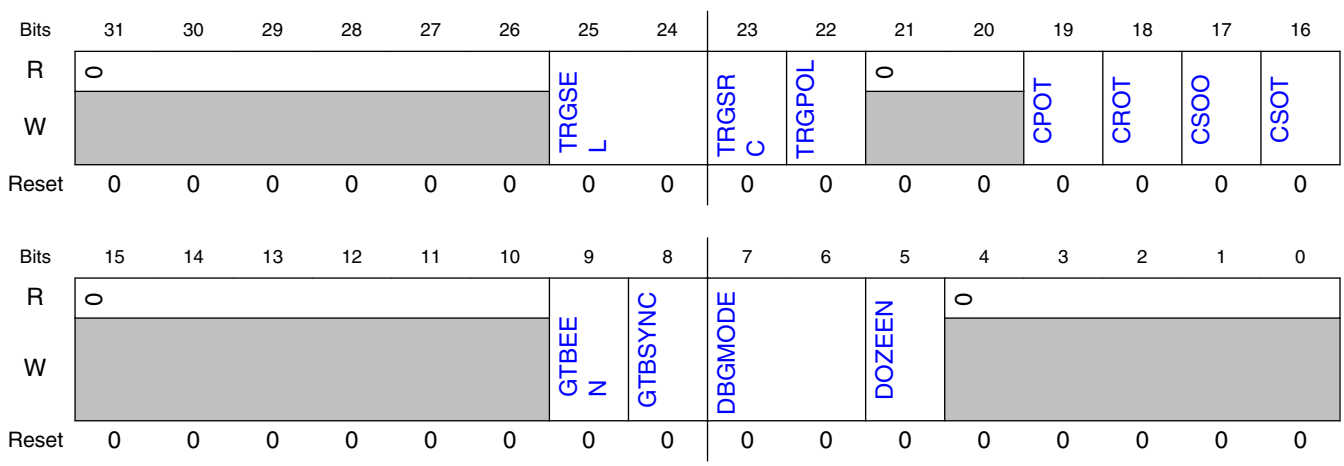
62.4.1.16.1 Offset

Register	Offset
CONF	84h

62.4.1.16.2 Function

This register selects the behavior in debug and wait modes and the use of an external global time base.

62.4.1.16.3 Diagram



62.4.1.16.4 Fields

Field	Function
31-26 —	Reserved
25-24 TRGSEL	Trigger Select Selects the input trigger to use for starting, reloading and/or pausing the counter. The source of the trigger (external or internal to the TPM) is configured by the TRGSRC field. This field should only be changed when the TPM counter is disabled. Refer to chip configuration section for available external trigger options. The available internal trigger sources are listed below. 01b - Channel 0 pin input capture 10b - Channel 1 pin input capture

Table continues on the next page...

Field	Function
	11b - Channel 0 or Channel 1 pin input capture
23 TRGSRC	Trigger Source Selects between internal (channel pin input capture) or external trigger sources. When selecting an internal trigger, the channel selected should be configured for input capture. Only a rising edge input capture can be used to initially start the counter using the CSOT configuration; either rising edge or falling edge input capture can be used to reload the counter using the CROT configuration; and the state of the channel input pin is used to pause the counter using the CPOT configuration. The channel polarity register can be used to invert the polarity of the channel input pins. This field should only be changed when the TPM counter is disabled. 0b - Trigger source selected by TRGSEL is external. 1b - Trigger source selected by TRGSEL is internal (channel pin input capture).
22 TRGPOL	Trigger Polarity Selects the polarity of the external trigger source. This field should only be changed when the TPM counter is disabled. 0b - Trigger is active high. 1b - Trigger is active low.
21-20 —	Reserved
19 CPOT	Counter Pause On Trigger When enabled, the counter will pause incrementing while the trigger remains asserted (level sensitive). While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled.
18 CROT	Counter Reload On Trigger When set, the TPM counter will reload with zero (and set PWM outputs to their reload state) when a rising edge is detected on the selected trigger input. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled. 0b - Counter is not reloaded due to a rising edge on the selected input trigger 1b - Counter is reloaded when a rising edge is detected on the selected input trigger
17 CSOO	Counter Stop On Overflow When set, the TPM counter will pause incrementing once the counter equals the MOD value and increments (this also sets the TOF). Reloading the counter with 0 due to writing to the counter register or due to a trigger input does not cause the counter to stop incrementing. Once the counter has stopped incrementing, the counter will not start incrementing unless it is disabled and then enabled again, or a rising edge on the selected trigger input is detected when CSOT is set. While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state. This field should only be changed when the TPM counter is disabled. 0b - TPM counter continues incrementing or decrementing after overflow 1b - TPM counter stops incrementing or decrementing after overflow.
16 CSOT	Counter Start on Trigger When set, the TPM counter will pause incrementing when it is enabled until a rising edge on the selected trigger input is detected. If the TPM counter is paused due to an overflow, a rising edge on the selected trigger input will also cause the TPM counter to start incrementing again. While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state.

Table continues on the next page...

Field	Function
	The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled. 0b - TPM counter starts to increment immediately, once it is enabled. 1b - TPM counter only starts to increment when it a rising edge on the selected input trigger is detected, after it has been enabled or after it has stopped due to overflow.
15-10 —	Reserved
9 GTBEEN	Global time base enable Configures the TPM to use an externally generated global time base counter. When an externally generated timebase is used, the internal TPM counter is not used by the channels but can be used to generate a periodic interruptor DMA request using the Modulo register and timer overflow flag. 0b - All channels use the internally generated TPM counter as their timebase 1b - All channels use an externally generated global timebase as their timebase
8 GTBSYNC	Global Time Base Synchronization When enabled, the TPM counter is synchronized to the global time base. It uses the global timebase enable, trigger, paused state and overflow to ensure the TPM counter starts incrementing at the same time as the global timebase, stops incrementing at the same time as the global timebase and is reset at the same time as the global timebase. This field should only be changed when the TPM counter is disabled. 0b - Global timebase synchronization disabled. 1b - Global timebase synchronization enabled.
7-6 DBGMODE	Debug Mode Configures the TPM behavior in debug mode. All other configurations are reserved. 00b - TPM counter is paused and does not increment during debug mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their default state. 11b - TPM counter continues in debug mode.
5 DOZEEN	Doze Enable Configures the TPM behavior in wait mode. 0b - Internal TPM counter continues in Doze mode. 1b - Internal TPM counter is paused and does not increment during Doze mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their default state.
4-0 —	Reserved

62.4.2 TPM register descriptions

62.4.2.1 TPM Memory map

TPM1 base address: 4103_1000h

TPM2 base address: 410A_8000h

TPM5 base address: 4026_0000h

TPM6 base address: 40A1_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0500_0007h
4h	Parameter Register (PARAM)	32	RO	0020_0402h
8h	TPM Global Register (GLOBAL)	32	RW	0000_0000h
10h	Status and Control (SC)	32	RW	0000_0000h
14h	Counter (CNT)	32	RW	0000_0000h
18h	Modulo (MOD)	32	RW	0000_FFFFh
1Ch	Capture and Compare Status (STATUS)	32	W1C	0000_0000h
20h	Channel (n) Status and Control (C0SC)	32	RW	0000_0000h
24h	Channel (n) Value (C0V)	32	RW	0000_0000h
28h	Channel (n) Status and Control (C1SC)	32	RW	0000_0000h
2Ch	Channel (n) Value (C1V)	32	RW	0000_0000h
64h	Combine Channel Register (COMBINE)	32	RW	0000_0000h
6Ch	Channel Trigger (TRIG)	32	RW	0000_0000h
70h	Channel Polarity (POL)	32	RW	0000_0000h
78h	Filter Control (FILTER)	32	RW	0000_0000h
80h	Quadrature Decoder Control and Status (QDCTRL)	32	RW	0000_0000h
84h	Configuration (CONF)	32	RW	0000_0000h

62.4.2.2 Version ID Register (VERID)

62.4.2.2.1 Offset

Register	Offset
VERID	0h

62.4.2.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

62.4.2.2.3 Fields

Field	Function
31-24 MAJOR	Major Version Number This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number 0000000000000001b - Standard feature set. 0000000000000011b - Standard feature set with Filter and Combine registers implemented. 0000000000000111b - Standard feature set with Filter, Combine and Quadrature registers implemented.

62.4.2.3 Parameter Register (PARAM)

62.4.2.3.1 Offset

Register	Offset
PARAM	4h

62.4.2.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								WIDTH							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRIG								CHAN							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0

62.4.2.3.3 Fields

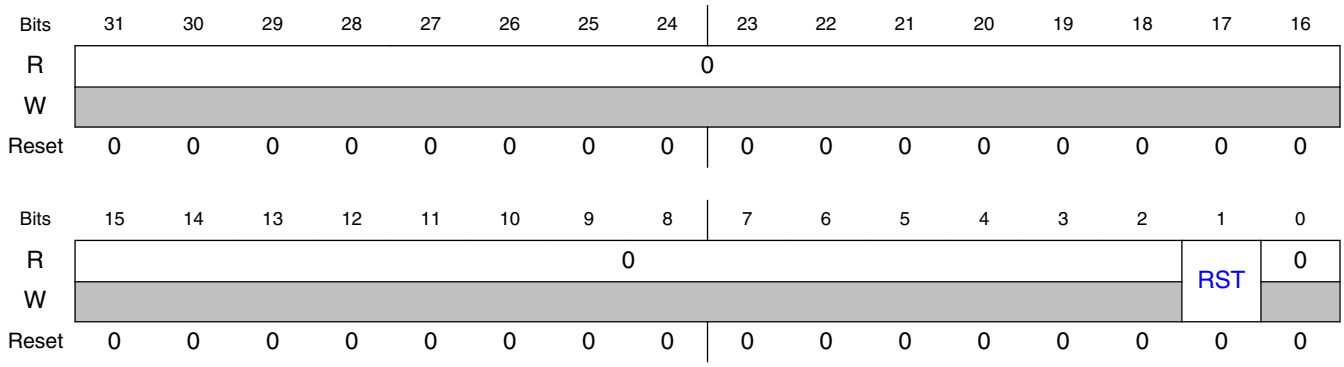
Field	Function
31-24 —	Reserved
23-16 WIDTH	Counter Width Width of the counter and timer channels.
15-8 TRIG	Trigger Count Number of trigger inputs implemented.
7-0 CHAN	Channel Count Number of timer channels implemented.

62.4.2.4 TPM Global Register (GLOBAL)

62.4.2.4.1 Offset

Register	Offset
GLOBAL	8h

62.4.2.4.2 Diagram



62.4.2.4.3 Fields

Field	Function
31-2 —	Reserved
1 RST	Software Reset Reset all internal logic and registers, except the Global Register. Remains set until cleared by software. 0b - Module is not reset. 1b - Module is reset.
0 —	Reserved

62.4.2.5 Status and Control (SC)

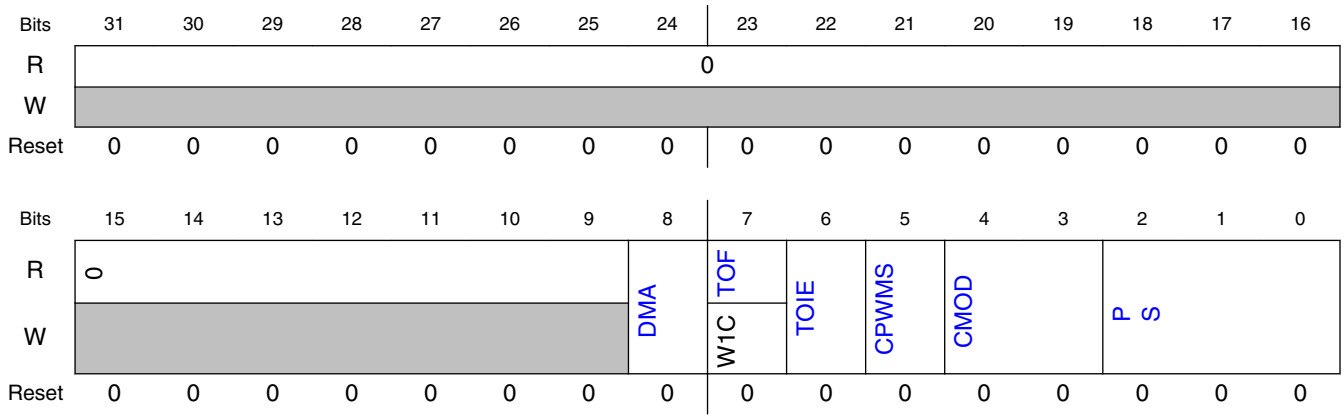
62.4.2.5.1 Offset

Register	Offset
SC	10h

62.4.2.5.2 Function

SC contains the overflow status flag and control bits used to configure the interrupt enable, module configuration and prescaler factor. These controls relate to all channels within this module.

62.4.2.5.3 Diagram



62.4.2.5.4 Fields

Field	Function
31-9 —	Reserved
8 DMA	DMA Enable Enables DMA transfers for the overflow flag. 0b - Disables DMA transfers. 1b - Enables DMA transfers.
7 TOF	Timer Overflow Flag Set by hardware when the TPM counter equals the value in the MOD register and increments. Writing a 1 to TOF clears it. Writing a 0 to TOF has no effect. If another TPM overflow occurs between the flag setting and the flag clearing, the write operation has no effect; therefore, TOF remains set indicating another overflow has occurred. In this case a TOF interrupt request is not lost due to a delay in clearing the previous TOF. 0b - TPM counter has not overflowed. 1b - TPM counter has overflowed.
6 TOIE	Timer Overflow Interrupt Enable Enables TPM overflow interrupts. 0b - Disable TOF interrupts. Use software polling or DMA request. 1b - Enable TOF interrupts. An interrupt is generated when TOF equals one.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the TPM to operate in up-down counting mode. This field is write protected. It can be written only when the counter is disabled. 0b - TPM counter operates in up counting mode. 1b - TPM counter operates in up-down counting mode.
4-3 CMOD	Clock Mode Selection Selects the TPM counter clock modes. When disabling the counter, this field remain set until acknolwedged in the TPM clock domain. 00b - TPM counter is disabled

Table continues on the next page...

Field	Function
	01b - TPM counter increments on every TPM counter clock 10b - TPM counter increments on rising edge of TPM_EXTCLK synchronized to the TPM counter clock 11b - TPM counter increments on rising edge of the selected external input trigger.
2-0 PS	Prescale Factor Selection Selects one of 8 division factors for the clock mode selected by CMOD. This field is write protected. It can be written only when the counter is disabled. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

62.4.2.6 Counter (CNT)

62.4.2.6.1 Offset

Register	Offset
CNT	14h

62.4.2.6.2 Function

The CNT register contains the TPM counter value.

Reset clears the CNT register. Writing any value to COUNT will trigger a reload of the counter, this will force PWM outputs to their reload state..

When debug is active, the TPM counter does not increment unless configured otherwise.

Reading the CNT register adds two wait states to the register access due to synchronization delays.

62.4.2.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

62.4.2.6.4 Fields

Field	Function
31-0 COUNT	Counter value

62.4.2.7 Modulo (MOD)

62.4.2.7.1 Offset

Register	Offset
MOD	18h

62.4.2.7.2 Function

The Modulo register contains the modulo value for the TPM counter. When the TPM counter reaches the modulo value and increments, the overflow flag (TOF) is set and the next value of TPM counter depends on the selected counting method (see Counter).

Writing to the MOD register latches the value into a buffer. The MOD register is updated with the value of its write buffer according to MOD Register Update . Additional writes to the MOD write buffer are ignored until the register has been updated.

It is recommended to initialize the TPM counter (write to CNT) before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

62.4.2.7.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MOD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MOD															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

62.4.2.7.4 Fields

Field	Function
31-0 MOD	Modulo value This field must be written with single 16-bit or 32-bit access.

62.4.2.8 Capture and Compare Status (STATUS)

62.4.2.8.1 Offset

Register	Offset
STATUS	1Ch

62.4.2.8.2 Function

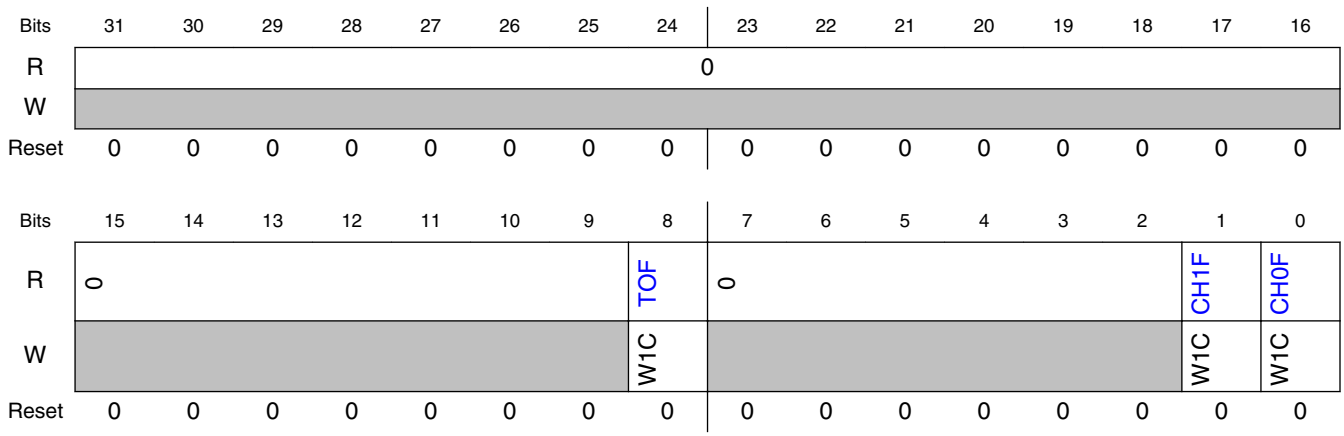
The STATUS register contains a copy of the status flag, CnSC[CHnF] for each TPM channel, as well as SC[TOF], for software convenience.

Each CHnF bit in STATUS is a mirror of CHnF bit in CnSC. All CHnF bits can be checked using only one read of STATUS. All CHnF bits can be cleared by writing all ones to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. Writing a 1 to CHF clears it. Writing a 0 to CHF has no effect.

If another event occurs between the flag setting and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

62.4.2.8.3 Diagram



62.4.2.8.4 Fields

Field	Function
31-9 —	Reserved
8 TOF	Timer Overflow Flag See register description 0b - TPM counter has not overflowed. 1b - TPM counter has overflowed.
7-2 —	Reserved
1 CH1F	Channel 1 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.
0 CH0F	Channel 0 Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.

62.4.2.9 Channel (n) Status and Control (C0SC - C1SC)

62.4.2.9.1 Offset

Register	Offset
C0SC	20h
C1SC	28h

62.4.2.9.2 Function

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

When switching from one channel mode to a different channel mode, the channel must first be disabled and this must be acknowledged in the TPM counter clock domain.

Table 62-5. Mode, Edge, and Level Selection

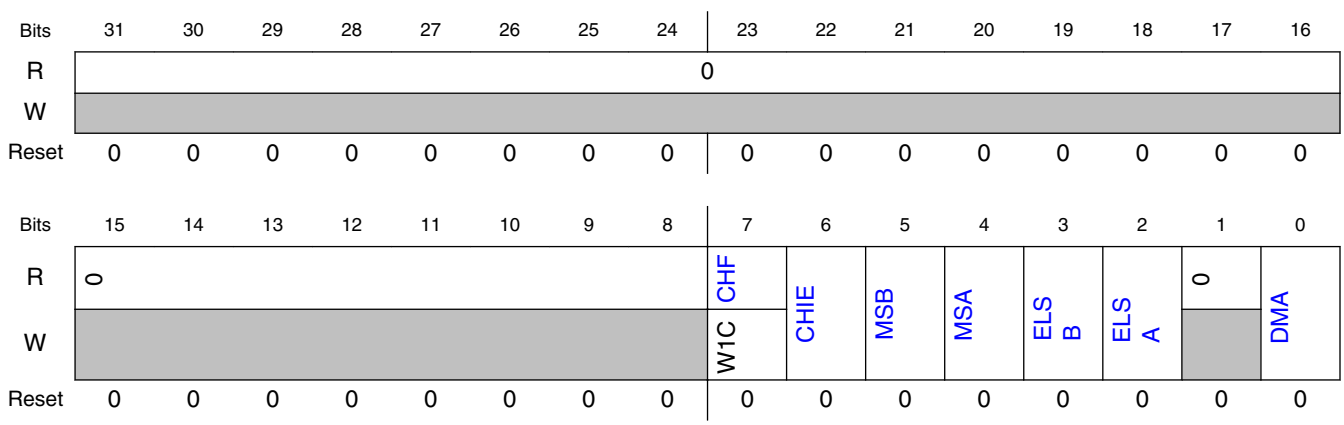
CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
0	00	00	None	Channel disabled
0	01	00	Software compare	Pin not used for TPM
0	00	01	Input capture	Capture on Rising Edge Only
		10		Capture on Falling Edge Only
		11		Capture on Rising or Falling Edge
	01	01	Output compare	Toggle Output on match
		10		Clear Output on match
		11		Set Output on match
	10	10	Edge-aligned PWM	High-true pulses (clear Output on counter match, set Output on counter reload, set Output when counter first enabled or paused)
		00		Reserved
		01		Low-true pulses (set Output on counter match, clear Output on counter reload, clear Output when counter first enabled or paused)
		11		Reserved

Table continues on the next page...

Table 62-5. Mode, Edge, and Level Selection (continued)

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
	11	10	Output compare	Pulse Output low on match
		01		Pulse Output high on match
1	10	10	Center-aligned PWM	High-true pulses (clear Output on counter match-up, set Output on counter match-down, set Output on counter reload or when counter first enabled or paused)
		00		Reserved
		01		Low-true pulses (set Output on match-up, clear Output on match-down, clear Output on counter reload or when counter first enabled or paused))
		11		Reserved

62.4.2.9.3 Diagram



62.4.2.9.4 Fields

Field	Function
31-8 —	Reserved
7	Channel Flag

Table continues on the next page...

Field	Function
CHF	Set by hardware when an event occurs on the channel. CHF is cleared by writing a 1 to the CHF bit. Writing a 0 to CHF has no effect. If another event occurs between the CHF sets and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the delay in clearing the previous CHF. 0b - No channel event has occurred. 1b - A channel event has occurred.
6 CHIE	Channel Interrupt Enable Enables channel interrupts. 0b - Disable channel interrupts. 1b - Enable channel interrupts.
5 MSB	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
4 MSA	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
3 ELSB	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.ph>
2 ELSA	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain.
1 —	Reserved
0 DMA	DMA Enable Enables DMA transfers for the channel. 0b - Disable DMA transfers. 1b - Enable DMA transfers.

62.4.2.10 Channel (n) Value (C0V - C1V)

62.4.2.10.1 Offset

Register	Offset
C0V	24h
C1V	2Ch

62.4.2.10.2 Function

These registers contain the captured TPM counter value for the input modes or the match value for the output modes.

In input capture mode, any write to a CnV register is ignored.

In compare modes, writing to a CnV register latches the value into a buffer. A CnV register is updated with the value of its write buffer according to CnV Register Update . Additional writes to the CnV write buffer are ignored until the register has been updated.

62.4.2.10.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

62.4.2.10.4 Fields

Field	Function
31-0 VAL	Channel Value Captured TPM counter value of the input modes or the match value for the output modes. This field must be written with single 16-bit or 32-bit access.

62.4.2.11 Combine Channel Register (COMBINE)

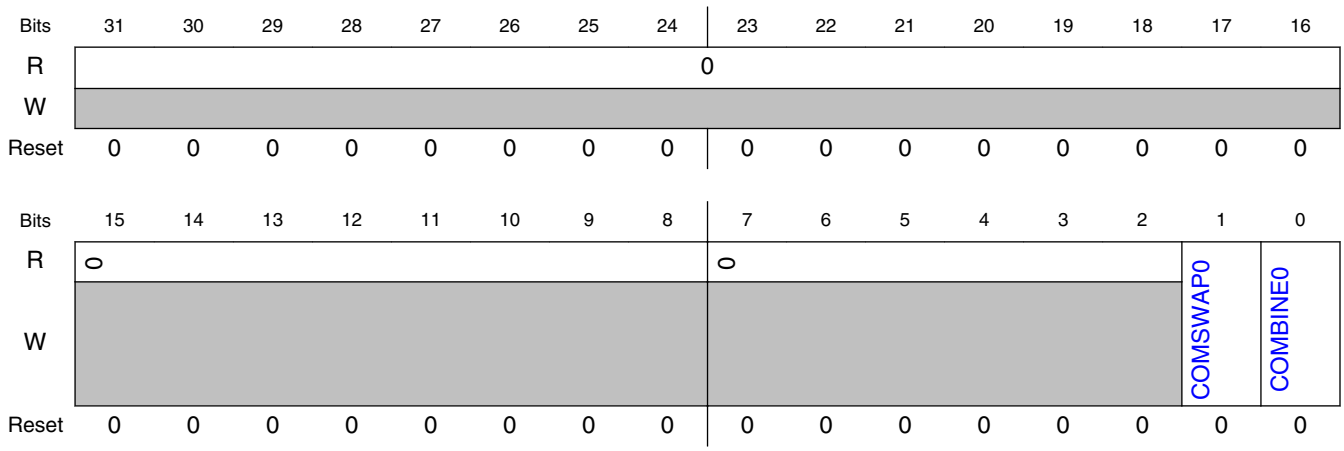
62.4.2.11.1 Offset

Register	Offset
COMBINE	64h

62.4.2.11.2 Function

This register contains the control bits used to configure the combine channel modes for each pair of channels (n) and (n+1), where n is all the even numbered channels.

62.4.2.11.3 Diagram



62.4.2.11.4 Fields

Field	Function
31-8 —	Reserved
7-2 —	Reserved
1 COMSWAP0	Combine Channel 0 and 1 Swap When set in combine mode, the even channel is used for the input capture and 1st compare, the odd channel is used for the 2nd compare. 0b - Even channel is used for input capture and 1st compare. 1b - Odd channel is used for input capture and 1st compare.
0 COMBINE0	Combine Channels 0 and 1 Enables the combine feature for channels 0 and 1. In input capture mode, the combined channels use the even channel input. In software compare modes, the even channel match asserts the output trigger and the odd channel match negates the output trigger. In PWM modes, the even channel match is used for the 1st compare and odd channel match for the 2nd compare. 0b - Channels 0 and 1 are independent. 1b - Channels 0 and 1 are combined.

62.4.2.12 Channel Trigger (TRIG)

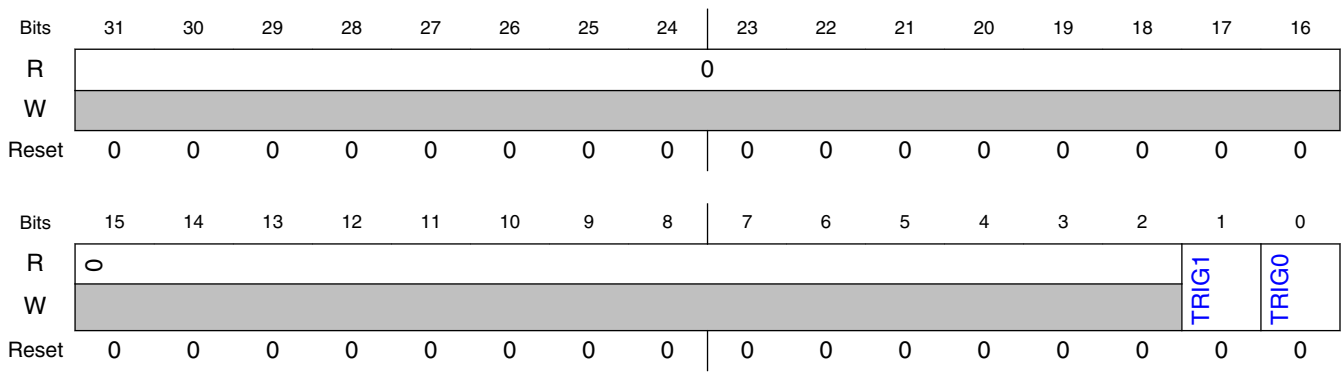
62.4.2.12.1 Offset

Register	Offset
TRIG	6Ch

62.4.2.12.2 Function

In input capture mode, configures the trigger input that is used by the channel to capture the counter value. In output compare or PWM mode, configures the trigger input used to modulate the channel output. When modulating the output, the output is forced to zero whenever the trigger is not asserted. Note that the even numbered channels share the first input trigger source and the odd numbered channels share the second input trigger source.

62.4.2.12.3 Diagram



62.4.2.12.4 Fields

Field	Function
31-2 —	Reserved
1 TRIG1	Channel 1 Trigger 0b - No effect. 1b - Configures trigger input 1 to be used by channel 1.
0 TRIG0	Channel 0 Trigger 0b - No effect. 1b - Configures trigger input 0 to be used by channel 0.

62.4.2.13 Channel Polarity (POL)

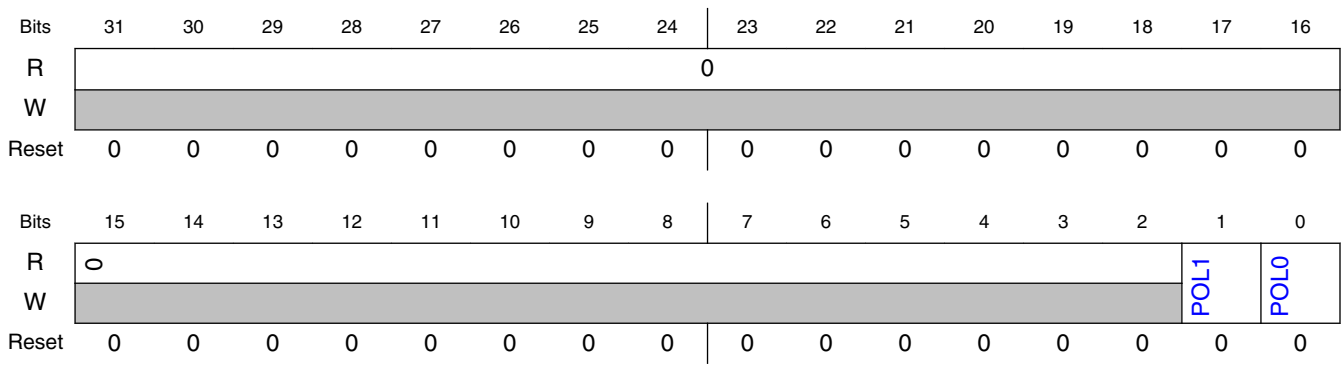
62.4.2.13.1 Offset

Register	Offset
POL	70h

62.4.2.13.2 Function

This register defines the input and output polarity of each of the channels.

62.4.2.13.3 Diagram



62.4.2.13.4 Fields

Field	Function
31-2 —	Reserved
1 POL1	Channel 1 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.
0 POL0	Channel 0 Polarity 0b - The channel polarity is active high. 1b - The channel polarity is active low.

62.4.2.14 Filter Control (FILTER)

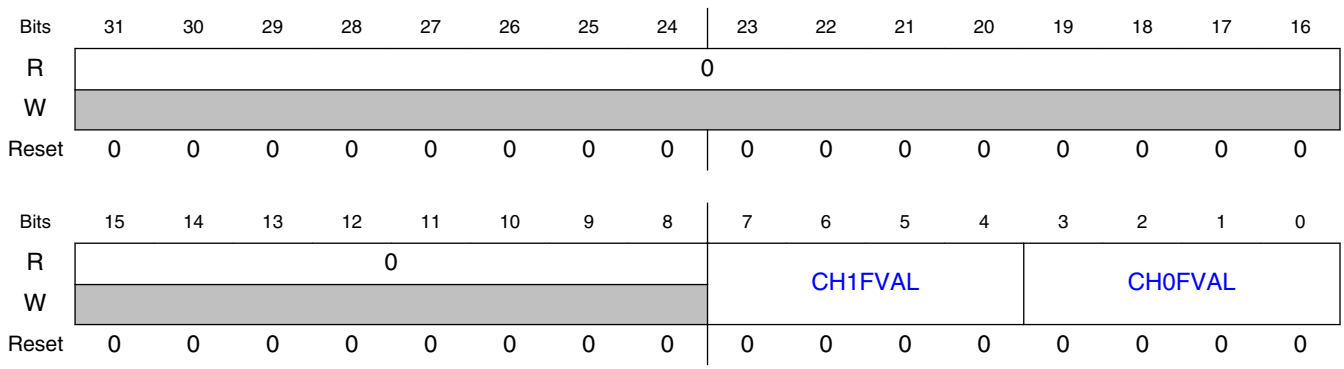
62.4.2.14.1 Offset

Register	Offset
FILTER	78h

62.4.2.14.2 Function

This register selects the filter value of the channel inputs, and an additional output delay value for the channel outputs. In PWM combine modes, the filter can be used to implement deadtime insertion.

62.4.2.14.3 Diagram



62.4.2.14.4 Fields

Field	Function
31-8 —	Reserved
7-4 CH1FVAL	Channel 1 Filter Value Sets the filter value for the channel 1 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH1FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 1 in PWM modes. Deadtime insertion is disabled when CH1FVAL is zero, otherwise deadtime insertion for channel 1 is configured as (CH1FVAL * 4) clock cycles.
3-0 CH0FVAL	Channel 0 Filter Value Sets the filter value for the channel 0 input when configured for input capture or software compare modes. The filter is disabled when the value is zero, otherwise the filter is configured as (CH0FVAL * 4) clock cycles. Sets the deadtime insertion time for channel 0 in PWM modes. Deadtime insertion is disabled when CH0FVAL is zero, otherwise deadtime insertion for channel 0 is configured as (CH0FVAL * 4) clock cycles.

62.4.2.15 Quadrature Decoder Control and Status (QDCTRL)

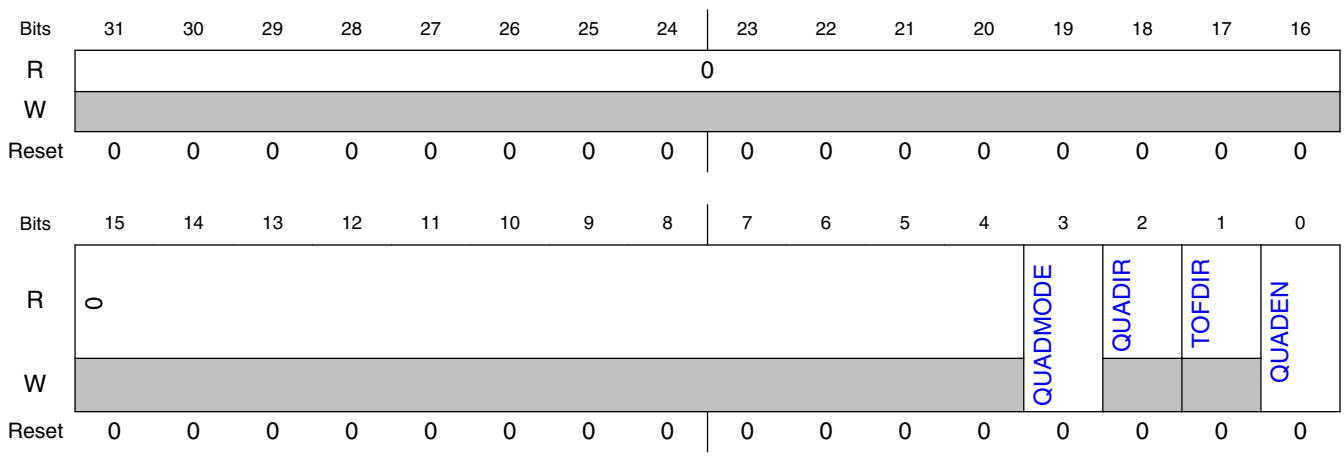
62.4.2.15.1 Offset

Register	Offset
QDCTRL	80h

62.4.2.15.2 Function

This register has the control and status bits for the quadrature decoder mode.

62.4.2.15.3 Diagram



62.4.2.15.4 Fields

Field	Function
31-4 —	Reserved
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the quadrature decoder mode. 0b - Phase encoding mode. 1b - Count and direction encoding mode.
2 QUADIR	Counter Direction in Quadrature Decode Mode Indicates the counting direction. 0b - Counter direction is decreasing (counter decrement). 1b - Counter direction is increasing (counter increment).
1 TOFDIR	TOFDIR Indicates if the TOF bit was set on the top or the bottom of counting.

Table continues on the next page...

Field	Function
	0b - TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (zero) to its maximum value (MOD register). 1b - TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (zero).
0 QUADEN	QUADEN Enables the quadrature decoder mode. In this mode, the channel 0 and channel 1 inputs control the TPM counter direction and can only be used for software compare. The quadrature decoder mode has precedence over the other modes. 0b - Quadrature decoder mode is disabled. 1b - Quadrature decoder mode is enabled.

62.4.2.16 Configuration (CONF)

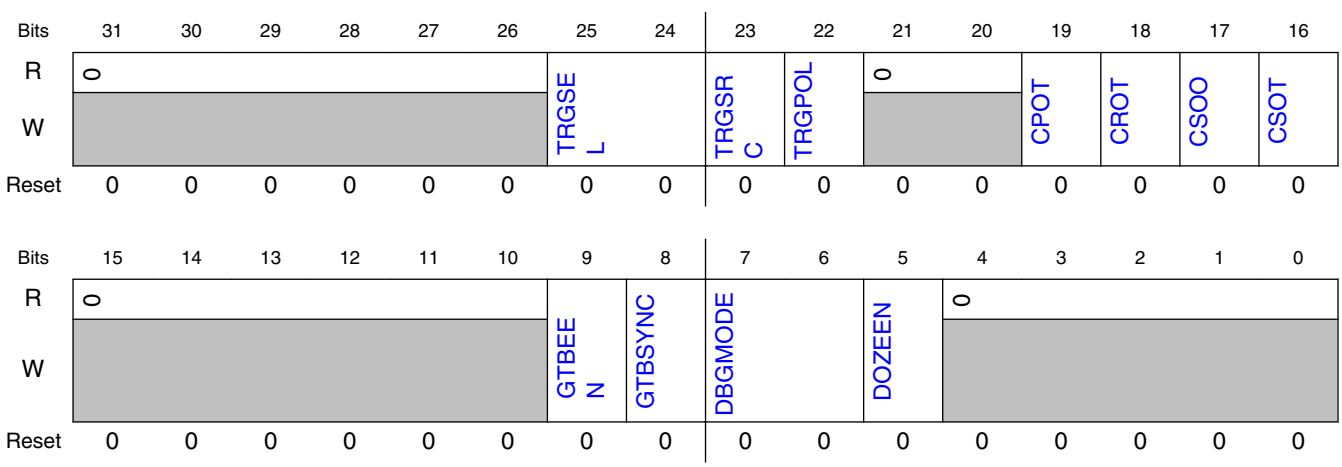
62.4.2.16.1 Offset

Register	Offset
CONF	84h

62.4.2.16.2 Function

This register selects the behavior in debug and wait modes and the use of an external global time base.

62.4.2.16.3 Diagram



62.4.2.16.4 Fields

Field	Function
31-26 —	Reserved
25-24 TRGSEL	Trigger Select Selects the input trigger to use for starting, reloading and/or pausing the counter. The source of the trigger (external or internal to the TPM) is configured by the TRGSRC field. This field should only be changed when the TPM counter is disabled. Refer to chip configuration section for available external trigger options. The available internal trigger sources are listed below. 01b - Channel 0 pin input capture 10b - Channel 1 pin input capture 11b - Channel 0 or Channel 1 pin input capture
23 TRGSRC	Trigger Source Selects between internal (channel pin input capture) or external trigger sources. When selecting an internal trigger, the channel selected should be configured for input capture. Only a rising edge input capture can be used to initially start the counter using the CSOT configuration; either rising edge or falling edge input capture can be used to reload the counter using the CROT configuration; and the state of the channel input pin is used to pause the counter using the CPOT configuration. The channel polarity register can be used to invert the polarity of the channel input pins. This field should only be changed when the TPM counter is disabled. 0b - Trigger source selected by TRGSEL is external. 1b - Trigger source selected by TRGSEL is internal (channel pin input capture).
22 TRGPOL	Trigger Polarity Selects the polarity of the external trigger source. This field should only be changed when the TPM counter is disabled. 0b - Trigger is active high. 1b - Trigger is active low.
21-20 —	Reserved
19 CPOT	Counter Pause On Trigger When enabled, the counter will pause incrementing while the trigger remains asserted (level sensitive). While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled.
18 CROT	Counter Reload On Trigger When set, the TPM counter will reload with zero (and set PWM outputs to their reload state) when a rising edge is detected on the selected trigger input. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled. 0b - Counter is not reloaded due to a rising edge on the selected input trigger 1b - Counter is reloaded when a rising edge is detected on the selected input trigger
17 CSOO	Counter Stop On Overflow

Table continues on the next page...

Field	Function
	When set, the TPM counter will pause incrementing once the counter equals the MOD value and increments (this also sets the TOF). Reloading the counter with 0 due to writing to the counter register or due to a trigger input does not cause the counter to stop incrementing. Once the counter has stopped incrementing, the counter will not start incrementing unless it is disabled and then enabled again, or a rising edge on the selected trigger input is detected when CSOT is set. While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state. This field should only be changed when the TPM counter is disabled. 0b - TPM counter continues incrementing or decrementing after overflow 1b - TPM counter stops incrementing or decrementing after overflow.
16 CSOT	Counter Start on Trigger When set, the TPM counter will pause incrementing when it is enabled until a rising edge on the selected trigger input is detected. If the TPM counter is paused due to an overflow, a rising edge on the selected trigger input will also cause the TPM counter to start incrementing again. While the counter is paused input capture events are ignored, and PWM outputs are forced to their default state. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled. 0b - TPM counter starts to increment immediately, once it is enabled. 1b - TPM counter only starts to increment when it a rising edge on the selected input trigger is detected, after it has been enabled or after it has stopped due to overflow.
15-10 —	Reserved
9 GTBEEN	Global time base enable Configures the TPM to use an externally generated global time base counter. When an externally generated timebase is used, the internal TPM counter is not used by the channels but can be used to generate a periodic interruptor DMA request using the Modulo register and timer overflow flag. 0b - All channels use the internally generated TPM counter as their timebase 1b - All channels use an externally generated global timebase as their timebase
8 GTBSYNC	Global Time Base Synchronization When enabled, the TPM counter is synchronized to the global time base. It uses the global timebase enable, trigger, paused state and overflow to ensure the TPM counter starts incrementing at the same time as the global timebase, stops incrementing at the same time as the global timebase and is reset at the same time as the global timebase. This field should only be changed when the TPM counter is disabled. 0b - Global timebase synchronization disabled. 1b - Global timebase synchronization enabled.
7-6 DBGMODE	Debug Mode Configures the TPM behavior in debug mode. All other configurations are reserved. 00b - TPM counter is paused and does not increment during debug mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their default state. 11b - TPM counter continues in debug mode.
5 DOZEEN	Doze Enable Configures the TPM behavior in wait mode. 0b - Internal TPM counter continues in Doze mode. 1b - Internal TPM counter is paused and does not increment during Doze mode. Trigger inputs and input capture events are ignored, and PWM outputs are forced to their default state.
4-0 —	Reserved

62.5 Functional description

The following sections describe the TPM features.

62.5.1 Clock domains

The TPM module supports two clock domains.

The bus clock domain is used by the register interface and for synchronizing interrupts and DMA requests.

The TPM counter clock domain is used to clock the counter and prescaler along with the output compare and input capture logic. The TPM counter clock is considered asynchronous to the bus clock, can be a higher or lower frequency than the bus clock and can remain operational in Stop mode. Multiple TPM instances are all clocked by the same TPM counter clock in support of the external timebase feature.

62.5.1.1 Counter Clock Mode

The CMOD[1:0] bits in the SC register either disable the TPM counter or select one of three possible clock sources for the TPM counter. After any reset, CMOD[1:0] = 0:0 so the TPM counter is disabled. The CMOD[1:0] can configure one of the following counter clock sources.

- Asynchronous counter clock
- External clock input pin
- External trigger source

The CMOD[1:0] bits may be read or written at any time. Disabling the TPM counter by writing zero to the CMOD[1:0] bits does not affect the TPM counter value or other registers, but must be acknowledged by the TPM counter clock domain before they read as zero.

The external clock input and external trigger source pass through a synchronizer clocked by the TPM counter clock to ensure that counter transitions are properly aligned to counter clock transitions. Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must be less than half of the counter clock frequency.

62.5.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter.

The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and TPM counter.

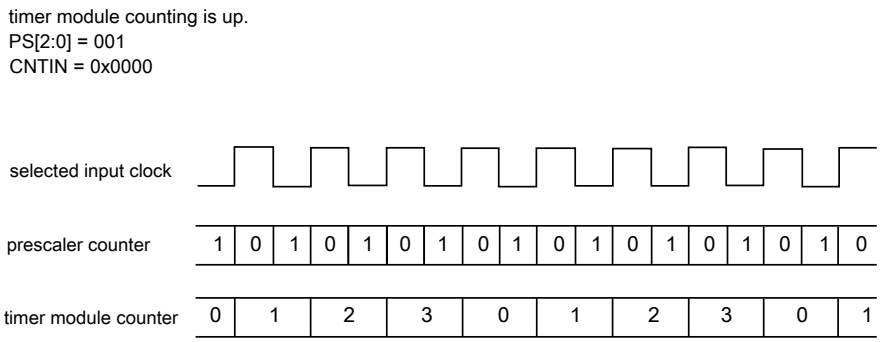


Figure 62-2. Example of the Prescaler Counter

62.5.3 Counter

The TPM has a 16-bit counter that is used by the channels either for input or output modes.

The counter updates from the selected clock divided by the prescaler.

The TPM counter has these modes of operation:

- up counting (see Up counting)
- up-down counting (see Up-down counting)

62.5.3.1 Up counting

Up counting is selected when SC[CPWMS] = 0.

The value of zero is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with zero.

The TPM period when using up counting is (MOD + 0x0001) × period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to zero.

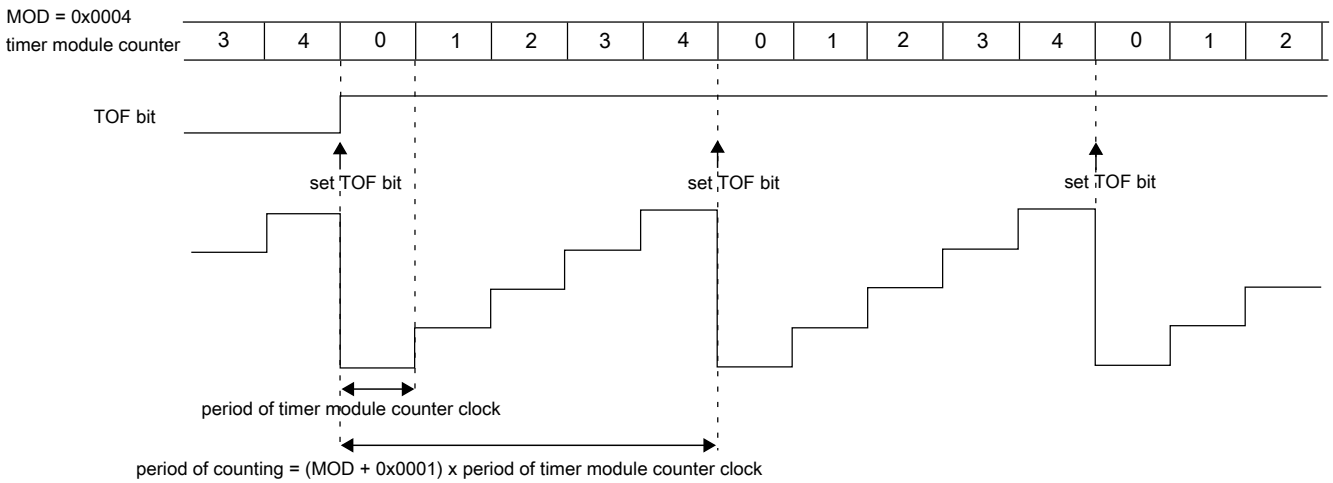


Figure 62-3. Example of TPM Up Counting

Note

- Configuring MOD = 0 is a redundant condition. In this case, the TPM counter is always equal to MOD and the TOF bit is set in each rising edge of the TPM counter clock.
- Configuring MOD = 1 and PS = 0 will attempt to set the TOF bit every second TPM counter clock. Due to synchronization delays between the counter clock and the bus clock, the TOF must be cleared by software two counter clock cycles before TOF can be set again.

62.5.3.2 Up-down counting

Up-down counting is selected when SC[CPWMS] = 1. When configured for up-down counting, configuring CONF[MOD] to less than 2 is not supported.

The value of 0 is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to zero and the up-down counting restarts.

The TPM period when using up-down counting is 2 × MOD × period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to (MOD – 1).

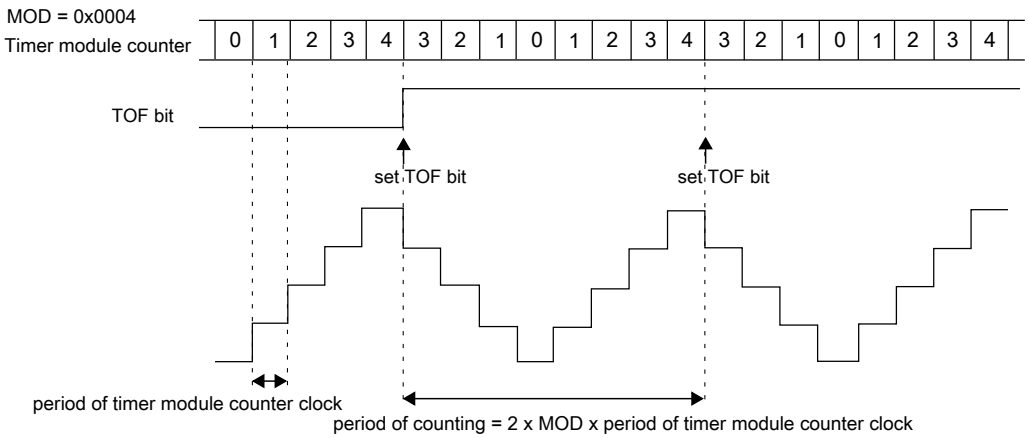


Figure 62-4. Example of up-down counting

62.5.3.3 Counter Reset

Any write to CNT resets the TPM counter and the channel outputs to their initial values (except for channels in output compare mode).

62.5.3.4 Global time base (GTB)

The global time base (GTB) is a TPM function that allows multiple TPM modules to share the same timebase. When the global time base is enabled (CONF[GTBEEN] = 1), the local TPM channels use the counter value, counter enable and overflow indication from the TPM generating the global time base. If the local TPM counter is not generating the global time base, then it can be used as an independent counter or pulse accumulator.

The local TPM counter can also be configured to synchronize to the global time base, by configuring (GTBSYNC = 1). When synchronized to the global time base, the local counter will use the counter enable and counter overflow indication from the TPM generating the global time base. This enables multiple TPM to be configured with the same phase, but with different periods (although the global time base must be configured with the longest period).

62.5.3.5 Counter trigger

The TPM counter can be configured to start, stop or reset in response to a hardware trigger input. The trigger input is synchronized to the asynchronous counter clock, so there is a 3 counter clock delay between the trigger assertion and the counter responding.

- When (CSOT = 1), the counter will not start incrementing until a rising edge is detected on the trigger input.
- When (CSOO= 1), the counter will stop incrementing whenever the TOF flag is set. The counter does not increment again unless it is disabled, or if CSOT = 1 and a rising edge is detected on the trigger input.
- When (CROT= 1), the counter will reset to zero as if an overflow occurred whenever a rising edge is detected on the trigger input. Any PWM channels will update to their reload state.
- When (CPOT = 1), the counter will pause incrementing whenever the trigger input is asserted. The counter will continue incrementing when the trigger input negates. PWM output channels are forced to their default state.

The polarity of the external input trigger can be configured by the TRGPOL register bit.

When an internal trigger source is selected, the trigger input is selected from one or more channel input capture events. The input capture filters are used with the internal trigger sources and the POLn bits can be used to invert the polarity of the input channels. Note that following restrictions apply with input capture channel sources.

- When (CSOT = 1), the counter will only start incrementing on a rising edge on the channel input, provided ELSnA = 1.
- When (CROT= 1), the counter will reset to zero on either edge of the channel input, as configured by ELSnB:ELSnA.
- When (CPOT = 1), the counter will pause incrementing whenever the channel input is asserted. PWM output channels are forced to their default state.

62.5.4 Input Capture Mode

The input capture mode is selected when (CPWMS = 0), (MSnB:MSnA = 0:0), and (ELSnB:ELSnA ≠ 0:0).

When a selected edge occurs on the channel input, the current value of the TPM counter is captured into the CnV register, at the same time the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1 (see the following figure).

When a channel is configured for input capture, the TPM_CHn pin is an edge-sensitive input. ELSnB:ELSnA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is counter clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register are ignored in input capture mode.

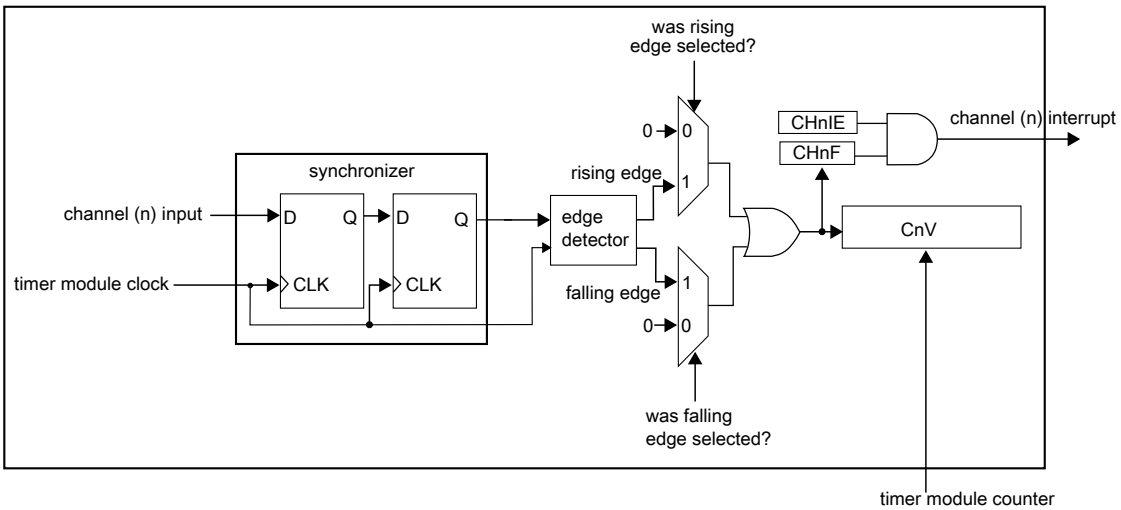


Figure 62-5. Input capture mode

The CHnF bit is set on the third rising edge of the counter clock after a valid edge occurs on the channel input.

62.5.5 Output Compare Mode

The output compare mode is selected when (CPWMS = 0), and (MSnB:MSnA = X:1).

In output compare mode, the TPM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared or toggled if MSnB is clear. If MSnB is set then the channel (n) output is pulsed high or low for as long as the counter matches the value in the CnV register.

When a channel is initially configured to output compare mode, the channel output updates with its negated value (logic 0 for set/toggle/pulse high and logic one for clear/ pulse low).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV).

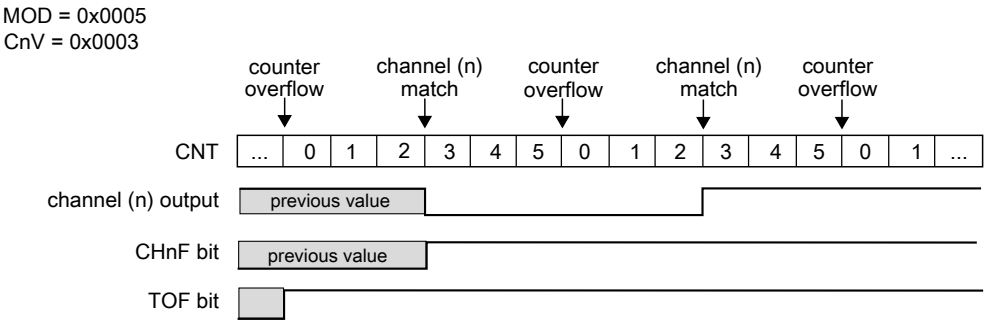


Figure 62-6. Example of the output compare mode when the match toggles the channel output

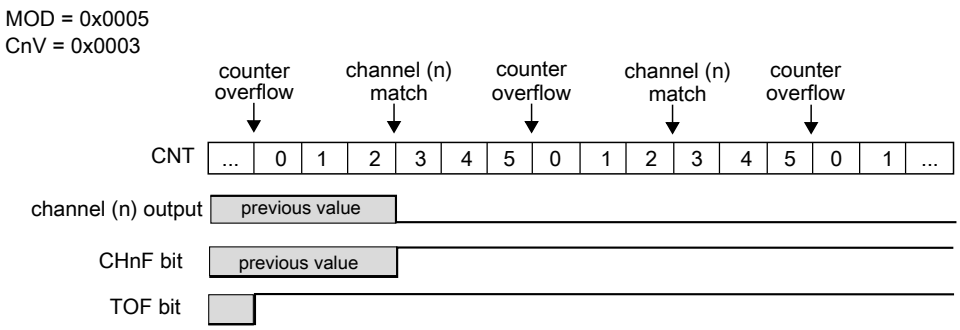


Figure 62-7. Example of the output compare mode when the match clears the channel output

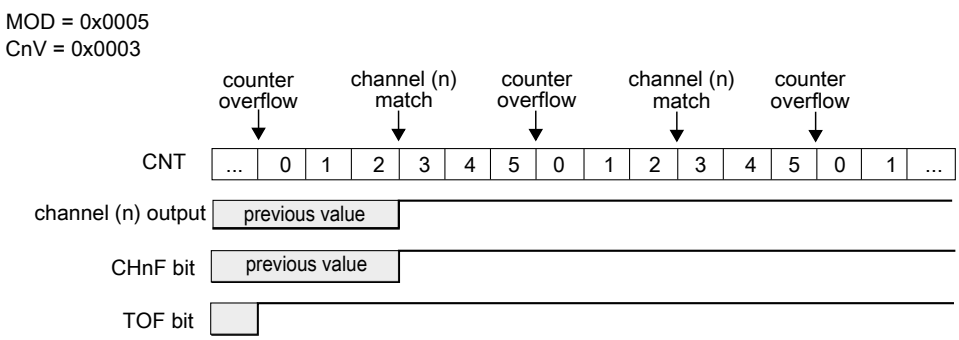


Figure 62-8. Example of the output compare mode when the match sets the channel output

It is possible to use the output compare mode with (MSnB:MSnA = 0:1) and (ELSnB:ELSnA = 0:0). In this case, when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not modified or controlled by TPM.

62.5.6 Edge-Aligned PWM (EPWM) Mode

The edge-aligned mode is selected when (CPWMS = 0), and (MSnB:MSnA = 1:0).

The EPWM period is determined by (MOD + 0x0001) and the pulse width (duty cycle) is determined by CnV.

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an TPM.

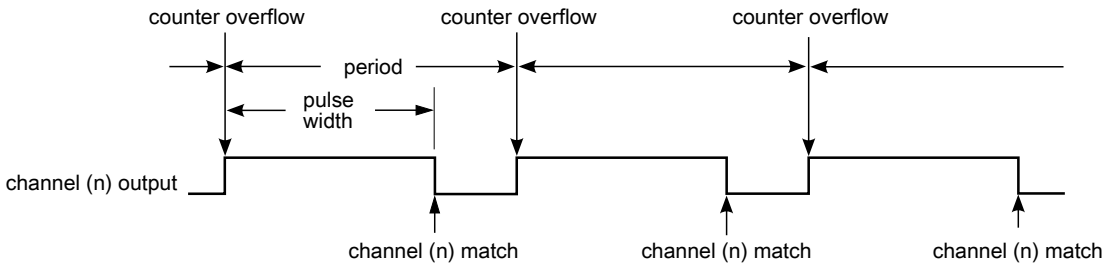


Figure 62-9. EPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow or reload (when the zero is loaded into the TPM counter), and it is forced low at the channel (n) match (TPM counter = CnV). When first enabled or whenever the TPM counter is paused, the channel (n) output is forced high.

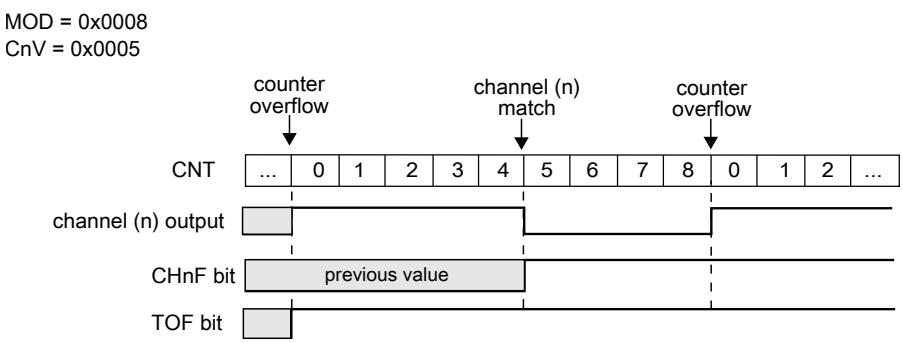


Figure 62-10. EPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:1), then the channel (n) output is forced low at the counter overflow or reload (when zero is loaded into the TPM counter), and it is forced high at the channel (n) match (TPM counter = CnV). When first enabled or whenever the TPM counter is paused, the channel (n) output is forced low.

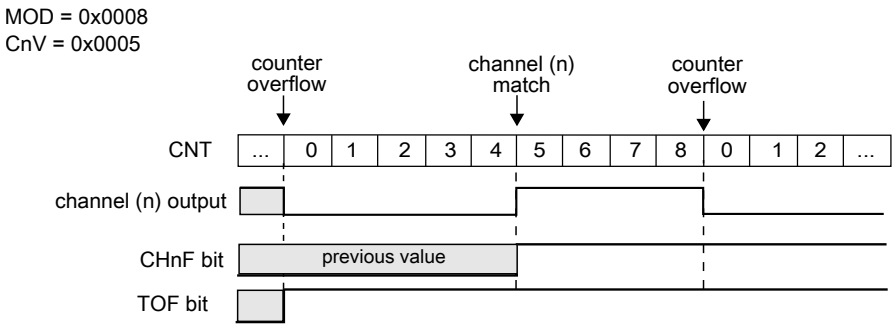


Figure 62-11. EPWM signal with ELSnB:ELSnA = 0:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal. If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set since there is never a channel (n) match. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

62.5.7 Center-Aligned PWM (CPWM) Mode

The center-aligned mode is selected when (CPWMS = 1) and (MSnB:MSnA = 1:0).

The CPWM pulse width (duty cycle) is determined by 2 × CnV and the period is determined by 2 × MOD (see the following figure). MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the TPM counter counts up until it reaches MOD and then counts down until it reaches zero.

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV) when the TPM counting is down (at the begin of the pulse width) and when the TPM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are when the TPM counter is zero.

The other channel modes are not designed to be used with the up-down counter (CPWMS = 1). Therefore, all TPM channels should be used in CPWM mode when (CPWMS = 1).

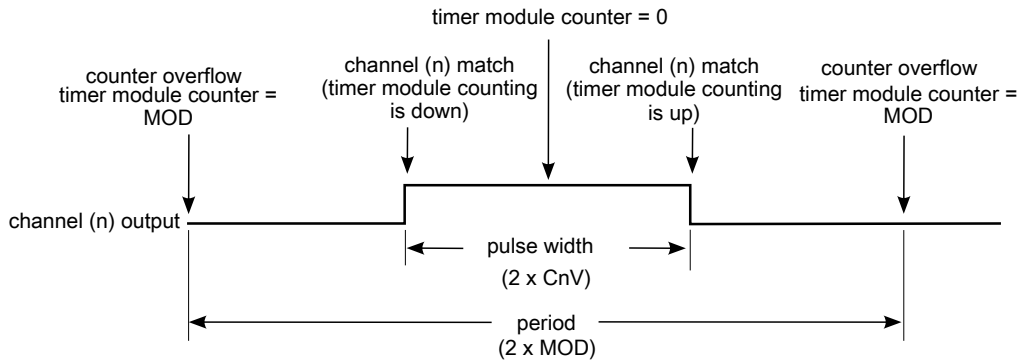


Figure 62-12. CPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (TPM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. When first enabled, when the TPM counter is reloaded, or whenever the TPM counter is paused, the channel (n) output is forced high.

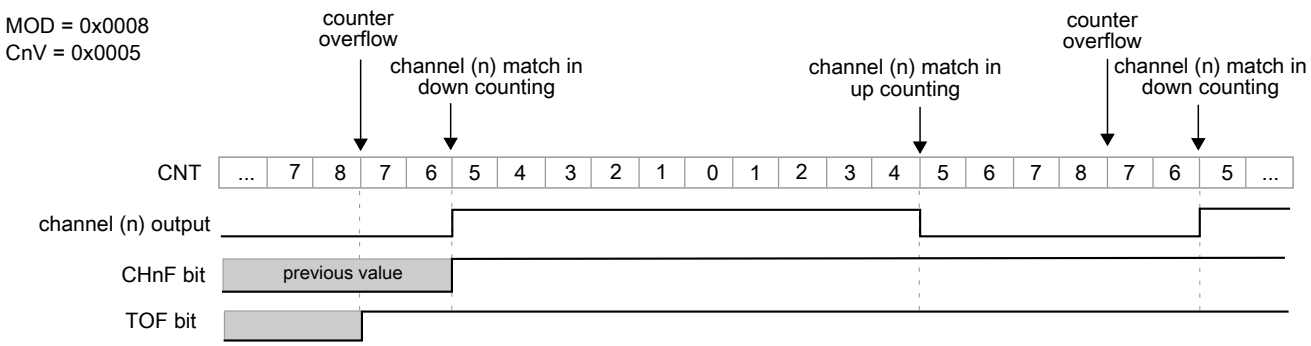


Figure 62-13. CPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:1), then the channel (n) output is forced low at the channel (n) match (TPM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. When first enabled, when the TPM counter is reloaded, or whenever the TPM counter is paused, the channel (n) output is forced low.

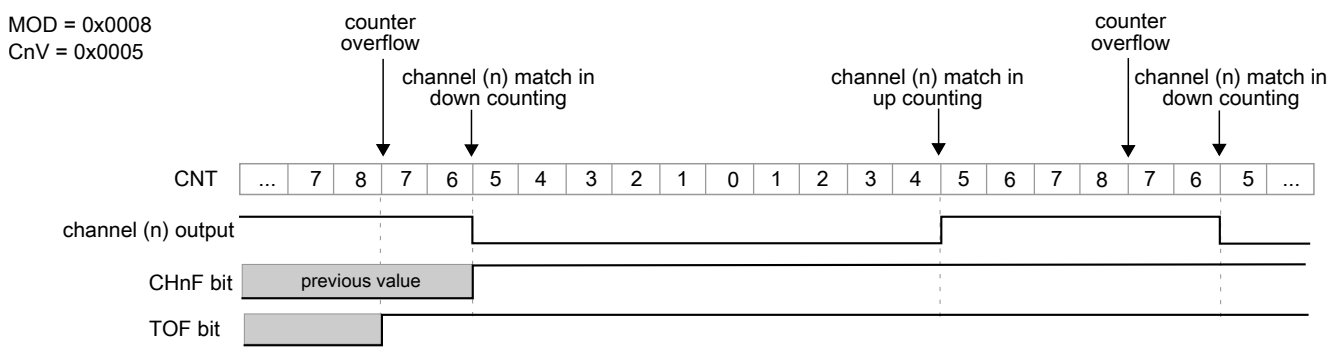


Figure 62-14. CPWM signal with ELSnB:ELSnA = 0:1

If (CnV = 0x0000) then the channel (n) output is a 0% duty cycle CPWM signal.

If (CnV > MOD), then the channel (n) output is a 100% duty cycle CPWM signal, although the CHnF bit is set when the counter changes from incrementing to decrementing. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle CPWM signal.

62.5.8 Combine PWM mode

The Combine PWM mode is selected when:

- MSnB:MSnA = 10
- COMBINEn = 1
- QUADEN = 0, and
- CPWMS = 0

In Combine PWM mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

In the Combine mode, the PWM period is determined by (MOD + 0x0001) and the PWM pulse width (duty cycle) is determined by (|C(n+1)V − C(n)V|).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = C(n)V). The CH(n+1)F bit is set and the channel (n +1) interrupt is generated, if CH(n+1)IE = 1, at the channel (n+1) match (TPM counter = C(n+1)V).

If channel (n) (ELSnB:ELSnA = 0:1), then the channel (n) output is forced low at the beginning of the period (TPM counter is zero) and at the channel (n+1) match (TPM counter = C(n+1)V). It is forced high at the channel (n) match (TPM counter = C(n)V).

If channel (n) (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the beginning of the period (TPM counter is zero) and at the channel (n+1) match (TPM counter = C(n+1)V). It is forced low at the channel (n) match (TPM counter = C(n)V).

When (COMSWAPn = 1), then the channel (n) output is forced low or high at the beginning of the period (TPM counter is zero) and at the channel (n) match (TPM counter = C(n)V). It is forced high or low at the channel (n+1) match (TPM counter = C(n+1)V).

The channel (n+1) output is generated the same as the channel (n) output, but the output polarity is controlled by the channel (n+1) ELSnB:ELSnA configuration.

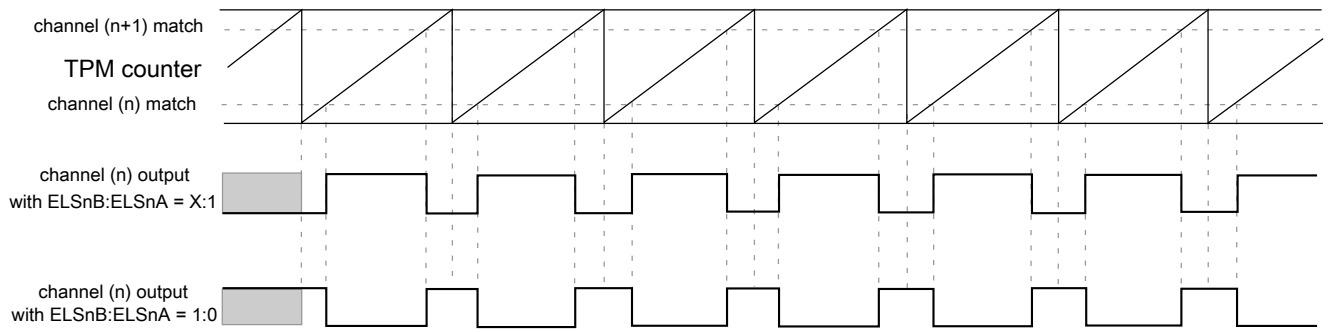


Figure 62-15. Combine mode

The following figures illustrate the PWM signals generation using Combine mode.

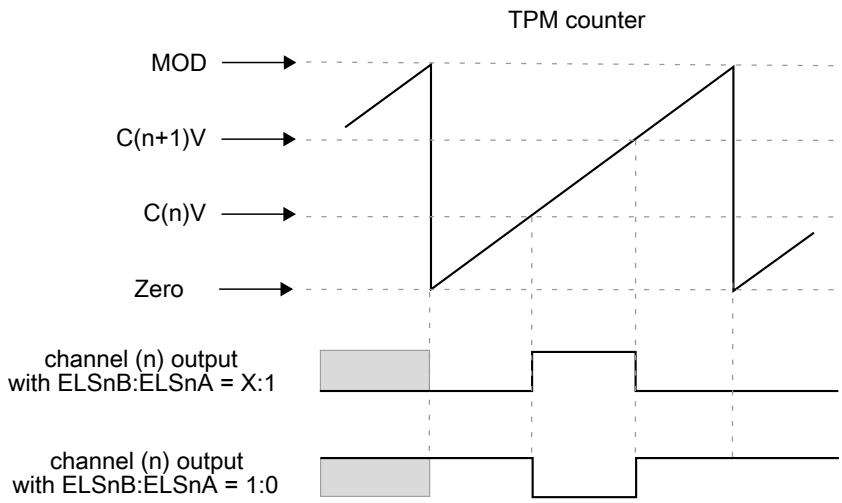


Figure 62-16. Channel (n) output if (C(n)V < MOD) and (C(n+1)V < MOD) and (C(n)V < C(n+1)V)

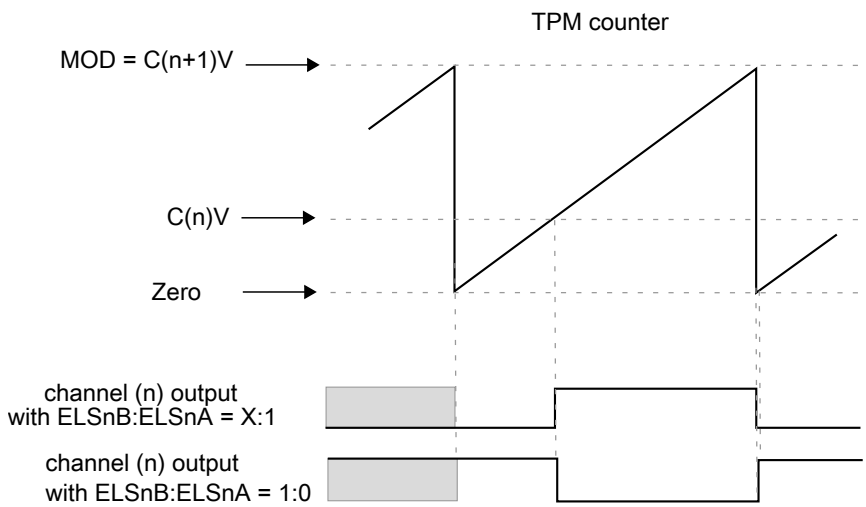


Figure 62-17. Channel (n) output if (C(n)V < MOD) and (C(n+1)V = MOD)

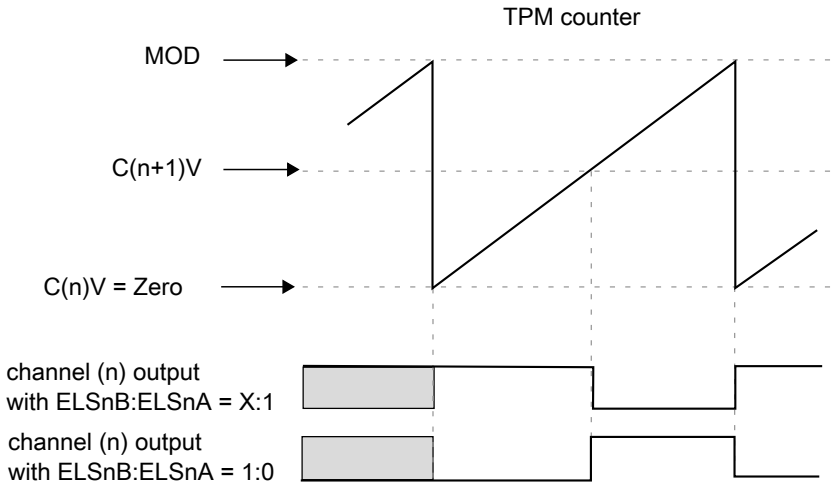


Figure 62-18. Channel (n) output if (C(n)V = zero) and (C(n+1)V < MOD)

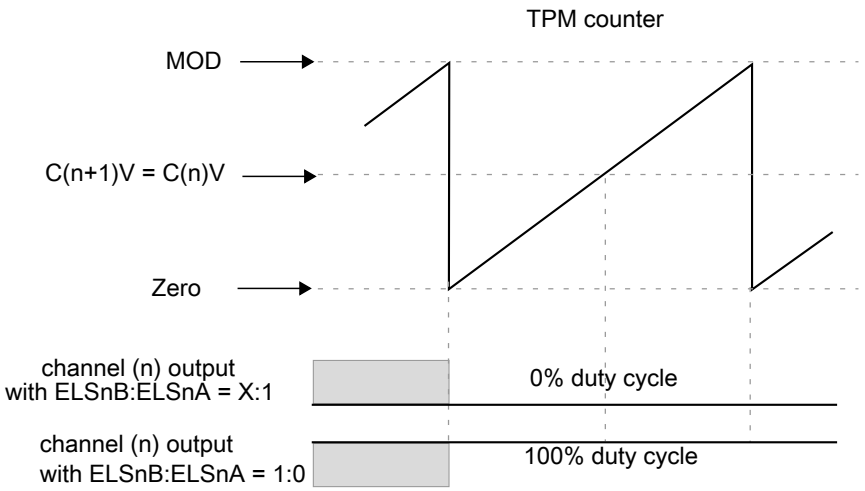


Figure 62-19. Channel (n) output if (C(n)V < MOD) and (C(n+1)V < MOD) and (C(n)V = C(n +1)V)

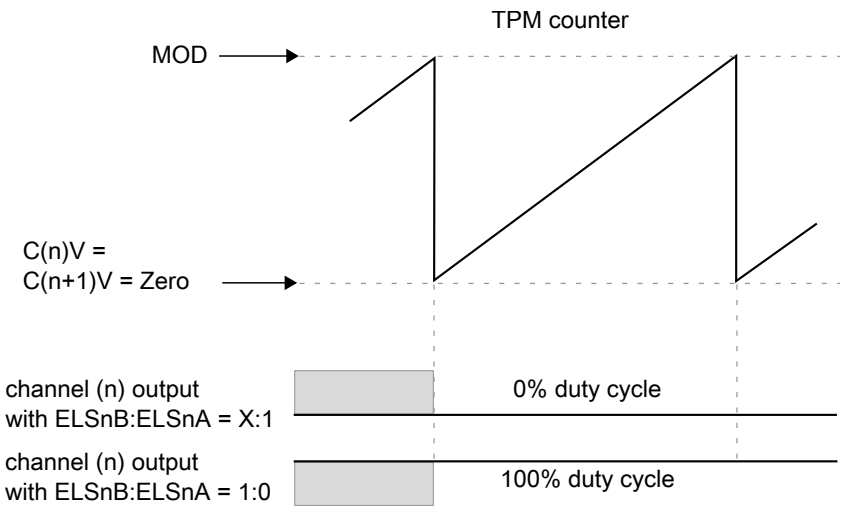


Figure 62-20. Channel (n) output if (C(n)V = C(n+1)V = zero)

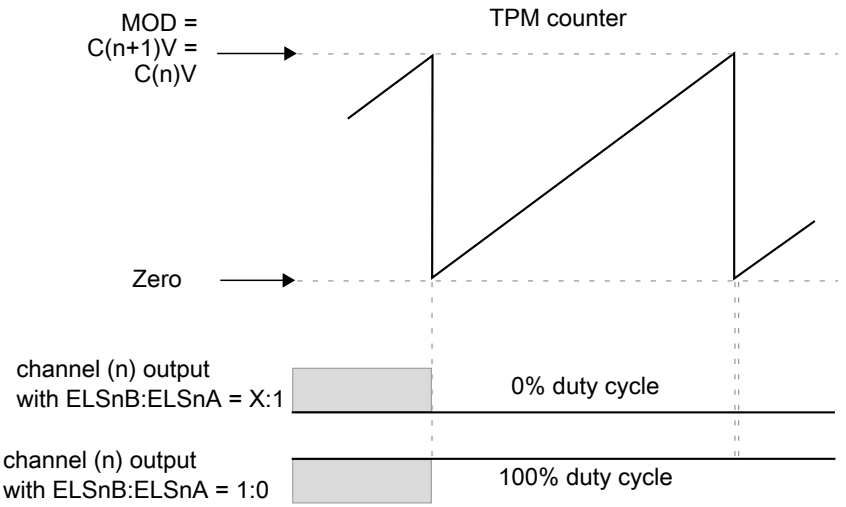


Figure 62-21. Channel (n) output if (C(n)V = C(n+1)V = MOD)

62.5.9 Combine Input Capture mode

The Combine Input Capture mode is selected if COMBINEn= 1 and MSnB:MSnA = 00 and ELSnB:ELSnA != 00. This mode allows to measure a pulse width of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode.

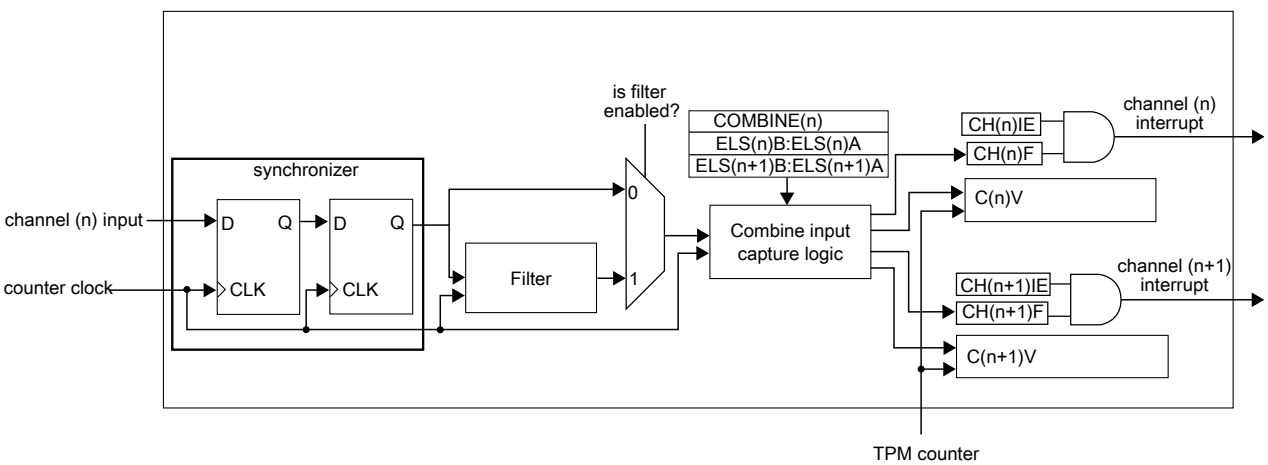


Figure 62-22. Combine Input Capture mode block diagram

The ELS(n)B:ELS(n)A bits select the edge that is captured by channel (n), and ELS(n +1)B:ELS(n+1)A bits select the edge that is captured by channel (n+1).

In the Combine Input Capture mode, only channel (n) input is used and channel (n+1) input is ignored, when COMSWAPn=1 then only channel (n+1) input is used and channel (n) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then CH(n)F bit is set and the channel (n) interrupt is generated (if CH(n)IE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input, then CH(n+1)F bit is set and the channel (n+1) interrupt is generated (if CH(n+1)IE = 1).

The C(n)V register stores the value of TPM counter when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the value of TPM counter when the selected edge by channel (n+1) is detected at channel (n) input.

Note

- The CH(n)F, CH(n)IE, MS(n)A, ELS(n)B, and ELS(n)A bits are channel (n) bits.
- The CH(n+1)F, CH(n+1)IE, MS(n+1)A, ELS(n+1)B, and ELS(n+1)A bits are channel (n+1) bits.
- The Combine Input Capture mode must be used with ELS(n)B:ELS(n)A = 0:1 or 1:0, ELS(n+1)B:ELS(n+1)A = 0:1 or 1:0.

62.5.10 Input Capture Filter

The input capture filter function is only in input capture mode, or in software compare mode when quadrature decoder mode is enabled.

First, the input signal is synchronized by the counter clock. Following synchronization, the input signal enters the filter block. See the following figure.

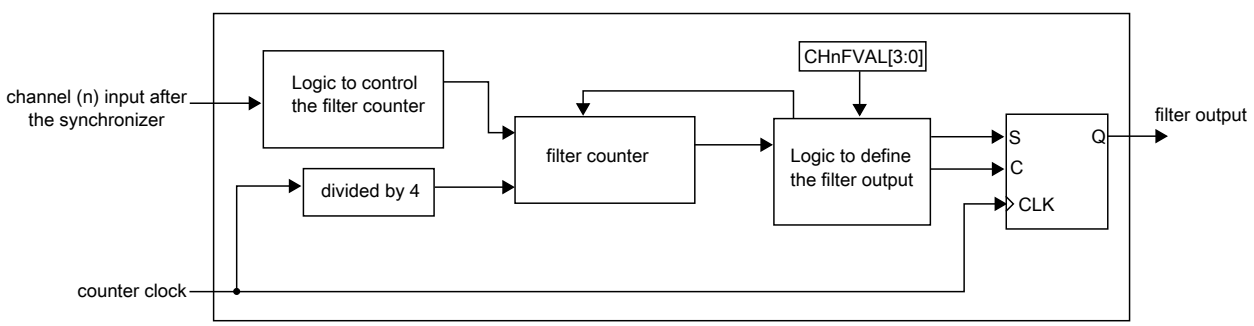


Figure 62-23. Channel input filter

When there is a state change in the input signal, the counter is reset and starts counting up. As long as the new state is stable on the input, the counter continues to increment. When the counter is equal to (CHnFVAL[3:0] × 4), the state change of the input signal is validated.

If the opposite edge appears on the input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. Any pulse that is shorter than the minimum value selected by (CHnFVAL[3:0] × 4 counter clocks) is regarded as a glitch and is not passed through the filter. A timing diagram of the input filter is shown in the following figure.

The filter function is disabled when CHnFVAL[3:0] bits are zero. In this case, the input signal is delayed by 2 rising edges of the counter clock. If (CHnFVAL[3:0] ≠ 0000), then the input signal is delayed by the minimum pulse width (CHnFVAL[3:0] × 4 system clocks) plus a further 3 rising edges of the system clock: two rising edges to the synchronizer, plus one more to the edge detector. In other words, CHnF is set (3 + 4 × CHnFVAL[3:0]) counter clock periods after a valid edge occurs on the channel input.

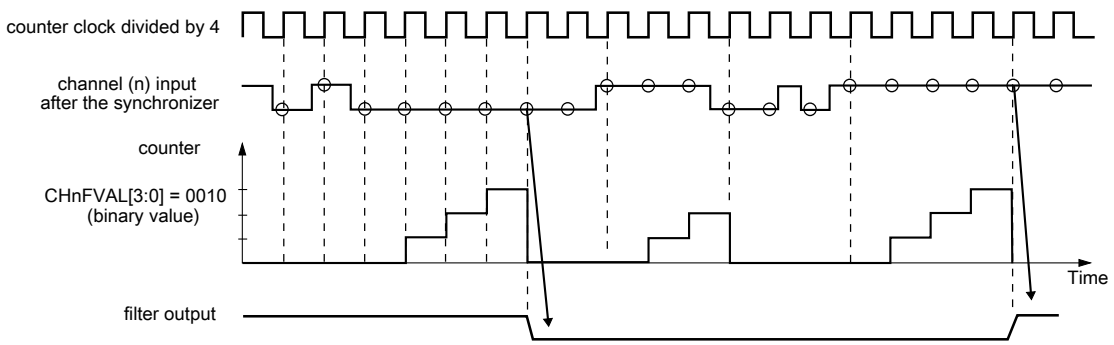


Figure 62-24. Channel input filter example

62.5.11 Deadtime insertion

The deadtime insertion is enabled in PWM combine modes for each pair of channels when CH(n)FVAL and CH(n+1)FVAL are non-zero. The deadtime delay that is used for each TPM channel is defined as (CH(n+1)FVAL[3:0] x 64) + (CH(n)FVAL[3:0] x 4).

The deadtime delay insertion ensures that no two complementary signals (even channel (n) and odd channel (n+1)) drive the active state at the same time.

If ELSnB:ELSnA = 0:1, ELS(n+1)B:ELS(n+1)A = 1:0, COMSWAPn = 0, and the deadtime is enabled, then when the channel (n) match (TPM counter = C(n)V) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set. Similarly, when the channel (n+1) match (TPM counter = C(n +1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. See the following figures.

If ELSnB:ELSnA = 1:0, ELS(n+1)B:ELS(n+1)A = 0:1, COMSWAPn = 1,and the deadtime is enabled, then when the channel (n+1) match (TPM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime

delay when the channel (n+1) output is set. Similarly, when the channel (n) match (TPM counter = CnV) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set.

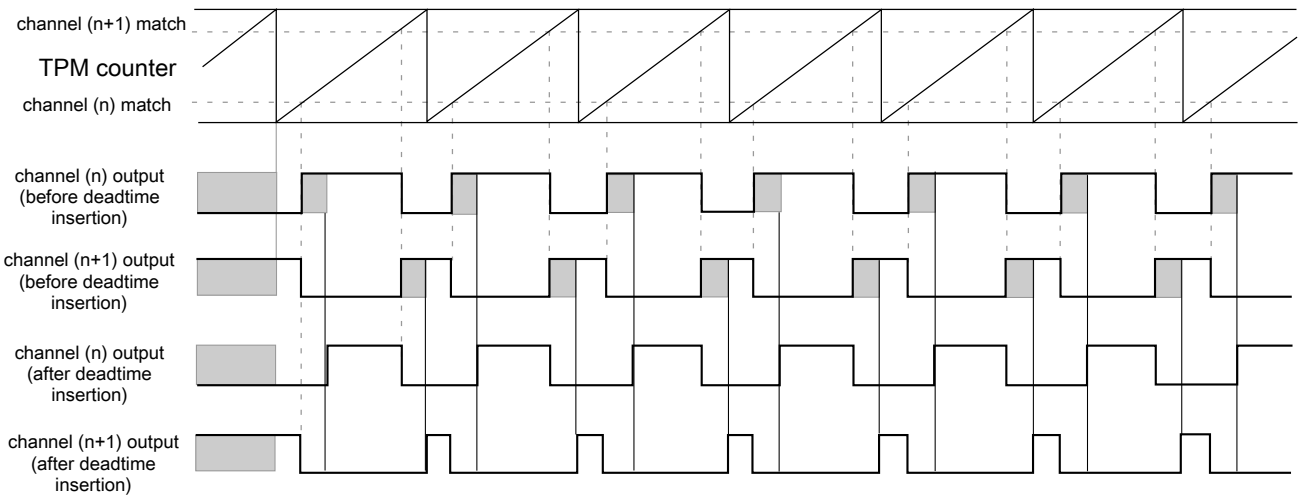


Figure 62-25. Deadtime insertion with ELSnB:ELSnA = 0:1, ELS(n+1)B:ELS(n+1)A = 1:0, COMSWAPn = 0

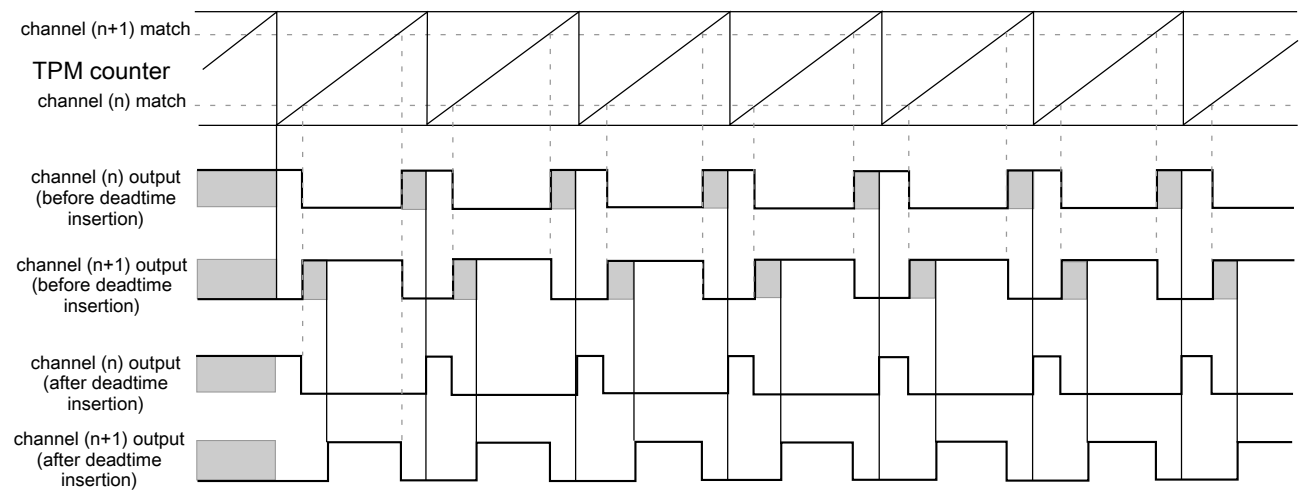


Figure 62-26. Deadtime insertion with ELSnB:ELSnA = 1:0, ELS(n+1)B:ELS(n+1)A = 0:1, COMSWAPn = 1

62.5.12 Quadrature Decoder mode

The Quadrature Decoder mode is selected if (QUADEN = 1). The Quadrature Decoder mode uses the channel 0 (phase A) and channel 1 (phase B) input signals to control the TPM counter increment and decrement. The following figure shows the quadrature decoder block diagram.

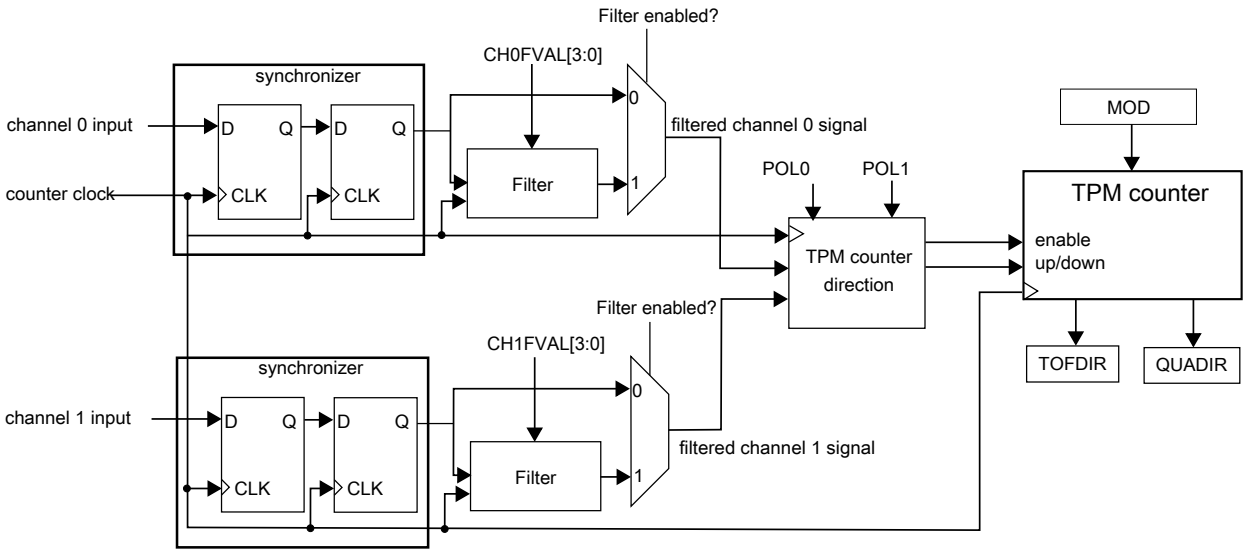


Figure 62-27. Quadrature Decoder block diagram

The input capture filter and channel polarity registers are used to configure the input filter and polarity for the channel 0 and channel 1 inputs in quadrature decode mode.

Note

Notice that the TPM counter is clocked by the channel 0 and channel 1 input signals when quadrature decoder mode is selected. Therefore In quadrature decoder mode, channel 0 and channel 1 can only be used in software compare mode and other TPM channels can only be used in input capture or output compare modes.

The QUADMODE selects the encoding mode used in the Quadrature Decoder mode. If QUADMODE = 1, then the count and direction encoding mode is enabled; see the following figure. In this mode, the channel 1 input value indicates the counting direction, and the channel 0 input defines the counting rate. The TPM counter is updated when there is a rising edge at channel 0 input signal.

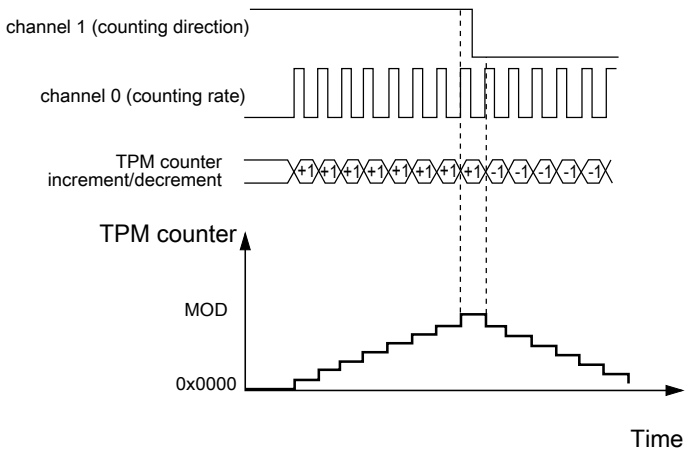


Figure 62-28. Quadrature Decoder – Count and Direction Encoding mode

If QUADMODE = 0, then the Phase Encoding mode is enabled; see the following figure. In this mode, the relationship between channel 0 and channel 1 signals indicates the counting direction, and channel 0 and channel 1 signals define the counting rate. The TPM counter is updated when there is an edge either at the channel 0 or channel 1 signals.

If CH0POL= 0 and CH1POL = 0, then the TPM counter increment happens when:

- there is a rising edge at channel 0 signal and channel 1 signal is at logic zero;
- there is a rising edge at channel 1 signal and channel 0 signal is at logic one;
- there is a falling edge at channel 1 signal and channel 0 signal is at logic zero;
- there is a falling edge at channel 0 signal and channel 1 signal is at logic one;

and the TPM counter decrement happens when:

- there is a falling edge at channel 0 signal and channel 1 signal is at logic zero;
- there is a falling edge at channel 1 signal and channel 0 signal is at logic one;
- there is a rising edge at channel 1 signal and channel 0 signal is at logic zero;
- there is a rising edge at channel 0 signal and channel 1 signal is at logic one.

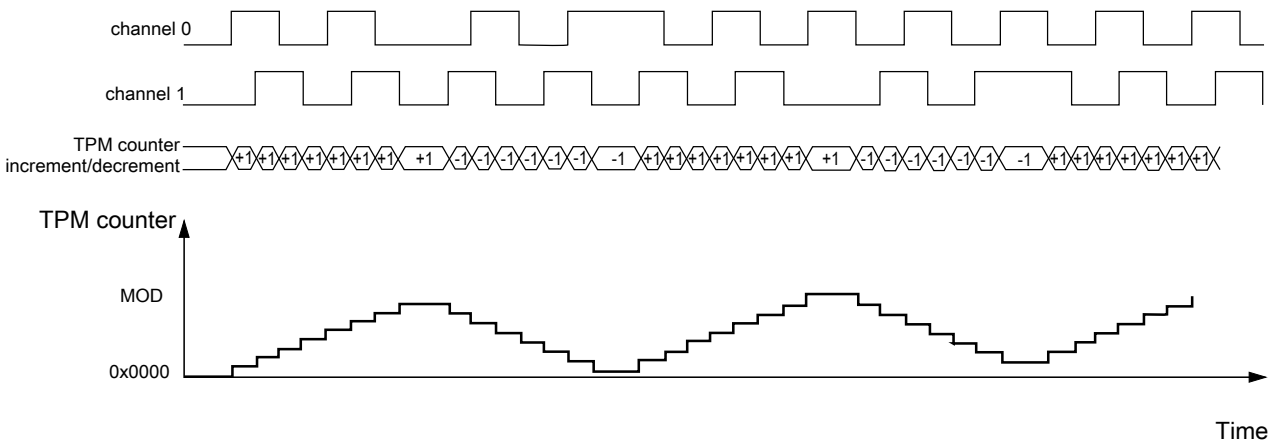


Figure 62-29. Quadrature Decoder – Phase Encoding mode

The following figure shows the TPM counter overflow in up counting. In this case, when the TPM counter changes from MOD to zero, TOF and TOFDIR bits are set. TOF bit indicates the TPM counter overflow occurred. TOFDIR indicates the counting was up when the TPM counter overflow occurred.

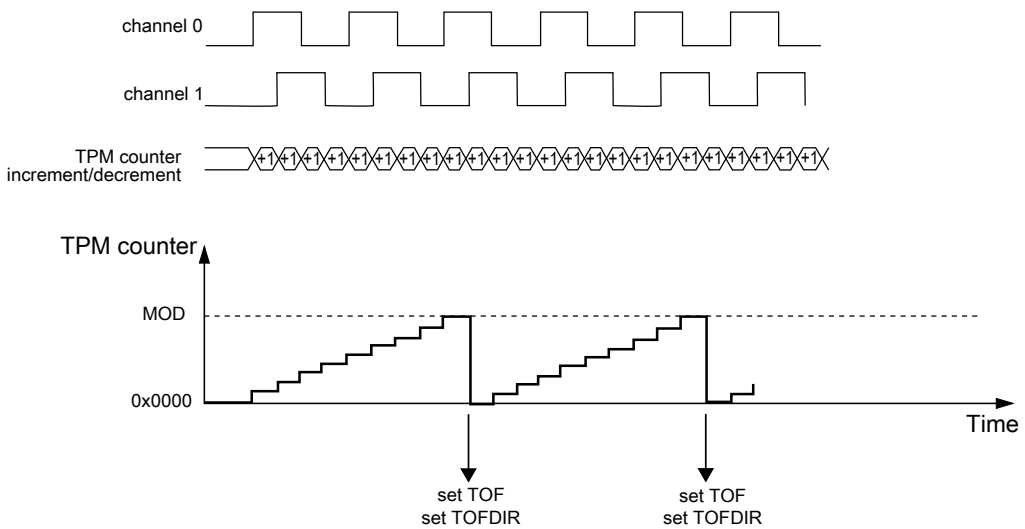


Figure 62-30. TPM Counter overflow in up counting for Quadrature Decoder mode

The following figure shows the TPM counter overflow in down counting. In this case, when the TPM counter changes from zero to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the TPM counter overflow occurred. TOFDIR indicates the counting was down when the TPM counter overflow occurred.

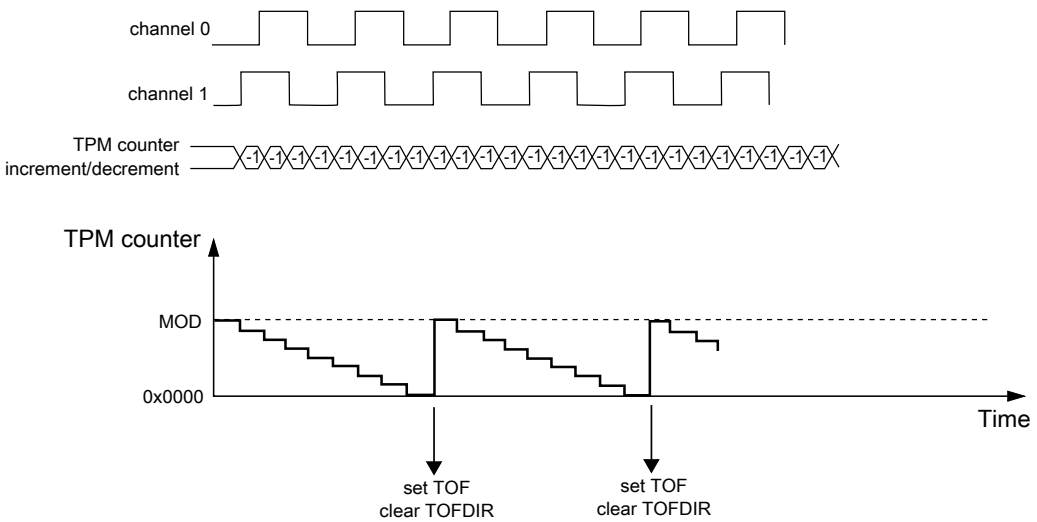


Figure 62-31. TPM counter overflow in down counting for Quadrature Decoder mode

62.5.13 Registers Updated from Write Buffers

62.5.13.1 MOD Register Update

If (CMOD[1:0] = 0:0) then MOD register is updated when MOD register is written.

If (CMOD[1:0] ≠ 0:0), then MOD register is updated according to the CPWMS bit, that is:

- If the selected mode is not CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to (MOD – 1).

62.5.13.2 CnV Register Update

If (CMOD[1:0] = 0:0) then CnV register is updated when CnV register is written.

If (CMOD[1:0] ≠ 0:0), then CnV register is updated according to the selected mode, that is:

- If the selected mode is output compare then CnV register is updated on the next TPM counter increment (end of the prescaler counting) after CnV register was written.

- If the selected mode is EPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to (MOD – 1).

62.5.14 DMA

The channel and overflow flags generate a DMA transfer request according to DMA and CHnIE/TOIE bits.

See the following table for more information.

Table 62-6. DMA Transfer Request

DMA	CHnIE/ TOIE	Channel/Overflow DMA Transfer Request	Channel/Overflow Interrupt
0	0	The channel/overflow DMA transfer request is not generated.	The channel/overflow interrupt is not generated.
0	1	The channel/overflow DMA transfer request is not generated.	The channel/overflow interrupt is generated if (CHnF/TOF = 1).
1	0	The channel/overflow DMA transfer request is generated if (CHnF/TOF = 1).	The channel/overflow interrupt is not generated.
1	1	The channel/overflow DMA transfer request is generated if (CHnF/TOF = 1).	The channel/overflow interrupt is generated if (CHnF/TOF = 1).

If DMA = 1, the CHnF/TOF bit can be cleared either by DMA transfer done or writing a one to CHnF/TOF bit (see the following table).

Table 62-7. Clear CHnF/TOF Bit

DMA	How CHnF/TOF Bit Can Be Cleared
0	CHnF/TOF bit is cleared by writing a 1 to CHnF/TOF bit.
1	CHnF/TOF bit is cleared either when the DMA transfer is done or by writing a 1 to CHnF/TOF bit.

62.5.15 Peripheral Triggers

The connection of the TPM peripheral triggers with other peripherals are device specific.

62.5.15.1 Output Triggers

The TPM generates output triggers for the counter and each channel that can be used to trigger events in other peripherals. The counter trigger asserts whenever the TOF is set and remains asserted until the next increment.

Each TPM channel generates both a pre-trigger output and a trigger output. The pre-trigger output asserts whenever the CHnF is set, the trigger output asserts on the first counter increment after the pre-trigger asserts, and then both the trigger and pre-trigger negate on the first counter increment after the trigger asserts.

When (COMBINEn = 1) in output compare modes, the pre-trigger output for both channel (n) and channel (n+1) will assert when CH(n)F is set and will negate when CH(n +1)F is set. The trigger continues to assert on the first counter increment after the pre-trigger asserts and negates at the same time as the pre-trigger negation.

62.5.15.2 Input Trigger

The TPM input trigger can be configured to perform the following operations with configurable polarity:

- Increment the counter on trigger rising edge
- Start incrementing the counter on trigger rising edge
- Reset/reload the counter on trigger rising edge
- Pause the counter while trigger high

The TPM input trigger is synchronized and must assert for at least two counter clock cycles so that the input trigger can be detected.

62.5.16 Reset Overview

The TPM is reset whenever any chip reset occurs.

When the TPM exits from reset:

- the TPM counter and the prescaler counter are zero and are stopped (CMOD[1:0] = 0:0);
- the timer overflow interrupt is zero;
- the channels interrupts are zero;
- the channels are in input capture mode;
- the channels outputs are zero;
- the channels pins are not controlled by TPM (ELS(n)B:ELS(n)A = 0:0).

62.5.17 TPM Interrupts

This section describes TPM interrupts.

62.5.17.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

62.5.17.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

Chapter 63
Low Power Periodic Interrupt Timer (LPIT)

63.1 Chip-specific LPIT information

Table 63-1. Reference links to related information

Topic	Related module	Reference
Full description	LPIT	LPIT
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

63.1.1 LPIT

Table 63-2. LPIT configuration

Parameter	Description
Name	LPIT
Instances	2
Configurable features	LPIT0-1 • 4 channels, 32-bit timer • LPIT0-1/DMA trigger assignments. See LPIT DMA trigger sources • LPIT0-1/ADC trigger assignments in TRGMUX table. See Table 38-2 and Table 38-3.
Interface speed	NA
External I/O pins	NA

63.1.2 LPIT DMA trigger sources

On this device, LPIT is not directly connected to DMAMUX, it is connected through TRGMUX module. TRGMUX connects a trigger source to LPIT depending on the dedicated TRGMUX register configuration for that module. As described in Table 38-2, Table 38-3, Table 38-4, and Table 38-5, TRGMUX[0/1] outputs 0-7 are connected to DMAMUX[0/1] and LPIT0/1 channels 0-3 can be selected as trigger source.

63.2 Introduction

63.2.1 Overview

LPIT is a low power periodic interrupt timer with multiple timer channels. After a timer reaches a programmed count, the respective timer channel will generate pre-trigger and trigger output signals, and these pre-trigger and trigger outputs can be used to trigger other modules on the device.

- Timers can be configured to be controlled using:
  - external triggers (triggers from outside the LPIT module)
  - or internal triggers (triggers from other timer channels inside the LPIT module).
- The timer channels can be chained together, to form a larger width timer.
- Depending on the timer modes, the timer channels may reload and count again, or stop after reaching the programmed count.

The next figure shows the interface of an LPIT module to the other modules on the SoC.

- The CPU interface provides the clock, reset, register read/write bus interface and handles interrupts from an LPIT.
- The trigger output signals from an LPIT may trigger other modules on the SoC, like the DMA, ADC, and other modules.
- Similarly, other timer modules may provide trigger inputs to an LPIT module, to control when an LPIT timer channel starts.

For the exact module interactions, see the SoC Configuration chapter in your device's hardware Reference Manual (RM).

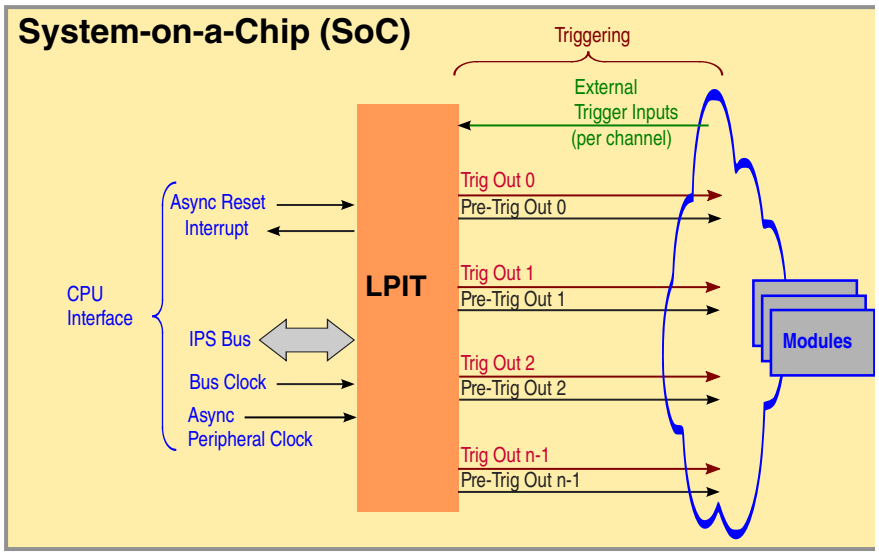


Figure 63-1. LPIT Interface in SoC

Each timer channel in LPIT can be configured to work in either compare modes or capture modes.

- **In compare mode:** the timers decrement when enabled and generate an output pre-trigger and trigger output. The trigger output is 1 clock cycle delayed of the pre-trigger pulse. Each timer channel start, reload and restart can be controlled via control bits. The timer can be configured to always decrement from a programmed start value, or decrement on selected trigger inputs or previous channel timeout (when channels are chained). By chaining timer channels, applications can achieve larger timeout durations.
- **In capture mode:** the timer can be used to perform measurements as the timer value is captured (in the timer value register) when a selected trigger input is asserted. The timer can support once-off or multiple measurements (like frequency measurements).

The timer channels operate on an asynchronous clock, which is independent from the register read/write access clock. Clock synchronization between the clock domains ensures normal operations.

63.2.2 Block Diagram

The next figure shows a detailed block diagram for an LPIT module. The programming model comprises of a global register set (common to all timer channels), and registers for each timer channel (that control their respective timer channels). Access to these registers gets synchronized to the asynchronous peripheral clock, then affects the timer channel registers.

- Each timer channel contains a 32-bit counter that loads the starting value and down counts on every peripheral clock's positive edge.
- After reaching a zero value (a channel timer timeout), a trigger output is generated.
- The counter enable is controlled using a timer enable register control bit, external or internal triggers or via previous channel timeout (when using timer chaining).
- After a channel timer timeout, an interrupt bit is also set, to tell the CPU about the timer timeout.

For more details, see the functional description section.

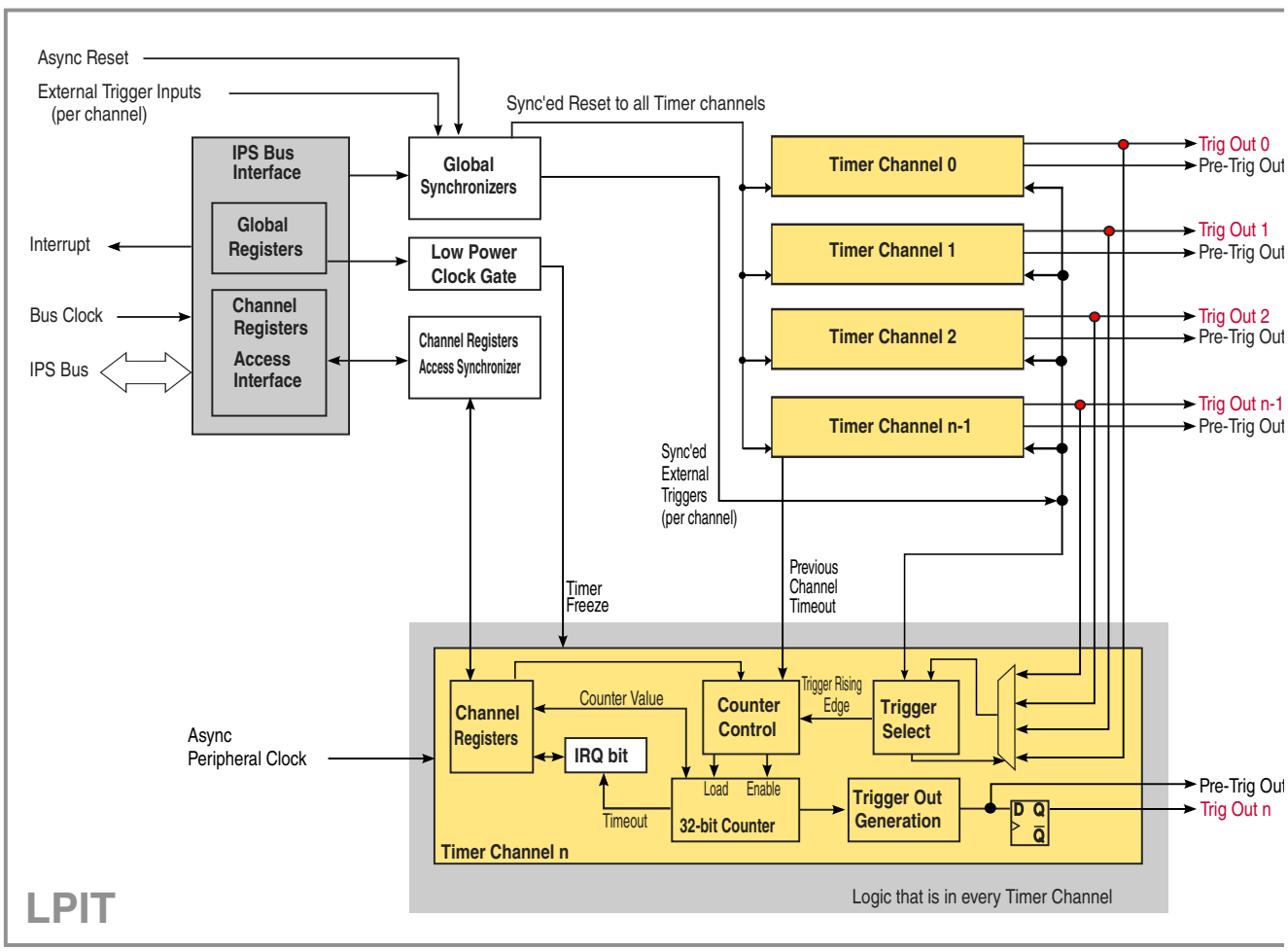


Figure 63-2. Detailed Block Diagram

63.3 Modes of operation

The LPIT module supports the following chip modes.

Table 63-3. Chip modes supported by the LPIT module

Chip mode	LPIT Operation
Run	Normal operation
Stop/Wait	Can continue operating, if the Doze Enable bit (MCR[DOZE_EN]) is set and the LPIT is using an external or internal clock source that remains operating during stop/wait modes.
Low Power Stop (also called Low Leakage Stop or LLS)	The Doze Enable (MCR[DOZE_EN]) bit is ignored and the LPIT will be disabled for the duration of low power mode.
Debug	Can continue operating provided the Debug Enable bit (MCR[DBG_EN]) is set.

63.4 Memory Map and Registers

63.4.1 LPIT register descriptions

The memory map comprises of 32-bit aligned registers, which can be accessed via 8-bit, 16-bit, or 32-bit accesses.

- Write access to reserved locations will generate a transfer error.
- Read access to reserved locations will also generate a transfer error, and the read data bus will show all 0s.

NOTE

- The Memory Map and complete module is in Big Endian format.
- The LPIT module will not check if programmed values in the registers are correct; software must ensure that correct programmed values are being written.

63.4.1.1 LPIT Memory map

LPIT0 base address: 4102_D000h

LPIT1 base address: 4027_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_0000h
4h	Parameter Register (PARAM)	32	RO	0000_0404h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
8h	Module Control Register (MCR)	32	RW	0000_0000h
Ch	Module Status Register (MSR)	32	W1C	0000_0000h
10h	Module Interrupt Enable Register (MIER)	32	RW	0000_0000h
14h	Set Timer Enable Register (SETTEN)	32	RW	0000_0000h
18h	Clear Timer Enable Register (CLRTEN)	32	WORZ	0000_0000h
20h	Timer Value Register (TVAL0)	32	RW	0000_0000h
24h	Current Timer Value (CVAL0)	32	RO	FFFF_FFFFh
28h	Timer Control Register (TCTRL0)	32	RW	0000_0000h
30h	Timer Value Register (TVAL1)	32	RW	0000_0000h
34h	Current Timer Value (CVAL1)	32	RO	FFFF_FFFFh
38h	Timer Control Register (TCTRL1)	32	RW	0000_0000h
40h	Timer Value Register (TVAL2)	32	RW	0000_0000h
44h	Current Timer Value (CVAL2)	32	RO	FFFF_FFFFh
48h	Timer Control Register (TCTRL2)	32	RW	0000_0000h
50h	Timer Value Register (TVAL3)	32	RW	0000_0000h
54h	Current Timer Value (CVAL3)	32	RO	FFFF_FFFFh
58h	Timer Control Register (TCTRL3)	32	RW	0000_0000h

63.4.1.2 Version ID Register (VERID)

63.4.1.2.1 Offset

Register	Offset
VERID	0h

63.4.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

63.4.1.2.3 Fields

Field	Function
31-24 MAJOR	Major Version Number Returns the major version number for the module design specification. Read-only field.
23-16 MINOR	Minor Version Number Returns the minor version number for the module design specification. Read-only field.
15-0 FEATURE	Feature Number Returns the feature set number. Read-only field.

63.4.1.3 Parameter Register (PARAM)

63.4.1.3.1 Offset

Register	Offset
PARAM	4h

63.4.1.3.2 Function

Provides details on the parameter settings that were used while incorporating this module into the device.

63.4.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EXT_TRIG								CHANNEL							
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

63.4.1.3.4 Fields

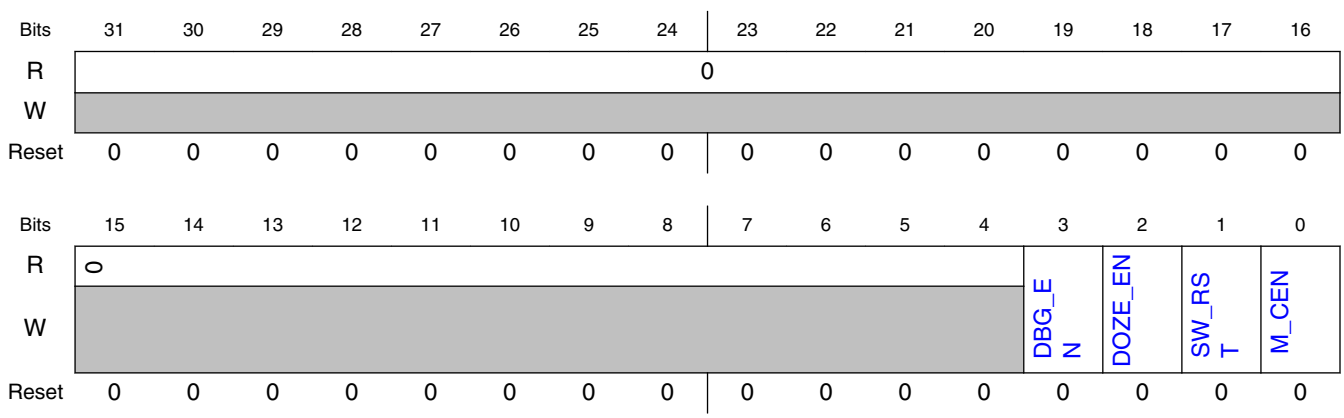
Field	Function
31-16 —	This read-only field is reserved and always has the value 0
15-8 EXT_TRIG	Number of External Trigger Inputs Number of external triggers implemented in this device
7-0 CHANNEL	Number of Timer Channels Number of timer channels implemented in this device

63.4.1.4 Module Control Register (MCR)

63.4.1.4.1 Offset

Register	Offset
MCR	8h

63.4.1.4.2 Diagram



63.4.1.4.3 Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0
3	Debug Enable Bit

Table continues on the next page...

Field	Function
DBG_EN	Stops the timer channels when the device enters Debug mode 0b - Stop timer channels in Debug mode 1b - Allow timer channels to continue to run in Debug mode
2 DOZE_EN	DOZE Mode Enable Bit Stops the timer channels when the device enters DOZE mode 0b - Stop timer channels in DOZE mode 1b - Allow timer channels to continue to run in DOZE mode
1 SW_RST	Software Reset Bit Resets all channels and registers, except the Module Control Register. The Software Reset bit remains set until cleared by software. **NOTE:** Before clearing the Software Reset bit, software must wait for 4 peripheral clocks (for clock synchronization and reset propagation). 0b - Timer channels and registers are not reset 1b - Reset timer channels and registers

0
M_CEN
	Module Clock Enable
	Enables the peripheral clock to the module timers.
	• The Module Clock Enable bit must be asserted when accessing these registers:
	  • Module Status Register (MSR)
	  • Set Timer Enable Register (SETTEN)
	  • Clear Timer Enable Register (CLRTEN)
	  • Timer Value Registers (TVAL*n*)
	  • Current Timer Value Registers (CVAL*n*)
	  • Timer Control Registers (TCTRL*n*)
	• Both the bus clock and peripheral clock must be enabled to allow for clock synchronization and updating the above registers.
	• Acessing the above mentioned registers while the Module Clock Enable bit (M_CEN) = '0', will assert a transfer error for that bus cycle.
	• Writing to Current Timer Value (CVAL*n*) registers and Reserved registers will always generate a transfer error.
	NOTE: There may be additional clock gating bits in your device that gate the peripheral clock to the LPIT module. Ensure that those additional clock gating bits are configured appropriately, to enable the peripheral clock to the LPIT module.
	0b - Disable peripheral clock to timers
	1b - Enable peripheral clock to timers

63.4.1.5 Module Status Register (MSR)

63.4.1.5.1 Offset

Register	Offset
MSR	Ch

63.4.1.5.2 Function

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled), reading or writing the Module Status register will generate a transfer error.

63.4.1.5.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												TIF3	TIF2	TIF1	TIF0
W													W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

63.4.1.5.4 Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0
3 TIF3	Channel 3 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1 • To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out 1b - Timeout has occurred (timer has timed out)
2 TIF2	Channel 2 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1 • To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out 1b - Timeout has occurred (timer has timed out)
1 TIF1	Channel 1 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1

Table continues on the next page...

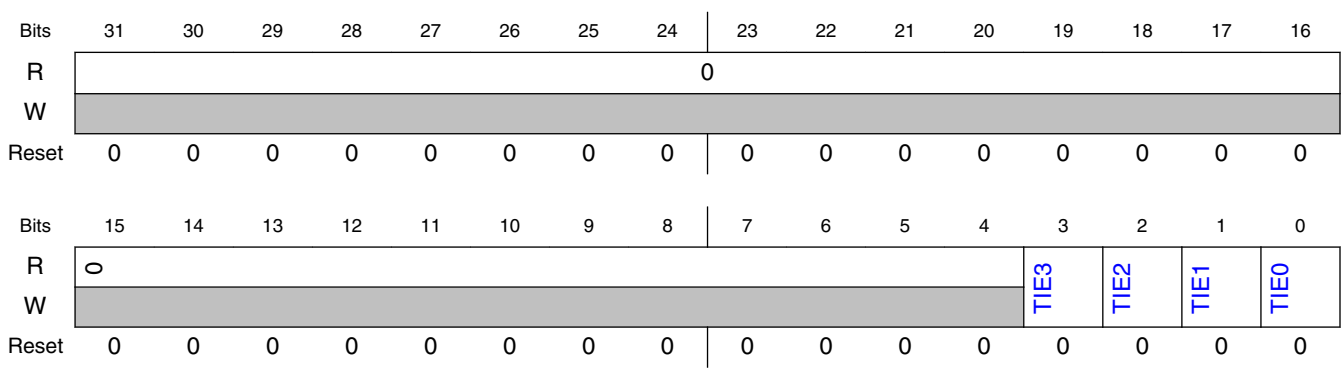
Field	Function
	• To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out 1b - Timeout has occurred (timer has timed out)
0 TIF0	Channel 0 Timer Interrupt Flag • In compare modes: at the end of the timer period, the channel timer interrupt flag is set to 1 • In capture modes: when the trigger asserts, the channel timer interrupt flag is set to 1 • To clear a channel timer interrupt flag, write logic 1 to it • Writing 0 to a channel timer interrupt flag has no effect 0b - Timer has not timed out 1b - Timeout has occurred (timer has timed out)

63.4.1.6 Module Interrupt Enable Register (MIER)

63.4.1.6.1 Offset

Register	Offset
MIER	10h

63.4.1.6.2 Diagram



63.4.1.6.3 Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0

Table continues on the next page...

Field	Function
3 TIE3	Channel 3 Timer Interrupt Enable Enables interrupt generation when: • the Channel 3 Timer Interrupt Enable bit (TIE3) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF3]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled
2 TIE2	Channel 2 Timer Interrupt Enable Enables interrupt generation when: • the Channel 2 Timer Interrupt Enable bit (TIE2) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF2]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled
1 TIE1	Channel 1 Timer Interrupt Enable Enables interrupt generation when: • the Channel 1 Timer Interrupt Enable bit (TIE1) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF1]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled
0 TIE0	Channel 0 Timer Interrupt Enable Enables interrupt generation when: • the Channel 0 Timer Interrupt Enable bit (TIE0) is set to 1 • and if the corresponding Timer Interrupt Flag (MSR[TIF0]) is asserted (=1, timeout has occurred) 0b - Disabled 1b - Enabled

63.4.1.7 Set Timer Enable Register (SETTEN)

63.4.1.7.1 Offset

Register	Offset
SETTEN	14h

63.4.1.7.2 Function

The Set Timer Enable register allows the simultaneous enabling of timer channels.

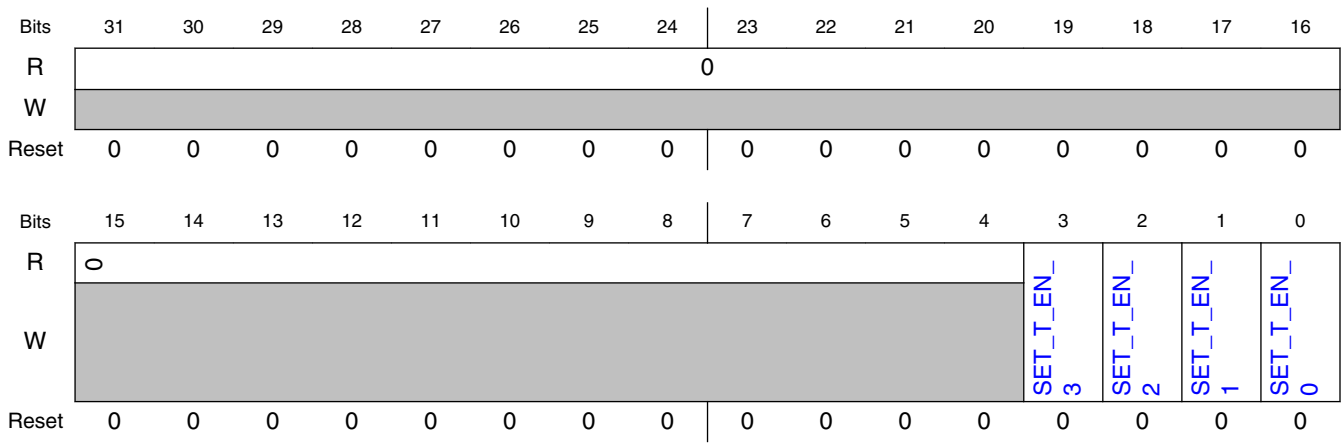
- Timer channels can be enabled by
  - either by writing '1' to Timer Enable bit (T_EN) in the respective TCTRLn register,
  - or by setting the corresponding Set Timer Enable bit (SET_T_EN_*n*) in the Set Timer Enable register (SETTEN).

- To disable timer channels simultaneously, use the Clear Timer Enable register (CLRTEN).
- Writing a '0' to the Set Timer Enable register has no effect.

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled)), reading or writing the Set Timer Enable register will generate a transfer error.

63.4.1.7.3 Diagram



63.4.1.7.4 Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0

3
SET_T_EN_3
	Set Timer 3 Enable
	To enable timer channel 3, write '1' to Set Timer 3 Enable bit.
	• The Set Timer 3 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL3 register.
	• Writing a 0 to Set Timer 3 Enable bit will not disable the counter.
	• The Set Timer 3 Enable bit will be cleared
	  • when Timer Enable bit (TCTRL3[T_EN]) is set to 0
	  • or when '1' is written to the Clear Timer 3 Enable bit (CLRTEN[CLR_T_EN_3]).
	
	0b - No effect
	1b - Enables Timer Channel 3

2 SET_T_EN_2	Set Timer 2 Enable To enable timer channel 2, write '1' to Set Timer 2 Enable bit. • The Set Timer 2 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL2 register.

Table continues on the next page...

Field	Function

	• Writing a 0 to Set Timer 2 Enable bit will not disable the counter.
	• The Set Timer 2 Enable bit will be cleared
	  • when Timer Enable bit (TCTRL2[T_EN]) is set to 0
	  • or when '1' is written to the Clear Timer 2 Enable bit (CLRTEN[CLR_T_EN_2]).
	
	0b - No Effect
	1b - Enables Timer Channel 2

1
 	
SET_T_EN_1
	Set Timer 1 Enable
	
	To enable timer channel 1, write '1' to Set Timer 1 Enable bit.
	• The Set Timer 1 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL1 register.
	• Writing a 0 to Set Timer 1 Enable bit will not disable the counter.
	• The Set Timer 1 Enable bit will be cleared
	  • when Timer Enable bit (TCTRL1[T_EN]) is set to 0
	  • or when '1' is written to the Clear Timer 1 Enable bit (CLRTEN[CLR_T_EN_1]).
	
	0b - No Effect
	1b - Enables Timer Channel 1

0
 	
SET_T_EN_0
	Set Timer 0 Enable
	
	To enable timer channel 0, write '1' to Set Timer 0 Enable bit.
	• The Set Timer 0 Enable bit can be used in addition to the Timer Enable bit (T_EN) in TCTRL0 register.
	• Writing a 0 to Set Timer 0 Enable bit will not disable the counter.
	• The Set Timer 0 Enable bit will be cleared
	  • when Timer Enable bit (TCTRL0[T_EN]) is set to 0
	  • or when '1' is written to the Clear Timer 0 Enable bit (CLRTEN[CLR_T_EN_0]).
	
	0b - No effect
	1b - Enables Timer Channel 0

63.4.1.8 Clear Timer Enable Register (CLRTEN)

63.4.1.8.1 Offset

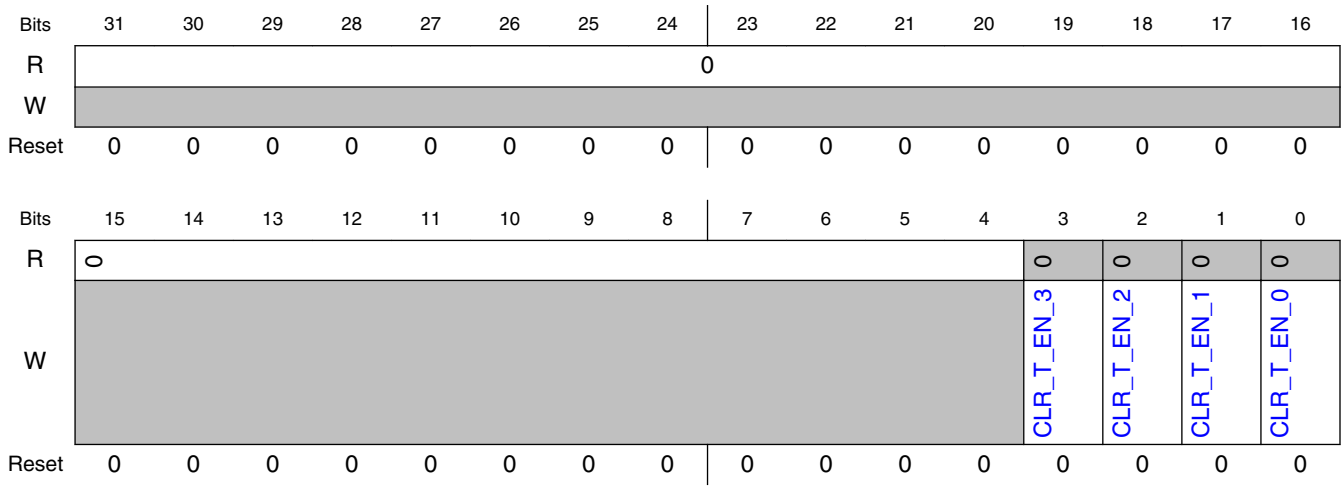
Register	Offset
CLRTEN	18h

63.4.1.8.2 Function

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled), reading or writing the Clear Timer Enable register will generate a transfer error.

63.4.1.8.3 Diagram



63.4.1.8.4 Fields

Field	Function
31-4 —	This read-only field is reserved and always has the value 0
3 CLR_T_EN_3	Clear Timer 3 Enable • To disable timer channel 3, write '1' to Clear Timer 3 Enable bit. • The Clear Timer 3 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL3 register. • Writing a 1 to Clear Timer 3 Enable bit will not enable the counter. • The Clear Timer 3 Enable bit is self-clearing, which means that the Clear Timer 3 Enable bit will always read 0. 0b - No Action 1b - Clear the Timer Enable bit (TCTRL3[T_EN]) for Timer Channel 3
2 CLR_T_EN_2	Clear Timer 2 Enable • To disable timer channel 2, write '1' to Clear Timer 2 Enable bit. • The Clear Timer 2 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL2 register. • Writing a 1 to Clear Timer 2 Enable bit will not enable the counter. • The Clear Timer 2 Enable bit is self-clearing, which means that the Clear Timer 2 Enable bit will always read 0. 0b - No Action 1b - Clear the Timer Enable bit (TCTRL2[T_EN]) for Timer Channel 2
1 CLR_T_EN_1	Clear Timer 1 Enable • To disable timer channel 1, write '1' to Clear Timer 1 Enable bit. • The Clear Timer 1 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL1 register. • Writing a 1 to Clear Timer 1 Enable bit will not enable the counter. • The Clear Timer 1 Enable bit is self-clearing, which means that the Clear Timer 1 Enable bit will always read 0.

Table continues on the next page...

Field	Function
	0b - No Action 1b - Clear the Timer Enable bit (TCTRL1[T_EN]) for Timer Channel 1
0 CLR_T_EN_0	Clear Timer 0 Enable • To disable timer channel 0, write '1' to Clear Timer 0 Enable bit. • The Clear Timer 0 Enable bit can be used in addition to Timer Enable bit (T_EN) in TCTRL0 register. • Writing a 1 to Clear Timer 0 Enable bit will not enable the counter. • The Clear Timer 0 Enable bit is self-clearing, which means that the Clear Timer 0 Enable bit will always read 0. 0b - No action 1b - Clear the Timer Enable bit (TCTRL0[T_EN]) for Timer Channel 0

63.4.1.9 Timer Value Register (TVAL0 - TVAL3)

63.4.1.9.1 Offset

Register	Offset
TVAL0	20h
TVAL1	30h
TVAL2	40h
TVAL3	50h

63.4.1.9.2 Function

- In compare modes: the Timer Value Registers (TVAL*n*) select the timeout period for the timer channels.
- In capture modes: the Timer Value Registers (TVAL*n*) are loaded with the value of the counter when the trigger asserts.

NOTE

Unless the Module Clock Enable bit (MCR[M_CEN]) is set (=1, peripheral clock to timers is enabled), then reading or writing the TVALn register will generate a transfer error.

63.4.1.9.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TMR_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TMR_VAL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

63.4.1.9.4 Fields

Field	Function

31-0
TMR_VAL
	Timer Value
	• In compare mode: Timer Value (TMR_VAL) is the timer channel start value.
	  • The timer will count down until the timer reaches 0, then the timer will generate an interrupt and load the Timer Value register (TVAL*n*) value again.
	  • Writing a new value to the Timer Value register (TVAL*n*) will not restart the timer channel; instead the new value will be loaded *after the timer expires*.
	  • To abort the current timer cycle and start a timer period with a new value, the timer channel must be disabled and enabled *again*.
	• In capture mode: whenever the trigger asserts, the Timer Value register stores the inverse of the counter value.
	
	00000000000000000000000000000000b - Invalid load value in compare mode
	00000000000000000000000000000001b - Invalid load value in compare mode
	00000000000000000000000000000010-11111111111111111111111111111111b - In compare mode: the value to be loaded; in capture mode, the value of the timer

63.4.1.10 Current Timer Value (CVAL0 - CVAL3)

63.4.1.10.1 Offset

Register	Offset
CVAL0	24h
CVAL1	34h
CVAL2	44h
CVAL3	54h

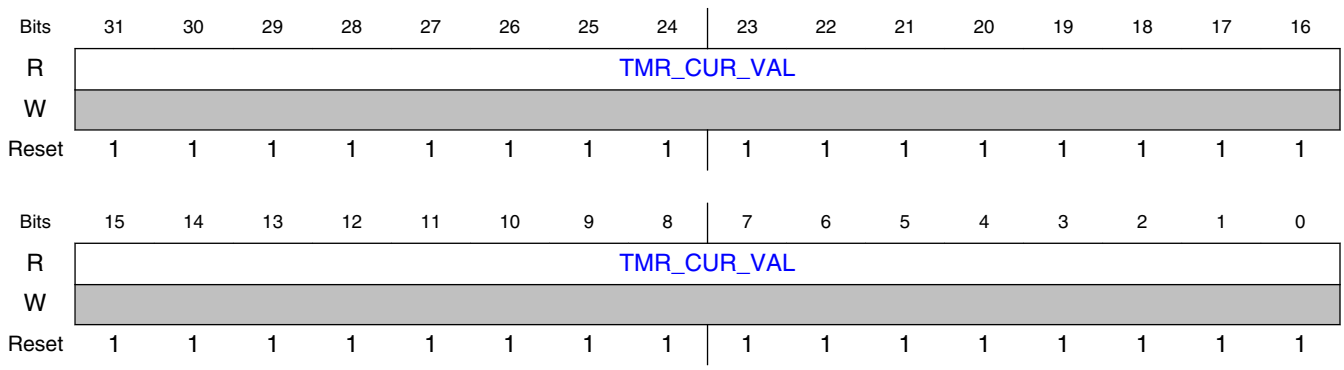
63.4.1.10.2 Function

The Current Timer Value registers indicate the current timer counter value.

NOTE

While the timer is running, CVALn register reads may not return the real value. If the timer value needs to be read, read it during an LPIT interrupt service routine.

63.4.1.10.3 Diagram



63.4.1.10.4 Fields

Field	Function
31-0 TMR_CUR_VAL	Current Timer Value Represents the current timer value, if the timer is enabled.

63.4.1.11 Timer Control Register (TCTRL0 - TCTRL3)

63.4.1.11.1 Offset

Register	Offset
TCTRL0	28h
TCTRL1	38h
TCTRL2	48h
TCTRL3	58h

63.4.1.11.2 Function

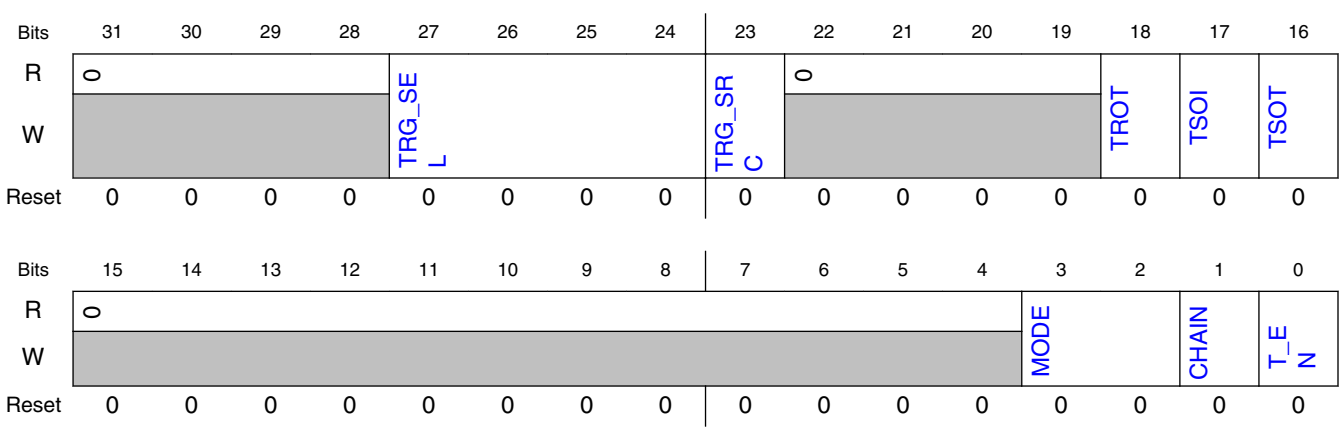
The Timer Control registers contain the control bits for each timer channel.

NOTE

- Unless the peripheral clock to the timers is enabled (Module Clock Enable bit (MCR[M_CEN]) is set (=1)), then reading or writing the Timer Control register will generate a transfer error.
- Timer controls should only be updated when the timer is disabled. In the Timer Control Register TCTRLn, these timer controls include:
  - Trigger Select TRG_SEL
  - Trigger Source TRG_SRC
  - Timer Reload On Trigger TROT
  - Timer Stop On Interrupt TSOI
  - Timer Start On Trigger TSOT
  - Timer Operation Mode MODE
  - Chain Channel CHAIN

  There are 2 ways to disable a timer: write 1 to the specific timer's Clear Timer Enable register bit (CLRTEN[CLR_T_EN_n]) or set the timer enable bit (TCTRLn[T_EN]) = 0 for that channel.

63.4.1.11.3 Diagram



63.4.1.11.4 Fields

Field	Function
31-28	This read-only field is reserved and always has the value 0

Table continues on the next page...

Field	Function
—	
27-24 TRG_SEL	Trigger Select Selects the trigger to use for starting and/or reloading the LPIT timer. • The TRG_SEL field selects one trigger from the set of internal or external triggers that are selected by the Trigger Source bit (TRG_SRC) • Recall that the TRG_SRC bit selects between internal and external trigger signals for each channel **NOTE:** The Trigger Select field should only be changed when the LPIT timer channel is disabled. 0000-0011b - Timer channel 0 - 3 trigger source is selected 0100-1111b - Reserved
23 TRG_SRC	Trigger Source Selects between internal or external trigger sources. The trigger to be used is selected using the TRG_SRC and TRG_SEL bits. Refer to the chip configuration section for available external trigger options. If a channel does not have an associated external trigger, then set the Trigger Source bit (TRG_SRC) = 1. 0b - Selects external triggers 1b - Selects internal triggers
22-19 —	This read-only field is reserved and always has the value 0
18 TROT	Timer Reload On Trigger When set, the LPIT timer will reload when a rising edge is detected on the selected trigger input. The trigger input is ignored if the LPIT is disabled during debug mode (DBGEN = 0) or DOZE mode (DOZE_EN = 0) 0b - Timer will not reload on the selected trigger 1b - Timer will reload on the selected trigger
17 TSOI	Timer Stop On Interrupt Controls whether the channel timer will stop after it (the channel timer) times out or not. 0b - The channel timer does not stop after timeout 1b - The channel timer will stop after a timeout, and the channel timer will restart based on Timer Start On Trigger bit (TSOT). When TSOT = 0, the channel timer will restart after a rising edge on the Timer Enable bit (T_EN) is detected (which means that the timer channel is disabled and then enabled). When TSOT = 1, the channel timer will restart after a rising edge on the selected trigger is detected.
16 TSOT	Timer Start On Trigger Controls when the timer starts decrementing. 0b - Timer starts to decrement immediately based on the restart condition (controlled by the Timer Stop On Interrupt bit (TSOI)) 1b - Timer starts to decrement when a rising edge on a selected trigger is detected
15-4 —	This read-only field is reserved and always has the value 0
3-2 MODE	Timer Operation Mode Configures the channel timer's mode of operation. The MODE bits control how the timer decrements. 00b - 32-bit Periodic Counter 01b - Dual 16-bit Periodic Counter 10b - 32-bit Trigger Accumulator 11b - 32-bit Trigger Input Capture
1 CHAIN	Chain Channel

Table continues on the next page...

Field	Function
	When enabled, the timer channel will decrement when timer channel N-1 trigger asserts. Timer channel 0 cannot be chained. 0b - Channel Chaining is disabled. The channel timer runs independently. 1b - Channel Chaining is enabled. The timer decrements on the previous channel's timeout.
0 T_EN	Timer Enable Enables or disables the Timer Channel 0b - Timer Channel is disabled 1b - Timer Channel is enabled

63.5 Functional description

63.5.1 LPIT programming model

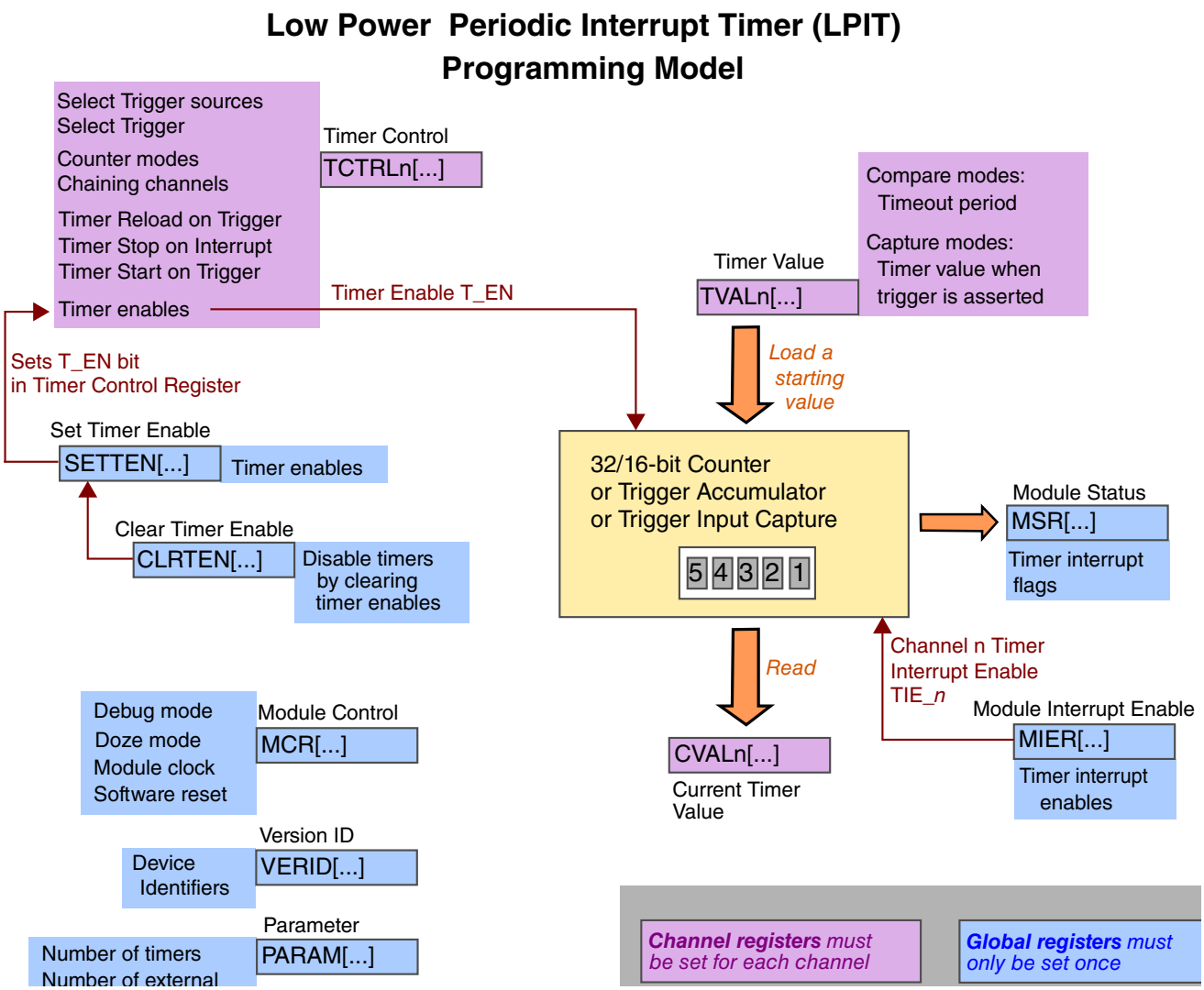


Figure 63-3. Programming Model

Global registers need only be set once:

- Version ID Register (VERID)
- Parameter Register (PARAM)
- Module Control Register (MCR)
- Module Status Register (MSR)
- Module Interrupt Enable Register (MIER)
- Set Timer Enable Register (SETTEN)
- Clear Timer Enable Register (CLRTEN)

Channel registers must be set for each channel:

- Timer Value Register (TVAL0 - TVAL3)

- Current Timer Value (CVAL0 - CVAL3)
- Timer Control Register (TCTRL0 - TCTRL3)

63.5.2 Initialization

Table 63-4. Initializing the LPIT module

Step		How? Why?
1	Enable the peripheral clock	by setting the Module Clock Enable bit (M_CEN) in the Module Control Register (MCR) register. **NOTE:** • Acessing certain registers (MSR, SETTEN, CLRTEN, TVAL, CVAL, and TCTRL) while M_CEN = 0 will lead to assertion of a transfer error for that bus access. • Writing to CVAL and Reserved registers will generate a transfer error. • There might be additional clock gating bits in the device that gate the peripheral clock to this module. When enabling the clock to this module, ensure that software is configuring those additional clock gating bits (in addition to M_CEN bit).
2	Wait for 4 peripheral clock cycles	to allow time for clock synchronization and reset de-assertion.
3	Configure timer control bits	for each timer channel that is to be enabled: • timer mode of operation bits TCTRLn[MODE] • trigger source selection bits TCTRLn[TRG_SEL, TRG_SRC] • trigger control bits TCTRLn[TROT, TSOT, TSOI] **NOTE:** Timer controls should only be updated when the timer is disabled. There are 2 ways to disable a timer: write 1 to the specific timer's Clear Timer Enable register bit (CLRTEN[CLR_T_EN_n]) or set the timer enable bit (TCTRLn[T_EN]) = 0 for that channel.
4	Configure the channels that are to be chained	by setting the CHAIN bit in the corresponding channel's TCTRLn register.
5	Set the timer timeout value	for channels configured in Compare Mode, by programming the appropriate value in TVAL register for those channels.
6	Configure TIEn bits in MIER register	for those channels that are required to generate interrupts on timer timeouts.
7	Configure the low power modes of the module	by setting the DBG_EN and DOZE_EN bits in the MCR register. This is common to all timer channels.
8	Enable the channel timers	by setting the corresponding T_EN bit in the corresponding channel's TCRTLn register.

Also:

- For channels configured in Capture Mode, the timer value can be read from TVALn register when a channel timeout occurs.
- At any time, the current value of the timer for any channel can be read by reading the corresponding channel's CVALn register. The M_CEN bit must be '1' when doing so.
- The timer interrupt flag bits (TIFn) in MSR register get asserted on timer timeout. To clear these timer interrupt flag bits, write '1' to them.

63.5.3 Timer Modes

The timer mode is configured by setting an appropriate value in the MODE bits in TCTRLn register.

Table 63-5. Timer modes that are supported

Timer Mode	Operation
32-bit Periodic Counter	• The counter will load, decrement down to 0 • which will set the timer interrupt flag and assert the output pre-trigger.
Dual 16-bit Periodic Counter	• The counter will load, then the lower 16-bits will decrement down to 0 • which will assert the output pre-trigger. • The upper 16-bits will then decrement down to 0 • which will set the timer interrupt flag and then negate the output pre-trigger.
32-bit Trigger Accumulator	• The counter will load on the 1st trigger rising edge and then decrement down to 0 on each trigger rising edge • which will set the timer interrupt flag and assert the output pre-trigger.

32-bit Trigger Input Capture	• The counter will load with 0xFFFF_FFFF and then decrement down to 0.
	• If a trigger rising edge is detected,
	  • then it will store the inverse of the current counter value in the timer value register
	  • which will set the timer interrupt flag and assert the output pre-trigger.

The timer operation is controlled by Trigger Control bits (TSOT, TSOI, TROT) which control the timer load, reload, start and restart of the timers.

NOTE

- The trigger output is asserted one Peripheral Timer Clock cycle later than pre-trigger output. The trigger output and the pre-trigger output de-assert at the same time.
- The pre-trigger output is asserted for 2 clock cycles and the trigger output is asserted for 1 clock cycle (except in 16-bit Periodic Counter mode, where both pre-trigger and trigger are asserted for many cycles depending on TMR_VAL[31:16]).
- Timer changes *that are based on external triggers* take effect after 4 peripheral clocks after the actual external trigger assertion (due to clock synchronization, rise edge detection and timer update).

63.5.4 Trigger Control for Timers

Various programmable bits control how the trigger inputs and the timer operate. TRG_SEL and TRG_SRC select the timer triggers:

- Trigger Select (TRG_SEL) selects the input trigger for the channel from all of the other channel's trigger outputs.
- Trigger Source (TRG_SRC) selects between the internal trigger and the external trigger inputs to the channel.

The selected trigger affects how the timer operates, using the configuration of the TROT, TSOI and TSOT bits.

Table 63-6. How bits control timer operations

If	=	Then
TSOI *Timer Stop On Interrupt*	1	then the counter stops on a Timer Interrupt flag (MSR[TIFn]) assertion. To reload and decrement, it requires: • a trigger (if TSOT = 1) • a T_EN rising edge (if TSOT = 0)
	0	then the counter does not stop after timeout
TROT *Timer Reload On Trigger*	1	then the counter is loaded on each trigger
	0	then the counter is loaded on every T_EN rising edge or timeout rising edge (timeout not used in Capture modes)
TSOT *Timer Start On Trigger*	1	then the counter will start to decrement on a trigger. Subsequent triggers are ignored until the counter times out.
	0	then the counter decrements immediately on the next clock edge. When channel is Chained or in Capture mode, TSOT has no effect.

In different timer modes, these programmable bits affect the timer operation differently:

Table 63-7. Time modes and programmable bits

Mode	Which programmable bits affect timer operations
32-bit Periodic Counter	All bits (TSOT, TSOI, TROT) affect the timer operation as described previously.
Dual 16-bit Periodic Counter	
32-bit Trigger Accumulator	• Only the TSOI bit controls the timer function. • TROT and TSOT bits have no effect on timer operations.
32-bit Input Trigger Capture	• Only TSOI and TROT bits control the timer function. • TSOT bit has no effect on timer operations.

63.5.5 Channel Chaining

Individual timer channels can be chained together to achieve a larger value of timeout. Chaining the timer channel causes them to work in a *'nested loop'* manner thereby leading to an effective timeout value of $TVAL_{CHn} \times (TVAL_{CHn-1} + 1)$.

The channels are chained by setting the CHAIN bit in corresponding channel's $TCTRL_{CHn}$ register. When a channel is chained, that channel's timer decrements on previous channel's timeout pulse, regardless of the timer mode (MODE bits). The TSOT bit does not have any effect if the channel timer (Channel 'n') is chained to previous channel's timer (Channel 'n-1').

63.5.6 Detailed timing

NOTE

The timing diagrams in these sections are not *cycle-accurate* (i.e., some cycles may not be shown), but the timing diagrams do show the timer channel behavior across several clock cycles.

Table 63-8. Timing diagrams list

Mode		Timing Diagram	TSOT	TROT	TSOI	CHAIN
32-bit periodic counter (compare mode)	00	Figure 63-4	0	0	0	0
		Figure 63-5	0	0	1	0
		Figure 63-6	0	1	0	0
		Figure 63-7	0	1	1	0
		Figure 63-8	1	0	0	0
		Figure 63-9	1	0	1	0
		Figure 63-10	1	1	0	0
		Figure 63-11	1	1	1	0
16-bit dual periodic counter (compare mode)	01	Figure 63-12	0	0	0	0
32-bit trigger accumulator mode	10	Figure 63-13	X	X	0	0
		Figure 63-14	X	X	1	0
32-bit trigger capture mode	11	Figure 63-15	X	0	0	0
		Figure 63-16	X	1	0	0
		Figure 63-17	X	0	1	0
Timer chaining: effects on timing operations	-	Figure 63-18	-	-	-	-

63.5.6.1 Mode=00: 32-bit periodic counter (compare mode)

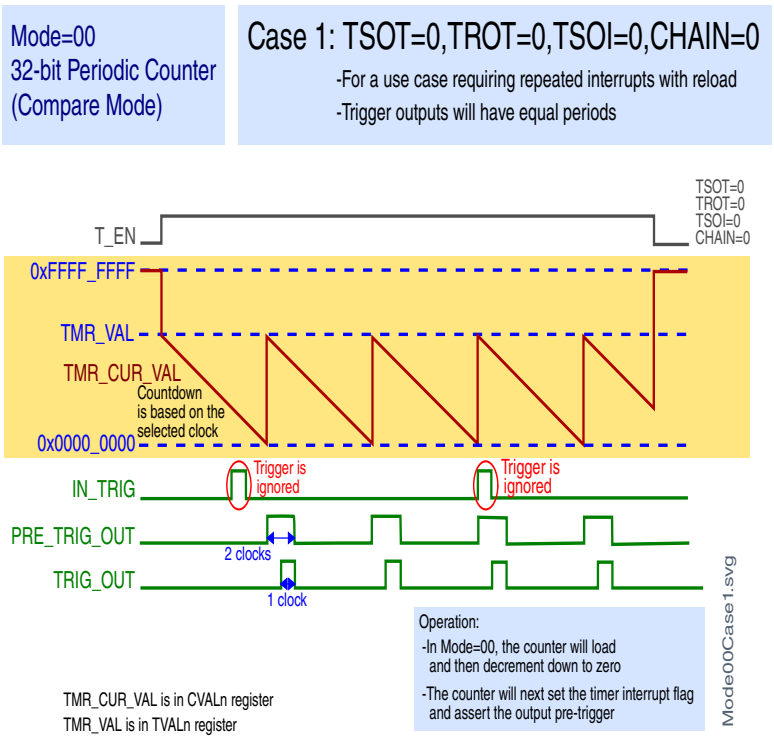


Figure 63-4. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

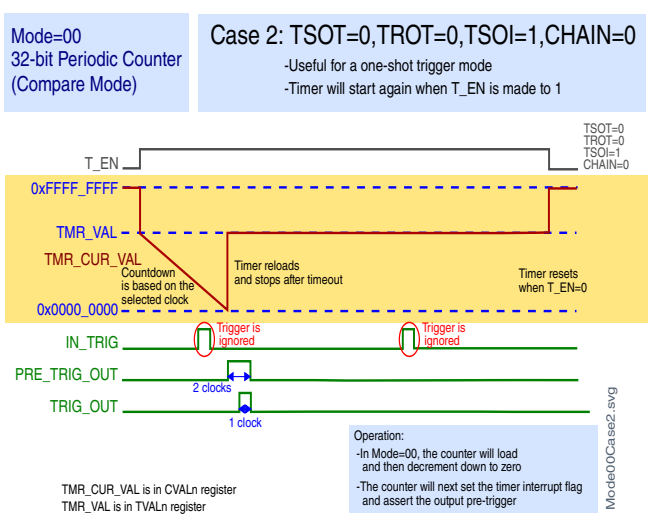


Figure 63-5. Case 2: TSOT = 0, TROT = 0, TSOI = 1, CHAIN = 0

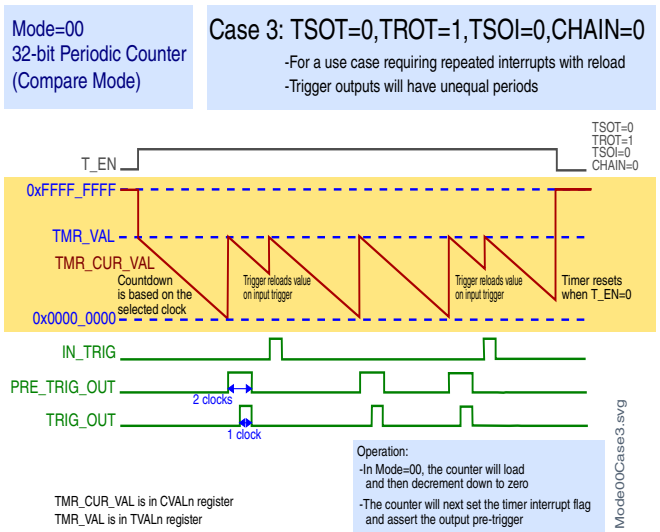


Figure 63-6. Case 3: TSOT = 0, TROT = 1, TSOI = 0, CHAIN = 0

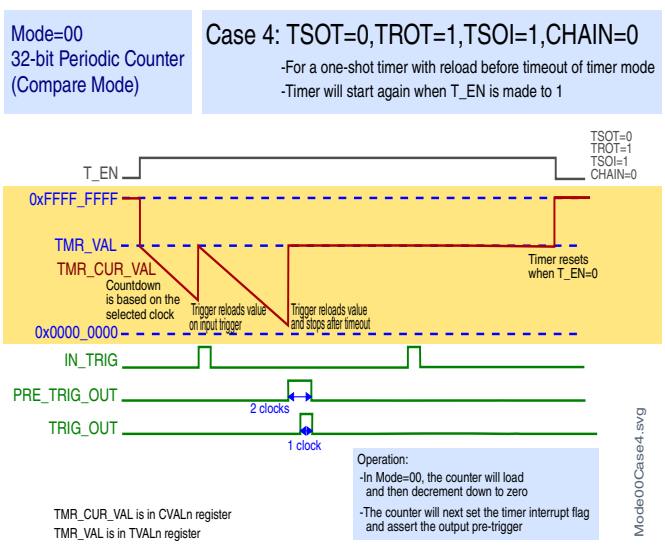


Figure 63-7. Case 4: TSOT = 0, TROT = 1, TSOI = 1, CHAIN = 0

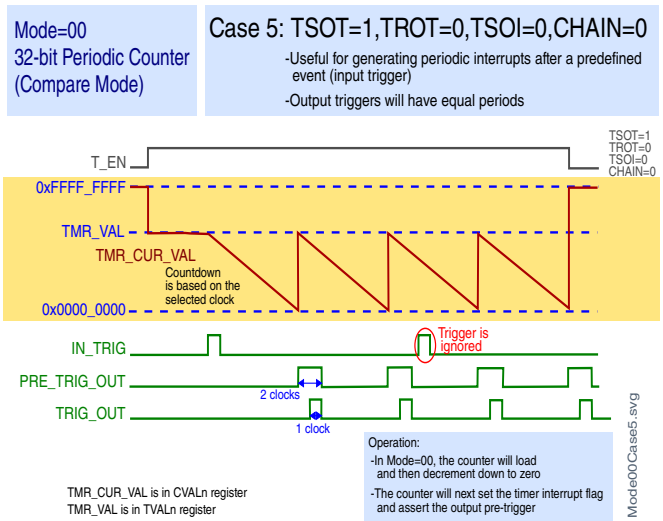


Figure 63-8. Case 5: TSOT = 1, TROT = 0, TSOI = 0, CHAIN = 0

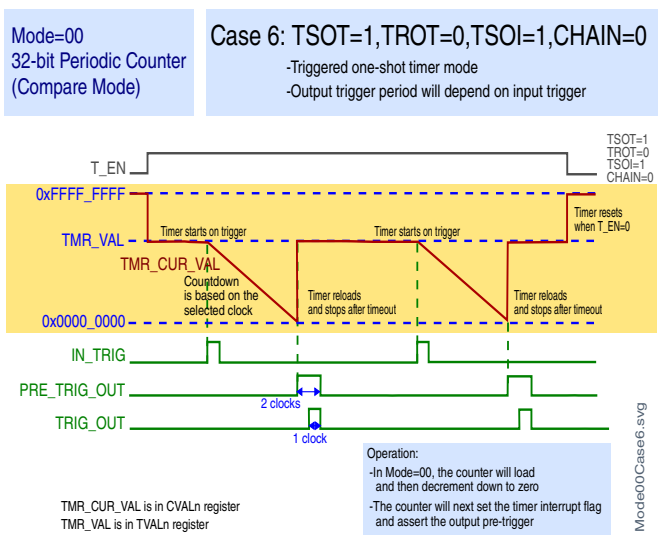


Figure 63-9. Case 6: TSOT = 1, TROT = 0, TSOI = 1, CHAIN = 0

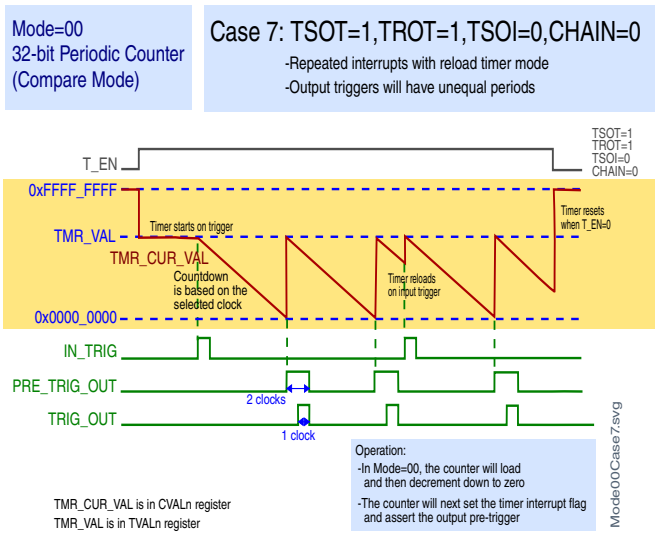


Figure 63-10. Case 7: TSOT = 1, TROT = 1, TSOI = 0, CHAIN = 0

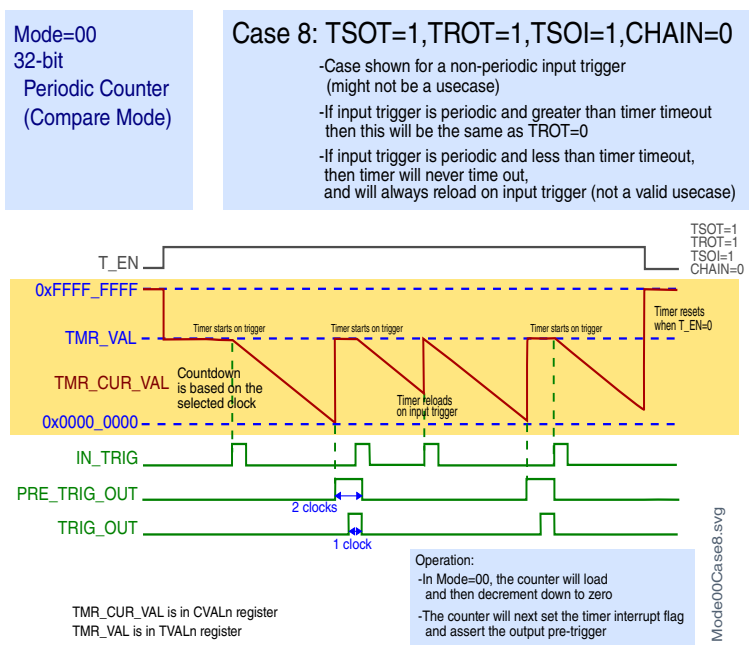


Figure 63-11. Case 8: TSOT = 1, TROT = 1, TSOI = 1, CHAIN = 0

63.5.6.2 Mode=01: 16-bit dual periodic counter (compare mode)

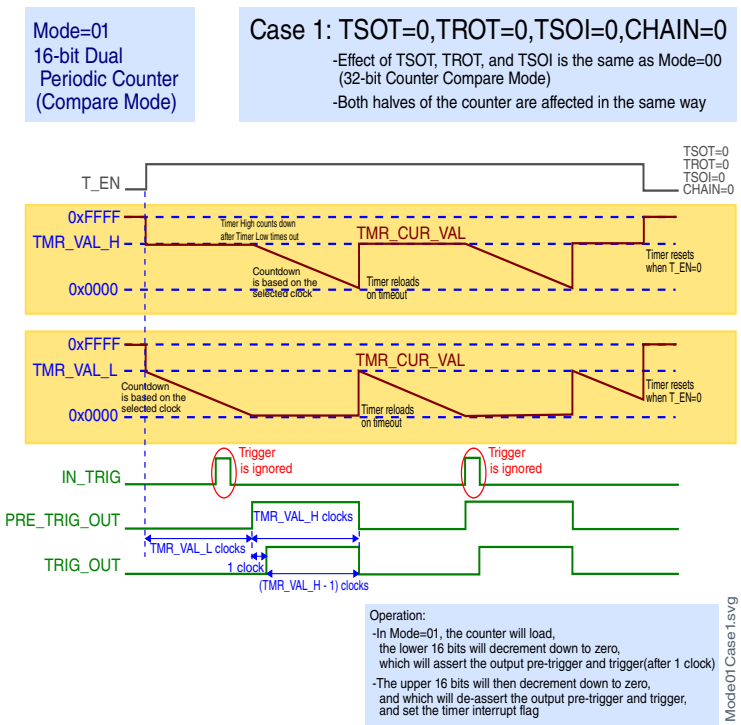


Figure 63-12. Case 1: TSOT = 0, TROT = 0, TSOI = 0, CHAIN = 0

Effect of Timer Control Bits in Mode=01:

- Effect of timer control bits are the same as for Mode=00. Refer to the individual figures of Mode=00.
- The Timer Interrupt (timeout) asserts when {TMR_H,TMR_L} = 0x0000_0000.
- Behavior for Mode = 01 is explained in the next table.

Table 63-9. Mode=01 timer control bits

TSOT	TROT	TSOI	Function	Effect on Timer
0	0	0	• For repeated interrupts with reload • Trigger outputs will have equal periods	Similar to Figure 63-12.
0	0	1	One-shot mode	• Both timers stop after first count down and then time out • Timers will not count again until T_EN is made 1 again • Similar to Figure 63-5.
0	1	0	• For repeated interrupts with reload • Trigger outputs will have unequal periods	• Both timers will reload TMR_VAL on trigger rise edge • Output triggers will clear on reload, if asserted • Similar to Figure 63-6.

Table continues on the next page...

Table 63-9. Mode=01 timer control bits (continued)

TSOT	TROT	TSOI	Function	Effect on Timer
0	1	1	Reloadable one-shot mode	• If a trigger occurs before timeout, then both timers reload and count down (as shown); the timers stop after timeout • A trigger assertion after timeout simply reloads TMR_VAL into the timers • The timers will not count again until T_EN is set to 1 again • Similar to Figure 63-7.
1	0	0	• For generating periodic interrupts after a predefined event (input trigger) • Output triggers will have equal periods	After T_EN rises, the timers do not start until the first trigger's rising edge • Subsequent triggers will have no effect • Similar to Figure 63-8.
1	0	1	• Triggered one-shot timer mode • Output trigger period will depend on input trigger	After T_EN rises, the timers do not start until the first trigger's rising edge. • The timer stops counting after a timeout assertion (a timeout is asserted) • The timer does not start counting again until a new trigger's rising edge is detected • Similar to Figure 63-9.
1	1	0	• For repeated interrupts with reload timer mode • Output triggers will have unequal periods	After T_EN rises, the timers do not start until the first trigger's rising edge • Subsequent triggers will cause the timer to reload TMR_VAL into both counters • The output triggers will clear after a reload, if asserted • Similar to Figure 63-10.
1	1	1	• For a non-periodic input trigger • If input trigger is periodic and greater than timer timeout, then this will be the same as TROT=0 (which is triggered one-shot timer mode; output trigger period will depend on input trigger)	After T_EN rises, the timers do not start until the first trigger's rising edge • The timers stops counting after a timeout assertion (a timeout is asserted) • A trigger's rising edge will cause the timers to reload and then count down • Similar to Figure 63-11.

63.5.6.3 Mode=10: 32-bit trigger accumulator mode

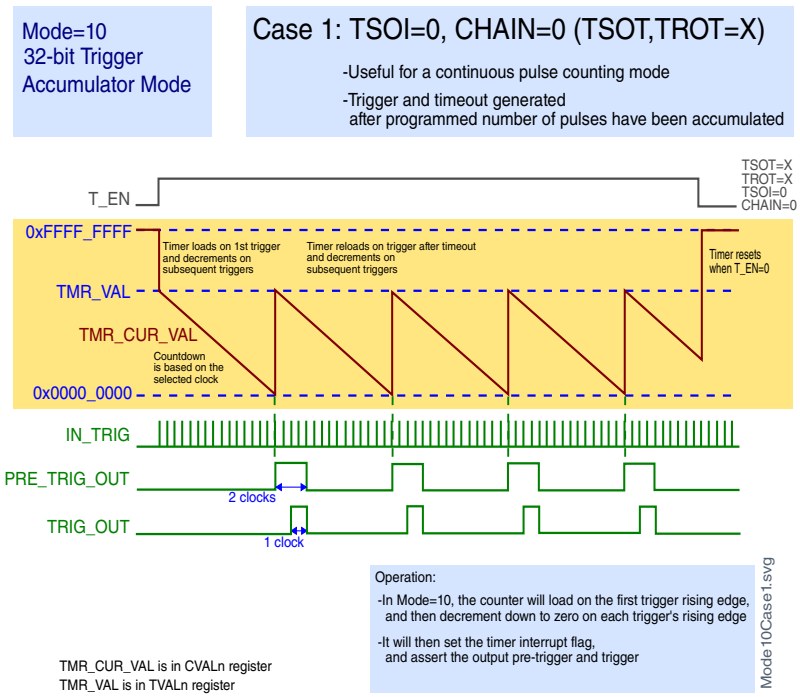


Figure 63-13. Case 1: TSOT = X, TROT = X, TSOI = 0, CHAIN = 0

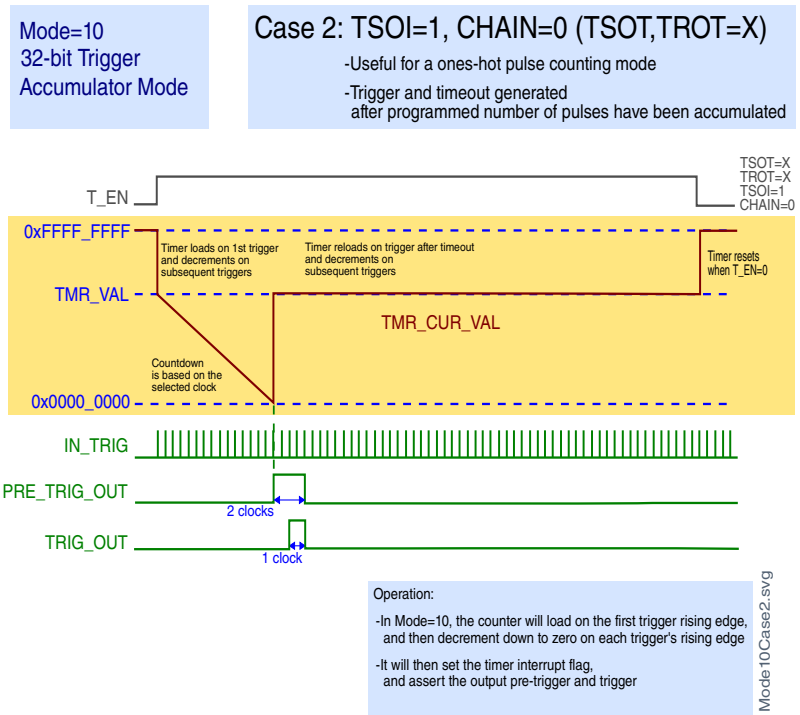


Figure 63-14. Case 2: TSOT = X, TROT = X, TSOI = 1, CHAIN = 0

63.5.6.4 Mode=11: 32-bit trigger capture mode

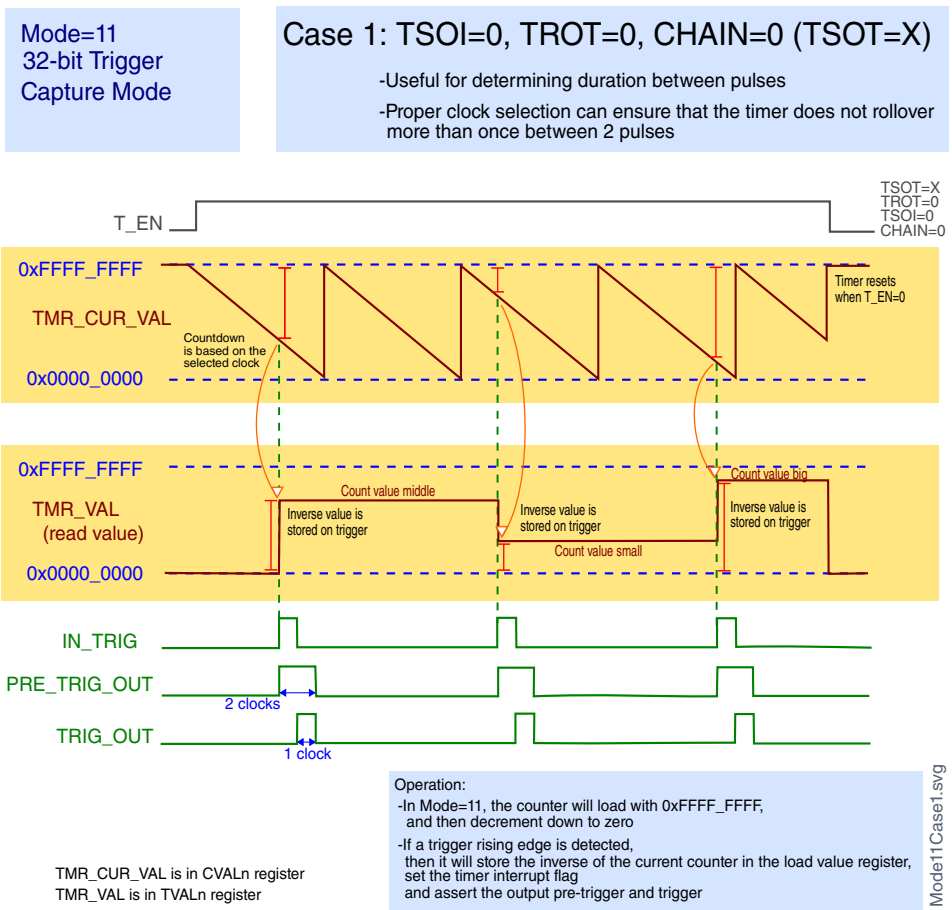


Figure 63-15. Case 1: TSOT = X, TROT = 0, TSOI = 0, CHAIN = 0

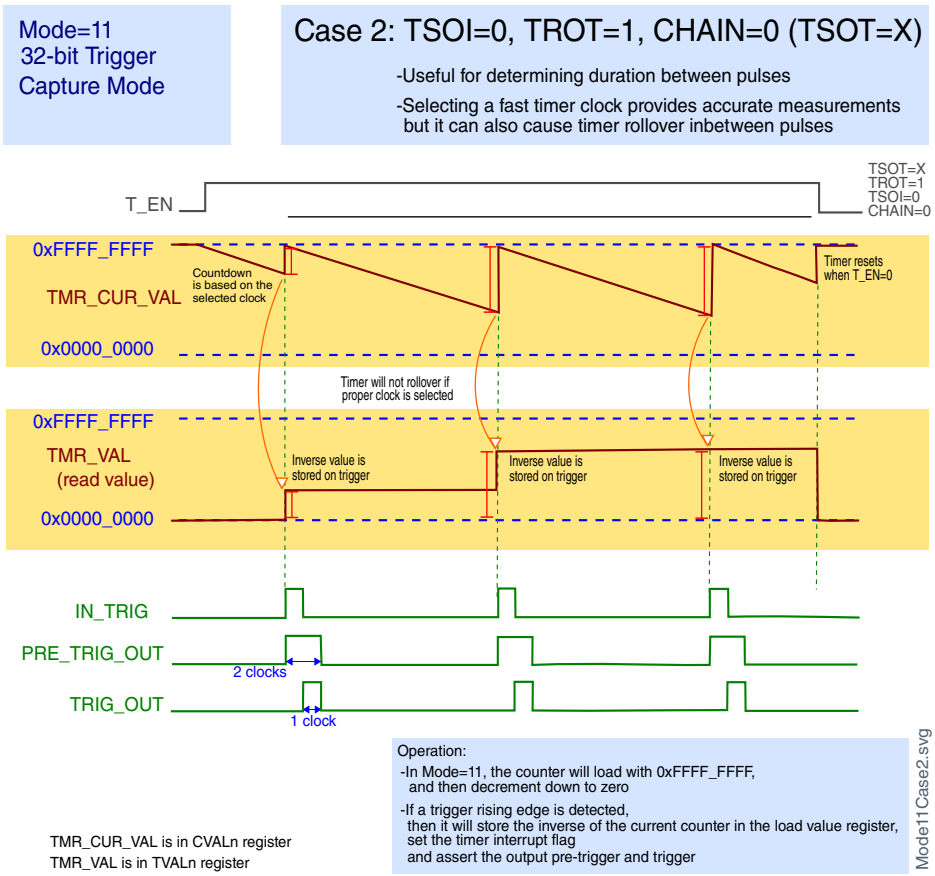


Figure 63-16. Case 2: TSOT = X, TROT = 1, TSOI = 0, CHAIN = 0

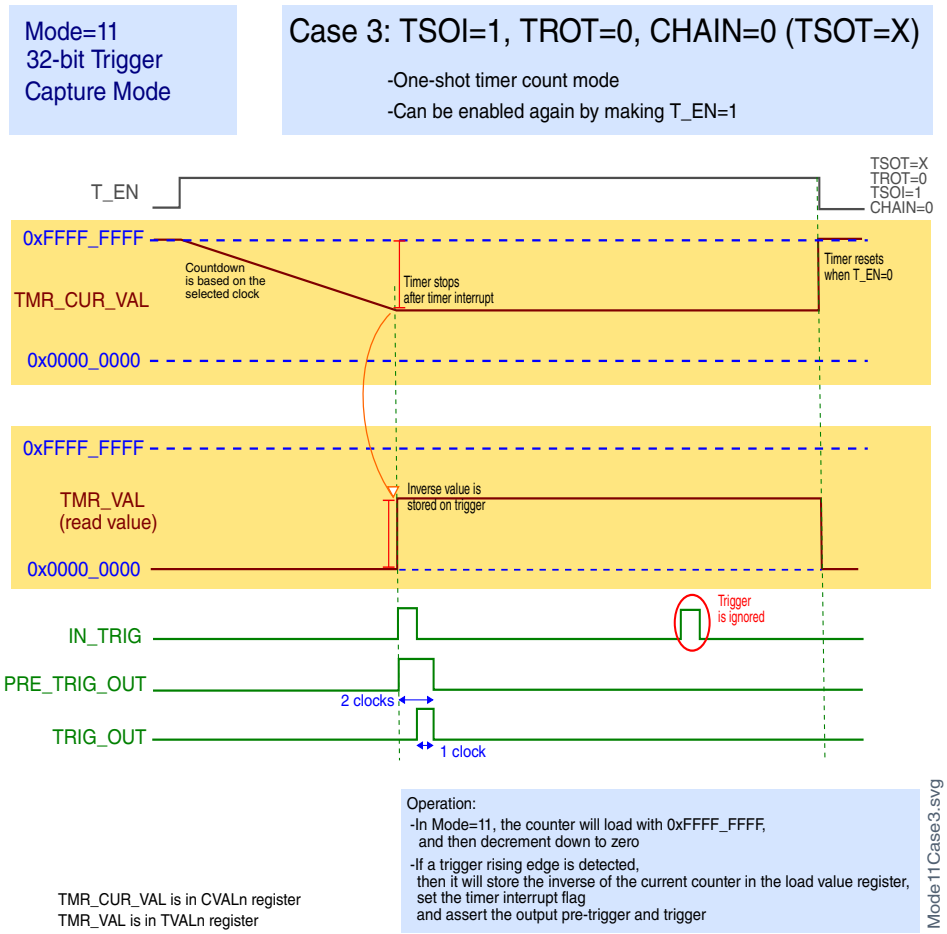


Figure 63-17. Case 3: TSOT = X, TROT = 0, TSOI = 1, CHAIN = 0

Case 4: TSOT = X, TROT = 1, TSOI = 1, CHAIN = 0

Same as previous case (Case 3), except that the timer reloads to 0xFFFF_FFFF and then stops. The timer does not start until T_EN is made 1 again. Case 4 is not a very useful case, because Case 3 covers this.

63.5.6.5 Timer chaining

Effect of Chaining	-Chaining causes Timer “n” to decrement on every timeout pulse (trigger output pulse) from Timer “n – 1”, regardless of what mode is configured in Timer “n” -Timers “n” and “n – 1” effectively form a larger width timer (64-bits) -More than 2 timer channels (or all timer channels) can be chained -It is preferred to have the same trigger source and timer controls configured for chained channels

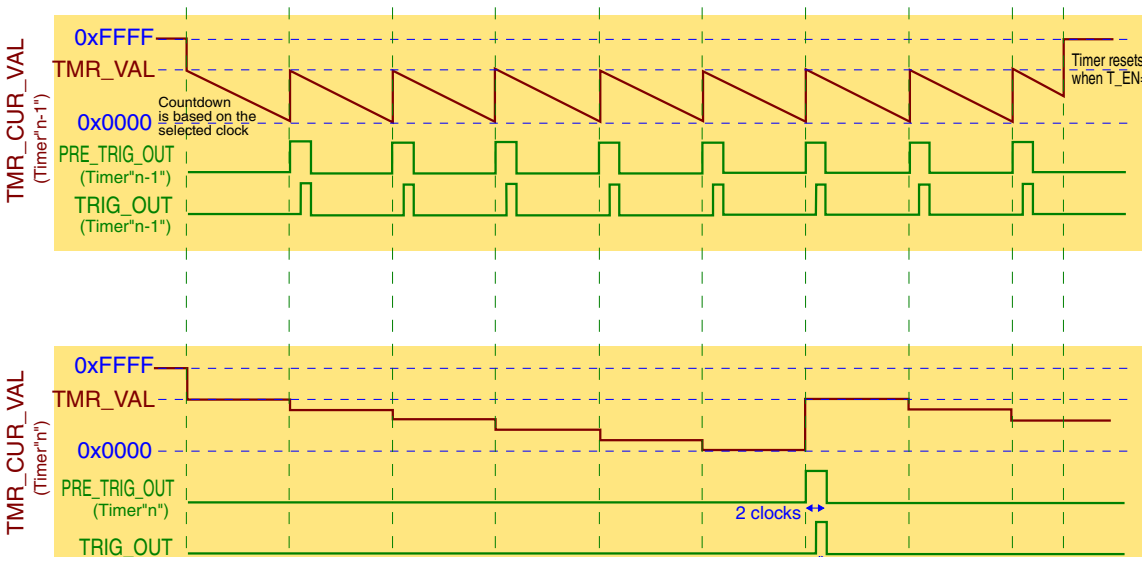


Figure 63-18. Chaining Effects

Chapter 64
Secure Non-Volatile Storage (SNVS)

64.1 Chip-specific SNVS information

Table 64-1. Reference links to related information

Topic	Related module	Reference
Full description	SNVS	SNVS
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

64.1.1 Secure Non-Volatile Storage (SNVS)

The SNVS module is designed to safely hold security-related data such as cryptographic key, time counter, monotonic counter, and general purpose security information. A part of the SNVS module belongs to the VBAT domain, that has its own dedicated power supply which is always on. This enables SNVS to keep this data valid and continue to increment the time counter when the power goes down in the rest of the device. The always-powered-up part of the module is isolated from the rest of the logic to ensure that it does not get corrupted when the device is powered down. The following table shows the configuration of the SNVS.

NOTE

The chapter may contain references to security-only topics. For complete SNVS chapter including security details, see the i.MX 7ULP Security Reference Manual

Table 64-2. SNVS configuration

Parameter	Description
Name	Secure Non-Volatile Storage (SNVS)
Instances	1
Features[1]	• Security state • Security violation inputs • Non-secure RTC • Clock monitor • Zeroizable master: 256-bit • General purpose register: 256-bit • Secure RTC • Secure Monotonic Counter • LP voltage monitor input • External tamper detect inputs: 1 • System power fail detector input • Power-on reset • Tamper input glitch filters
Interface speed	NA
External I/O pins	See the IOMUXC spreadsheet attached to i.MX 7ULPRM for pin details. SNVS/RTC Pins (MISC Pins Tab) and SEC_VIO_B Security Violation Input (IO Signal Table).

1. The complete chapter including all security features is available only in i.MX 7ULP Security Reference Manual

64.2 SNVS introduction

SNVS is a companion module to the CAAM module.

SNVS incorporates both security and non-security functionality. The SNVS non-security functionality is described in this document, but the SNVS security functionality is described only in the Security Reference Manual.

SNVS non-security functions:

- Realtime Counter (RTC) - a software accessible realtime counter
  - RTC can be set to the value in the SRTC
- Periodic Interrupt - a hardware-generated interrupt that occurs periodically at software-specified frequency
- General Purpose Register - a set of registers used to hold 256 bits of data specified by software

  - If the SNVS_LP power input is connected to an uninterrupted power supply, e.g. a coin cell battery, the GPR value is maintained when main SoC is powered off
- Chip power-on/power-off - If the SNVS_LP power input is connected to an uninterrupted power supply and the Power On button input signal is connected to a power button external to the chip, logic within SNVS_LP can be used to wake the chip from a power down.

64.2.1 SNVS feature list

The following table summarizes the features of SNVS:

Table 64-3. SNVS feature list

Feature	Description	Links for Further Information
Real time counter (RTC)	• The RTC is driven by a dedicated clock, which is off when the system power is down. • Programmable time alarm interrupt • Periodic interrupt can be generated with software-selected frequency.	SNVS_HP Real Time Counter
General-purpose register	• The general-purpose register is available to software to store 256 bits of data. • The general-purpose register is zeroized when a security violation is detected. • If the SNVS_LP power input is connected to an uninterrupted power supply (see SNVS power domains), the general-purpose register value is retained even if the main chip is powered down.	Using the General-Purpose Register
Register access restrictions	• Some registers/values can be written only once per boot cycle.	privileged and non-privileged registers
Wakeup from power off	• Input signal from off chip requests SNVS_LP to power on the main SoC (Assuming that the SNVS_LP power input is connected to an uninterrupted power supply (see SNVS power domains). • Hardware debounces the input signal using software-specified signal bounce characteristics	LP Wake-Up Interrupt Enable

64.2.2 SNVS functional description

SNVS implements several non-security features that involve software interaction:

- reading or writing the Realtime Counter (RTC) (This is a non-privileged operation.) - software can also instruct SNVS to load the current SRTC value into the RTC
- reading or writing the General Purpose Register (GPR) (Note that there may be a significant delay when reading or writing registers in the LP section if the LP clock is different from the HP clock.)

The following sections describe in more detail the operation of SNVS.

64.3 SNVS Structure

SNVS is organized as two major sub-modules:

- Low-Power Section of SNVS (SNVS_LP)

  The SNVS_LP section provides hardware that enables secure storage and protection of sensitive data. The SNVS module is designed to safely hold security-related data such as cryptographic key, time counter, monotonic counter, and general purpose security information.

  The SNVS_LP block implements the following functional units:
  - Control and Status Registers
  - General Purpose Registers

  When the LP section is powered by a backup battery the state of these registers is maintained even when the main chip power is off. (see SNVS power domains)

- High-Power Section of SNVS (SNVS_HP)

  The SNVS_HP section contains all SNVS status and configuration registers. It implements all features that enable system communication and provisioning of the SNVS_LP section.

  The SNVS_HP provides an interface between SNVS_LP and the rest of the system.

  The SNVS_HP block implements the following functional units:
  - IP Bus Interface
  - SNVS_LP Interface
  - Zeroizable Master Key Programming Mechanism
  - Real Time Counter with Alarm Control and Status Registers
  - Control and status registers

  SNVS_HP is in the chip's power supply domain and thus receives power along with the rest of the chip.

The following figure illustrates the structure of SNVS.

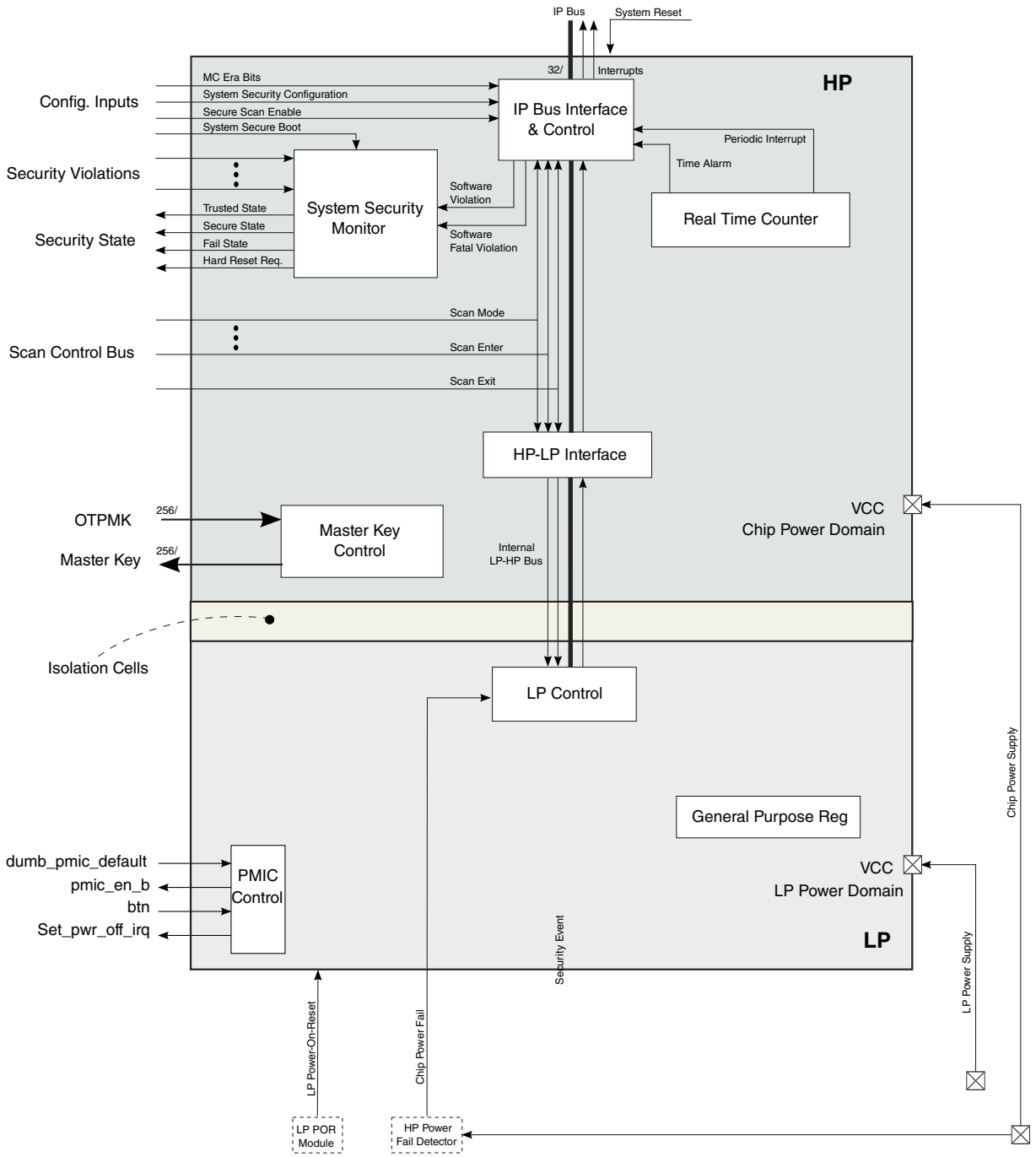


Figure 64-1. SNVS Block Diagram

64.3.1 SNVS power domains

In some versions of SNVS (including this version), the LP (Low Power) section is implemented in an independent power domain from the HP (High Power) section, and most other logic on the chip. Throughout the SNVS documentation whenever mention is

made of "always-on" logic, this assumes a version of SNVS that implements an independent power domain for the LP section, and that the power for this section is supplied by an uninterrupted power supply. The purpose for the independent power domain is so that data can be retained and certain logic can remain functional even when the main chip logic is powered down. But this is possible only if the LP domain remains powered via an uninterrupted power supply when the main chip power domain is powered off. Usually this uninterrupted power supply would be a coin-cell battery, with possibly some power management logic to power the LP section from main power (and perhaps recharge the coin cell battery) when main power is on, and switch to coin-cell power when the main power is off. In versions of SNVS with an independent LP power domain the LP section can be electrically isolated from the rest of the chip logic to ensure that its logic does not get corrupted when the main chip is powered down. If the battery runs down or is removed, an LP POR will occur when the LP section next powers up. Note that some OEMs may choose to connect LP power to HP/main chip power and dispense with a coin cell battery. In that case the SNVS will operate the same as an SNVS without an independent LP power domain. No state will be retained in the LP section when the chip is powered down, and an LP POR will occur whenever there is an HP POR.

64.3.2 Power glitch detector (PGD)

SNVS_LP incorporates a mechanism to detect glitches on the SNVS_LP power supply that might cause the LP control, status, and secure counter values to change. The mechanism works as follows:

1. The PGD register (LPPGDR) is loaded with the known specific value 4173_6166h as part of the SNVS initialization process.
2. Subsequently, this register's value is continuously compared to the hardwired value 4173_6166h.
3. If the comparison indicates that any bit has changed, a power glitch violation is asserted.

Power glitch detection is always enabled and cannot be disabled. At LP POR this register is reset to all 0's, so the hardwired comparison fails and a power glitch violation is reported. Therefore, before programming any feature in the SNVS the power glitch violation should be cleared. The initialization software should write the proper value (4173_6166h) into LPPGDR (see SNVS_LP Power Glitch Detector Register (LPPGDR)) and should then clear the power glitch record in the LP status register (see SNVS_LP Status Register (LPSR)).

The following figure shows the PGD mechanism.

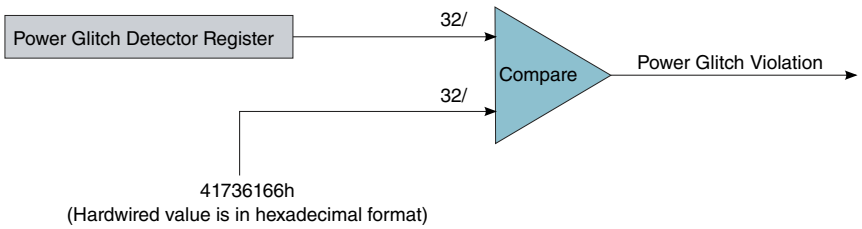


Figure 64-2. Power glitch detector

64.3.3 SNVS clock sources

The SNVS has the following clock sources:

- System peripheral clock input. This clock is used by the SNVS's internal logic, for example, the Security State Machine. This clock can be gated outside of the module when the SNVS indicates that it is not in use.
- HP RTC clock. This clock is used by SNVS_HP real-time counter. This clock does not need to be synchronous with other clocks.

64.4 Runtime Procedures

SNVS implements a number of features that are intended to be accessed by software at runtime (as opposed to accessed at boot time). These features include:

- Real Time Clock (see SNVS_HP Real Time Counter)
- General Purpose Register (see Using the General-Purpose Register)

Procedures for using these features are described in the following sections.

64.4.1 Using SNVS Timer Facilities

SNVS incorporates timer facilities that can optionally generate an interrupt at a specified time. As described in the following sections, SNVS_HP incorporates a Real Time Counter that is available for general use, and SNVS_LP incorporates a Secure Real Time Counter intended for security applications.

64.4.1.1 SNVS_HP Real Time Counter

SNVS_HP implements a real time counter that can be read or written by any application; it has no privileged software access restrictions. When the chip is powered down the RTC is not active and it is reset at chip POR. The RTC can be used to generate a functional interrupt request either at a specific time, or at a specific frequency, or both. To generate an interrupt request at a specific time HPTA_EN is set to 0, the desired time is written to HPTA_MS and HPTA_LS and then HPTA_EN is set to 1. HPTA_EN, HPTA_MS and HPTA_LS can be written by any software that has access to SNVS registers; there are no privileged access restrictions. The counter can be synchronized to the SNVS_LP SRTC by writing to the HP_TS bit of SNVS_HP Control Register. This is particularly useful if the SNVS_LP is powered from an uninterrupted power source (e.g. a coin cell battery) because the RTC can then be set from a chip-internal time source.

64.4.1.2 RTC/SRTC control bits setting

All SNVS registers are programmed from the register bus, consequently any software-initiated changes are synchronized with the IP clock. Several registers can also change synchronously with the RTC/SRTC clock after they are programmed. To avoid IP clock and RTC/SRTC clock synchronization issues, the following values can be changed only when the corresponding function is disabled.

Table 64-4. RTC/SRTC synchronized values list

Function	Value/register	Control bit setting
HP section		
HP Real Time Counter	HPRTCMR and HPRTCLR Registers	RTC_EN = 0 : HPRTCMR/HPRTCLR **can** be programmed RTC_EN = 1 : HPRTCMR/HPRTCLR **cannot** be programmed
HP Time Alarm	HPTAMR and HPTALR Registers	HPTA_EN = 0 : HPTAMR/HPTALR **can** be programmed HPTA_EN = 1 : HPTAMR/HPTALR **cannot** be programmed
LP section		
LP Secure Real Time Counter	LPRTCMR and LPRTCLR Registers	SRTC_ENV = 0 : LPRTCMR/LPRTCLR **can** be programmed SRTC_ENV = 1 : LPRTCMR/LPRTCLR **cannot** be programmed
LP Time Alarm	LPTAR Register	LPTA_EN = 0 : LPTAR **can** be programmed LPTA_EN = 1 : LPTAR **cannot** be programmed

Use the following steps to program synchronized values:

1. Check the enable bit value. If set, clear it.

2. Verify that the enable bit is cleared. There are two reasons to verify the enable bit's setting:

- Enable bit clearing does not happen immediately; it takes three IP clock cycles and two RTC/SRTC clock cycles to change the enable bit's value.
- If the enable bit is locked for programming, it cannot be cleared.

3. Program the desired value.

4. Set the enable bit; it takes three IP clock cycles and two RTC/SRTC clock cycles for the bit to set.

NOTE

Incrementing the value programmed into RTC/SRTC registers by two compensates for the two RTC/SRTC clock cycle delay that is required to enable the counter.

64.4.1.3 Reading RTC and SRTC values

Software should follow the following procedure to ensure that it has read correct data from the RTC (HPRTCMR and HPRTCLR) and SRTC (LPSRTCMR and LPSRTCLR) registers:

- Read the most-significant half and the least-significant half of the RTC/SRTC and then read both halves again. If the values read are the same both times, the value is correct.
- If the two consecutive pairs of reads yield different results, perform two more reads.

The worst case scenario may require three sessions of two consecutive pairs of reads. There are several reasons that the values may be incorrectly read initially:

- Synchronization issues between the RTC/SRTC clock and the system clock
- Since the counter continues to increment, there may be a carry from the least-significant 32-bits to the most-significant bits in between reading the two halves of the counter

64.4.2 Using Other SNVS Registers

The sections below describe how to use the General Purpose Register.Monotonic Counter.The sections below describe how to use the General Purpose Register and the Monotonic Counter.

64.4.2.1 Using the General-Purpose Register

SNVS implements a 256-bit general-purpose register allows software to store a small amount of data. To maintain backward compatibility with versions of SNVS that implement only a 32-bit general purpose register, the most-significant word of the general purpose register is aliased to the original legacy address, and to maintain backward compatibility with versions of snvs_module_name that implement a 128-bit general purpose register, the most-significant half of the general purpose register is aliased to the previous legacy address address. The data in the GPR will be retained during system power-down mode as long as the SNVS_LP remains powered by an uninterrupted power source (e.g. coin cell battery).

64.5 Reset and Initialization of SNVS

SNVS is implemented in two sections (HP and LP) that both must be initialized by software. If the SNVS_LP is powered by an uninterrupted power source that is separate from main SoC power, then SNVS can operate in either of two modes, depending upon whether the main SoC power is on or off. During main SoC power-down SNVS_HP is powered-down, but SNVS_LP is powered from the backup power supply and is electrically isolated from the rest of the chip. In this mode SNVS_LP keeps its registers' values and monitors the SNVS_LP tamper detection inputs, but the LP registers cannot be read or written. During main SoC power-up the isolation of SNVS_LP is disabled and both SNVS_HP and SNVS_LP are powered from the main SoC power. Both LP and HP registers can be read and written (locks and privilege modes permitting). Signals between the SNVS_HP and SNVS_LP sections are enabled and all SNVS functions are operational.

Since the HP and LP sections reside in different power domains, the POR for the two sections can occur at different times. If the SNVS_LP section remains powered by an uninterrupted power source when the main SoC power is off, SNVS_LP is initialized rarely, typically once when the device is first powered on and again whenever the battery is replaced. During main SoC power-up the isolation of SNVS_LP is disabled and both SNVS_HP and SNVS_LP are powered from the main SoC power. Signals between the

SNVS_HP and SNVS_LP sections are enabled and all SNVS functions are operational. The SNVS_HP section is powered from the main SoC power, so it must be initialized after the device is powered on.

- Initializing the LP section
  - The following steps should be completed to properly initialize the SNVS LP section (required at SoC POR):
    - Software should write the proper initialization value (41736166h) into the LP Power Glitch Detector Register and clear the power glitch record in the LP status register. See Power glitch detector (PGD) for more details.
- Initializing the HP section
  - The following steps should be completed to properly initialize the SNVS HP section (required on HP POR, i.e. SoC POR):
    - Perform normal boot to put the SNVS into a functional state (Non-secure, Trusted, Secure) (see HP Command Register, SSM_ST bitfield).
    - Program SNVS general functions/configurations (see HP Control Register).
    - Set lock bits. The MS bit of the HP Lock Register should be set before starting any functional operation. See i.MX 7ULP Security Reference Manual for details.

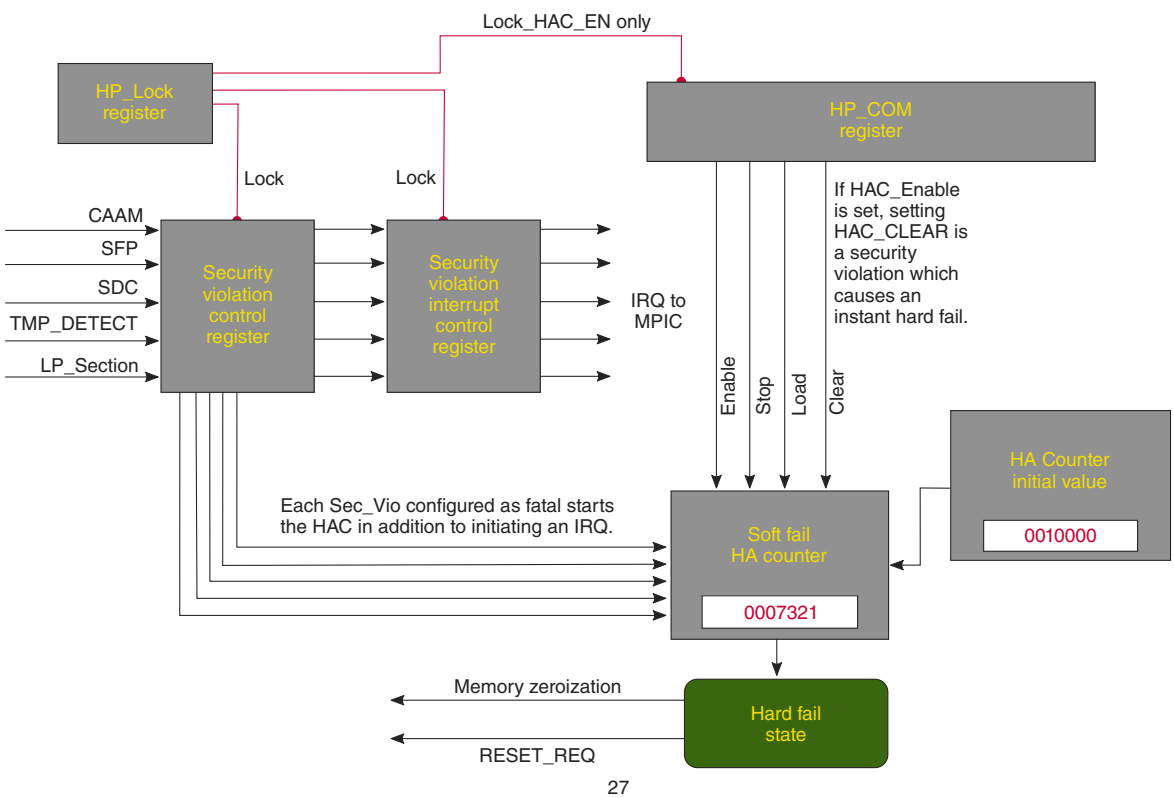


Figure 64-3. Relationship Between the Registers

64.5.1 Initialization Checklists

SNVS is implemented in two sections (HP and LP) that both must be initialized by software. If the SNVS_LP is powered by an uninterrupted power source that is separate from main SoC power, then SNVS can operate in either of two modes, depending upon whether the main SoC power is on or off. During main SoC power-down SNVS_HP is powered-down, but SNVS_LP is powered from the backup power supply and is electrically isolated from the rest of the chip. In this mode SNVS_LP keeps its registers' values and monitors the SNVS_LP tamper detection inputs, but the LP registers cannot be read or written. During main SoC power-up the isolation of SNVS_LP is disabled and both SNVS_HP and SNVS_LP are powered from the main SoC power. Both LP and HP registers can be read and written (locks and privilege modes permitting). Signals between the SNVS_HP and SNVS_LP sections are enabled and all SNVS functions are operational.

Since the HP and LP sections reside in different power domains, the POR for the two sections can occur at different times. If the SNVS_LP section remains powered by an uninterrupted power source when the main SoC power is off, SNVS_LP is initialized rarely, typically once when the device is first powered on and again whenever the battery is replaced. During main SoC power-up the isolation of SNVS_LP is disabled and both SNVS_HP and SNVS_LP are powered from the main SoC power. Signals between the SNVS_HP and SNVS_LP sections are enabled and all SNVS functions are operational. The SNVS_HP section is powered from the main SoC power, so it must be initialized whenever the device is powered on. If the SNVS_LP section is powered from the main SoC power rather than from an uninterrupted power source, the SNVS_LP section must also be initialized at SoC POR.

64.6 SNVS register descriptions

This section contains detailed register descriptions for the SNVS registers. Each description includes a standard register diagram and register table. The register table provides detailed descriptions of the register bit and field functions, in bit order.

SNVS registers consist of two types:

- Privileged read/write accessible
- Non-privileged read/write accessible

Privileged read/write accessible registers can only be accessed for read/write by privileged software. Unauthorized write accesses are ignored, and unauthorized read accesses return zero. Non-privileged software can access privileged access registers when the non-privileged software access enable bit is set in the SNVS_HP Command Register.

- Non-Secure
- Trusted
- Secure

Non-privileged read/write accessible registers are read/write accessible by any software.

The LP register values are set only on LP POR and are unaffected by System (HP) POR. The HP registers are set only on System POR and are unaffected by LP POR.

The following table shows the SNVS main memory map.

NOTE

For more information on security-related bitfields, see the Security Reference Manual.

64.6.1 SNVS Memory map

SNVS base address: 4107_0000

Offset	Register	Width (In bits)	Access	Reset value
4h	SNVS_HP Command Register (HPCOMR)	32	RW	0000_0000h
8h	SNVS_HP Control Register (HPCR)	32	RW	0000_0000h
14h	SNVS_HP Status Register (HPSR)	32	RW	8000_0000h
24h	SNVS_HP Real Time Counter MSB Register (HPRTCMR)	32	RW	0000_0000h
28h	SNVS_HP Real Time Counter LSB Register (HPRTCLR)	32	RW	0000_0000h
2Ch	SNVS_HP Time Alarm MSB Register (HPTAMR)	32	RW	0000_0000h
30h	SNVS_HP Time Alarm LSB Register (HPTALR)	32	RW	0000_0000h
34h	SNVS_LP Lock Register (LPLR)	32	RW	0000_0000h
38h	SNVS_LP Control Register (LPCR)	32	RW	0000_0020h
4Ch	SNVS_LP Status Register (LPSR)	32	RW	0000_0008h
5Ch	SNVS_LP Secure Monotonic Counter MSB Register (LPSMCMR)	32	RW	0000_0000h
60h	SNVS_LP Secure Monotonic Counter LSB Register (LPSMCLR)	32	RW	0000_0000h
64h	SNVS_LP Power Glitch Detector Register (LPPGDR)	32	RW	0000_0000h
68h	SNVS_LP General Purpose Register 0 (legacy alias) (LPGPR0_legacy_alias)	32	RW	0000_0000h
90h - 9Ch	SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0_alias - LPGPR3_alias)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
100h - 11Ch	SNVS_LP General Purpose Registers 0 .. 7 (LPGPR0 - LPGPR7)	32	RW	0000_0000h
BF8h	SNVS_HP Version ID Register 1 (HPVIDR1)	32	RO	003E_0104h
BFCh	SNVS_HP Version ID Register 2 (HPVIDR2)	32	RO	0600_0000h

64.6.2 SNVS_HP Command Register (HPCOMR)

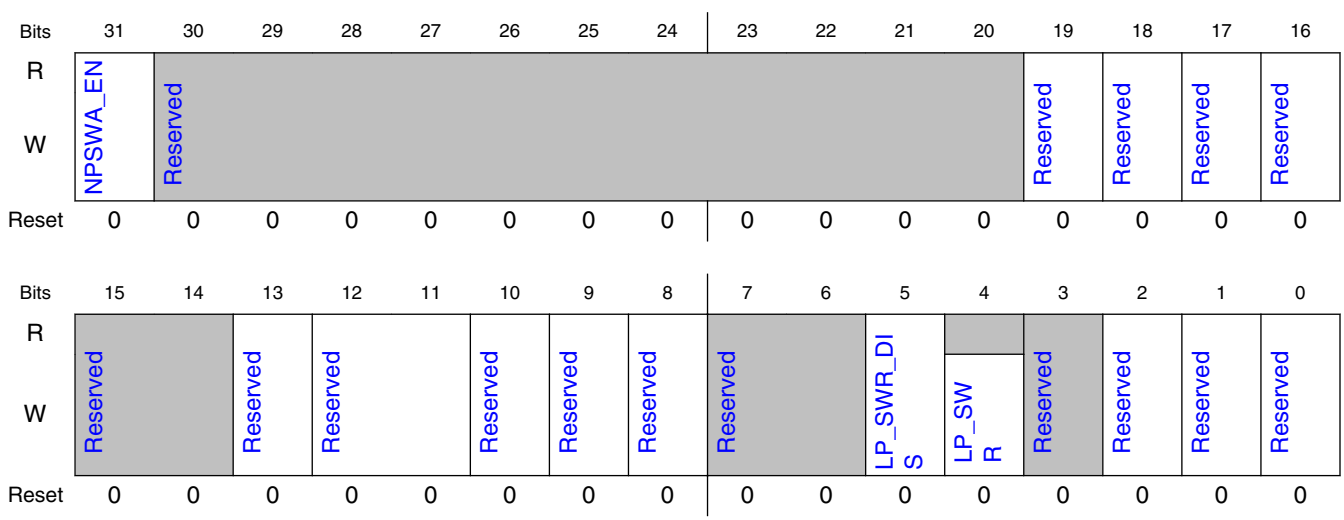
64.6.2.1 Offset

Register	Offset
HPCOMR	4h

64.6.2.2 Function

The SNVS_HP Command Register contains the command, configuration, and control bits for the SNVS block. This is a privileged write register.

64.6.2.3 Diagram



64.6.2.4 Fields

Field	Function
31 NPSWA_EN	Non-Privileged Software Access Enable When set, allows non-privileged software to access all SNVS registers, including those that are privileged software read/write access only. 0 Only privileged software can access privileged registers 1 Any software can access privileged registers
30-20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15-14 —	Reserved
13 —	Reserved
12-11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7-6 —	Reserved
5 LP_SWR_DIS	LP Software Reset Disable When set, disables the LP software reset. Once set, this bit can only be reset by the system reset. 0b - LP software reset is enabled 1b - LP software reset is disabled
4 LP_SWR	LP Software Reset When set to 1, the registers in the SNVS_LP section are reset. 0b - No Action 1b - Reset LP section
3	Reserved

Table continues on the next page...

Field	Function
—	
2 —	Reserved
1 —	Reserved
0 —	Reserved

64.6.3 SNVS_HP Control Register (HPCR)

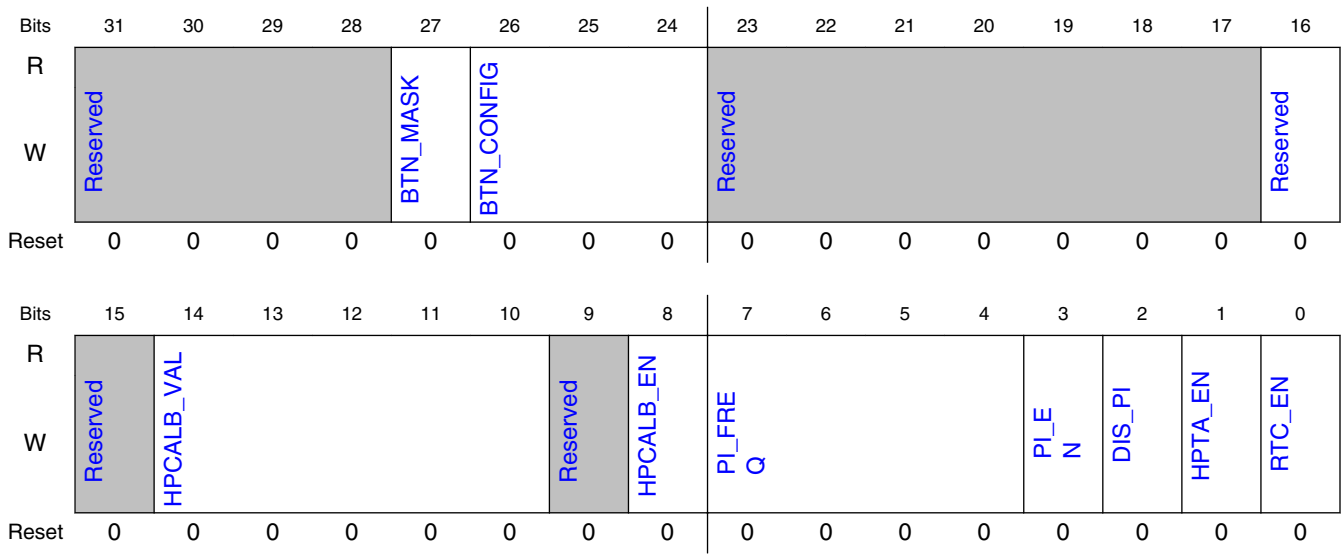
64.6.3.1 Offset

Register	Offset
HPCR	8h

64.6.3.2 Function

The SNVS_HP Control Register contains various control bits of the HP section of SNVS. This is *not* a privileged write register.

64.6.3.3 Diagram



64.6.3.4 Fields

Field	Function
31-28 —	Reserved
27 BTN_MASK	Button interrupt mask. This bit is used to mask the ipi_snvs_btn_int_b (button) interrupt request. 0: Interrupt disabled 1: Interrupt enabled
26-24 BTN_CONFIG	Button Configuration. This field is used to configure which feature of the button (BTN) input signal constitutes "active". 000: Button signal is active high 001: Button signal is active low 010: Button signal is active on the falling edge 011: Button signal is active on the rising edge 100: Button signal is active on any edge All other patterns are Reserved
23-17 —	Reserved
16 —	Reserved
15	Reserved

Table continues on the next page...

Field	Function
—	
14-10 HPCALB_VAL	HP Calibration Value Defines signed calibration value for the HP Real Time Counter. This field can be programmed only when RTC Calibration is disabled (HPCALB_EN is not set). This is a 5-bit 2's complement value, hence the allowable calibration values are in the range from -16 to +15 counts per 32768 ticks of the counter. 00000b - +0 counts per each 32768 ticks of the counter 00001b - +1 counts per each 32768 ticks of the counter 00010b - +2 counts per each 32768 ticks of the counter 01111b - +15 counts per each 32768 ticks of the counter 10000b - -16 counts per each 32768 ticks of the counter 10001b - -15 counts per each 32768 ticks of the counter 11110b - -2 counts per each 32768 ticks of the counter 11111b - -1 counts per each 32768 ticks of the counter
9 —	Reserved
8 HPCALB_EN	HP Real Time Counter Calibration Enabled Indicates that the time calibration mechanism is enabled. 0b - HP Timer calibration disabled 1b - HP Timer calibration enabled
7-4 PI_FREQ	Periodic Interrupt Frequency Defines frequency of the periodic interrupt. The interrupt is generated when a zero-to-one or one-to-zero transition occurs on the selected bit of the HP Real Time Counter and Real Time Counter and Periodic Interrupt are both enabled (RTC_EN and PI_EN are set). It is recommended to program this field when Periodic Interrupt is disabled (PI_EN is not set). The possible frequencies are: 0000b - - bit 0 of the HPRTCLR is selected as a source of the periodic interrupt 0001b - - bit 1 of the HPRTCLR is selected as a source of the periodic interrupt 0010b - - bit 2 of the HPRTCLR is selected as a source of the periodic interrupt 0011b - - bit 3 of the HPRTCLR is selected as a source of the periodic interrupt 0100b - - bit 4 of the HPRTCLR is selected as a source of the periodic interrupt 0101b - - bit 5 of the HPRTCLR is selected as a source of the periodic interrupt 0110b - - bit 6 of the HPRTCLR is selected as a source of the periodic interrupt 0111b - - bit 7 of the HPRTCLR is selected as a source of the periodic interrupt 1000b - - bit 8 of the HPRTCLR is selected as a source of the periodic interrupt 1001b - - bit 9 of the HPRTCLR is selected as a source of the periodic interrupt 1010b - - bit 10 of the HPRTCLR is selected as a source of the periodic interrupt 1011b - - bit 11 of the HPRTCLR is selected as a source of the periodic interrupt 1100b - - bit 12 of the HPRTCLR is selected as a source of the periodic interrupt 1101b - - bit 13 of the HPRTCLR is selected as a source of the periodic interrupt 1110b - - bit 14 of the HPRTCLR is selected as a source of the periodic interrupt 1111b - - bit 15 of the HPRTCLR is selected as a source of the periodic interrupt
3 PI_EN	HP Periodic Interrupt Enable The periodic interrupt can be generated only if the HP Real Time Counter is enabled. 0b - HP Periodic Interrupt is disabled 1b - HP Periodic Interrupt is enabled
2 DIS_PI	Disable periodic interrupt in the functional interrupt 0b - Periodic interrupt will trigger a functional interrupt 1b - Disable periodic interrupt in the function interrupt
1 HPTA_EN	HP Time Alarm Enable

Table continues on the next page...

Field	Function
	When set, the time alarm interrupt is generated if the value in the HP Time Alarm Registers is equal to the value of the HP Real Time Counter. 0b - HP Time Alarm Interrupt is disabled 1b - HP Time Alarm Interrupt is enabled
0 RTC_EN	HP Real Time Counter Enable. This bit syncs with the 32KHz clock. It won't update with the bus clock. 0b - RTC is disabled 1b - RTC is enabled

64.6.4 SNVS_HP Status Register (HPSR)

64.6.4.1 Offset

Register	Offset
HPSR	14h

64.6.4.2 Function

The HP Status Register reflects the internal state of the SNVS. This is *not* a privileged write register.

64.6.4.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved			Reserved	Reserved		Reserved								
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				Reserved				BI	BTN	Reserved	LPDIS	Reserved		PI	HPTA
W									W1C						W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0[1]	0	0	0	0

1. The value of Low Power Disable is determined by the *no_battery* input signal to SNVS.

64.6.4.4 Fields

Field	Function
31 —	Reserved
30-28 —	Reserved
27 —	Reserved
26-25 —	Reserved
24-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7 BI	Button Interrupt Signal ipi_snvs_btn_int_b was asserted.
6 BTN	Button Value of the BTN input. This is the external button used for PMIC control. 0: BTN not pressed 1: BTN pressed
5 —	Reserved
4 LPDIS	Low Power Disable If 1, the low power section has been disabled by means of an input signal to SNVS.
3-2 —	Reserved
1 PI	Periodic Interrupt Indicates that periodic interrupt has occurred since this bit was last cleared. 0b - No periodic interrupt occurred. 1b - A periodic interrupt occurred.
0 HPTA	HP Time Alarm Indicates that the HP Time Alarm has occurred since this bit was last cleared. 0b - No time alarm interrupt occurred. 1b - A time alarm interrupt occurred.

64.6.5 SNVS_HP Real Time Counter MSB Register (HPRTCMR)

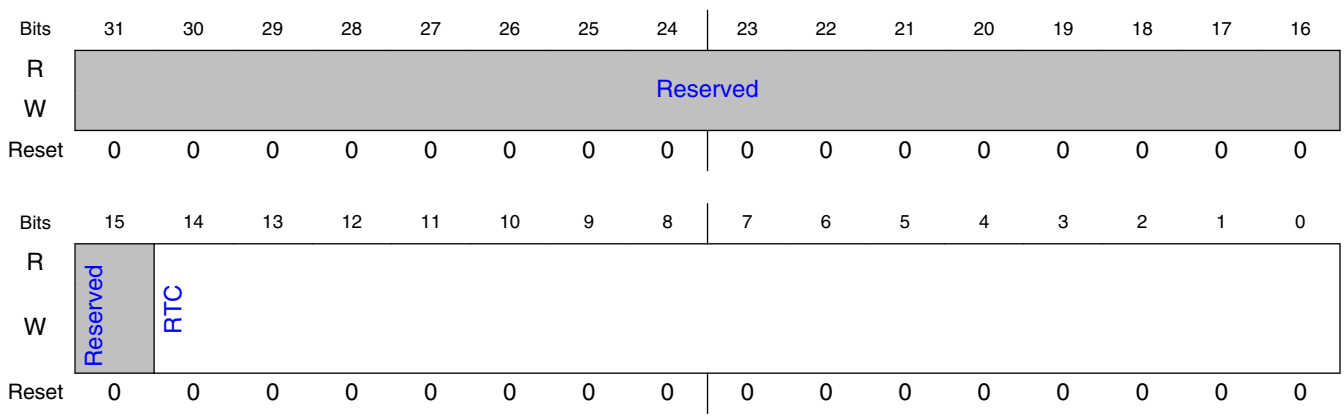
64.6.5.1 Offset

Register	Offset
HPRTCMR	24h

64.6.5.2 Function

The SNVS_HP Real Time Counter MSB register contains the 15 most-significant bits of the HP Real Time Counter. This is *not* a privileged write register.

64.6.5.3 Diagram



64.6.5.4 Fields

Field	Function
31-15 —	Reserved
14-0 RTC	HP Real Time Counter The most-significant 15 bits of the RTC. This register can be programmed only when RTC is not active (RTC_EN bit is not set).

64.6.6 SNVS_HP Real Time Counter LSB Register (HPRTCLR)

64.6.6.1 Offset

Register	Offset
HPRTCLR	28h

64.6.6.2 Function

The SNVS_HP Real Time Counter LSB register contains the 32 least-significant bits of the HP real time counter. This is *not* a privileged write register.

64.6.6.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RTC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RTC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.6.4 Fields

Field	Function
31-0 RTC	HP Real Time Counter least-significant 32 bits. This register can be programmed only when RTC is not active (RTC_EN bit is not set).

64.6.7 SNVS_HP Time Alarm MSB Register (HPTAMR)

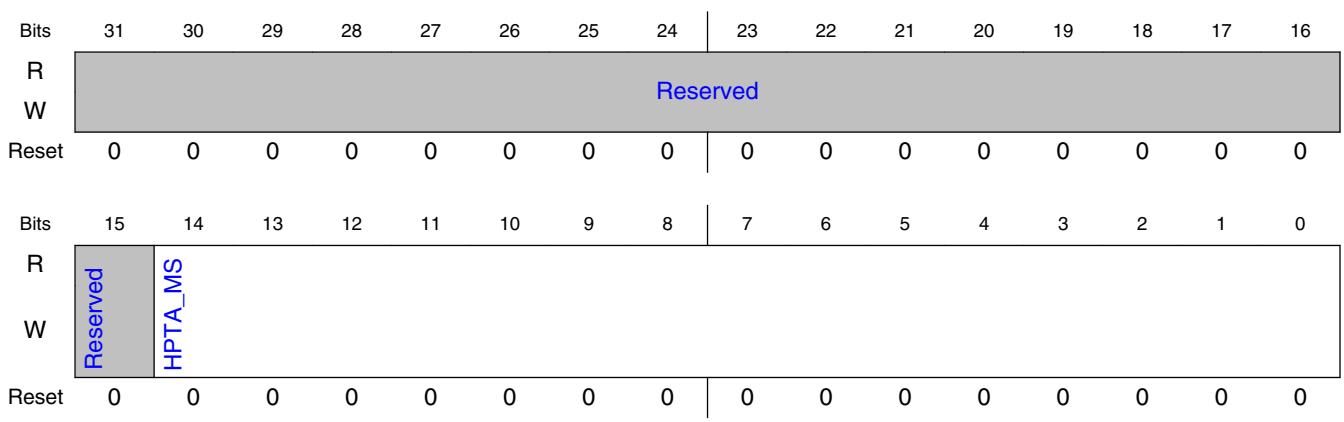
64.6.7.1 Offset

Register	Offset
HPTAMR	2Ch

64.6.7.2 Function

The SNVS_HP Time Alarm MSB register contains the most-significant bits of the SNVS_HP Time Alarm value. This is *not* a privileged write register.

64.6.7.3 Diagram



64.6.7.4 Fields

Field	Function
31-15 —	Reserved
14-0 HPTA_MS	HP Time Alarm, most-significant 15 bits. This register can be programmed only when HP time alarm is disabled (HPTA_EN bit is not set).

64.6.8 SNVS_HP Time Alarm LSB Register (HPTALR)

64.6.8.1 Offset

Register	Offset
HPTALR	30h

64.6.8.2 Function

The SNVS_HP Time Alarm LSB register contains the 32 least-significant bits of the SNVS_HP Time Alarm value. This is *not* a privileged write register.

64.6.8.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HPTA_LS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	HPTA_LS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.8.4 Fields

Field	Function
31-0 HPTA_LS	HP Time Alarm, 32 least-significant bits. This register can be programmed only when HP time alarm is disabled (HPTA_EN bit is not set).

64.6.9 SNVS_LP Lock Register (LPLR)

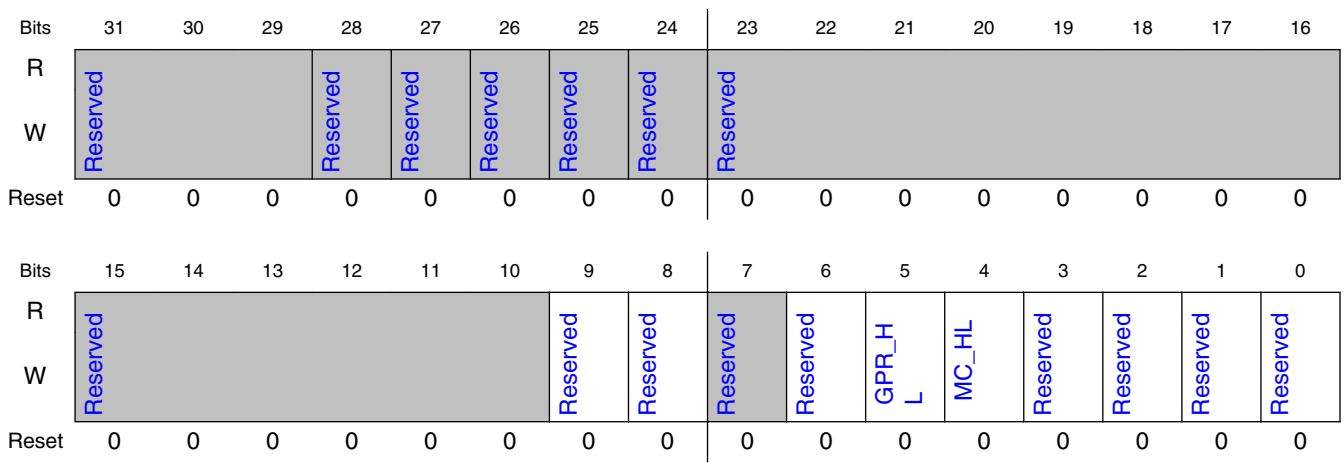
64.6.9.1 Offset

Register	Offset
LPLR	34h

64.6.9.2 Function

The SNVS_LP Lock Register contains lock bits for the SNVS_LP registers. This is a privileged write register.

64.6.9.3 Diagram



64.6.9.4 Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved

Table continues on the next page...

Field	Function
25 —	Reserved
24 —	Reserved
23-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 GPR_HL	General Purpose Register Hard Lock When set, prevents any writes to the GPR. Once set, this bit can only be reset by the LP POR. 0b - Write access is allowed. 1b - Write access is not allowed.
4 MC_HL	Monotonic Counter Hard Lock When set, prevents any writes (increments) to the MC Registers and MC_ENV bit. Once set, this bit can only be reset by the LP POR. 0b - Write access (increment) is allowed. 1b - Write access (increment) is not allowed.
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

64.6.10 SNVS_LP Control Register (LPCR)

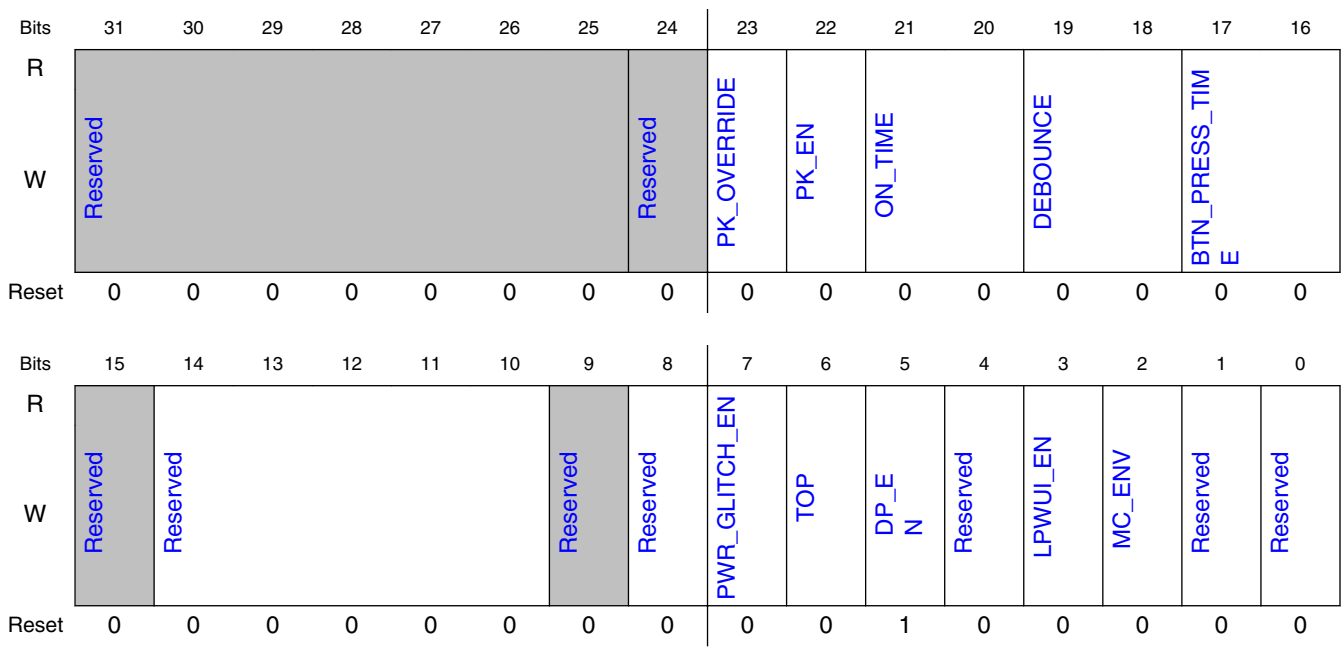
64.6.10.1 Offset

Register	Offset
LPCR	38h

64.6.10.2 Function

The SNVS_LP Control Register contains various control bits of the LP section of SNVS. This is a privileged write register.

64.6.10.3 Diagram



64.6.10.4 Fields

Field	Function
31-25 —	Reserved
24 —	Reserved

Table continues on the next page...

Field	Function
23 PK_OVERRIDE	PMIC On Request Override The value written to PK_OVERRIDE will be asserted on output signal snvs_lp_pk_override. That signal is used to override the IOMUX control for the PMIC I/O pad.
22 PK_EN	PMIC On Request Enable The value written to PK_EN will be asserted on output signal snvs_lp_pk_en. That signal is used to turn off the pullup/pulldown circuitry in the PMIC I/O pad.
21-20 ON_TIME	The ON_TIME field is used to configure the period of time after BTN is asserted before pmic_en_b is asserted to turn on the SoC power. 00: 500msec off->on transition time 01: 50msec off->on transition time 10: 100msec off->on transition time 11: 0msec off->on transition time
19-18 DEBOUNCE	This field configures the amount of debounce time for the BTN input signal. 00: 50msec debounce 01: 100msec debounce 10: 500msec debounce 11: 0msec debounce
17-16 BTN_PRESS_TIME	This field configures the button press time out values for the PMIC Logic. 00 : 5 secs 01 : 10 secs 10 : 15 secs 11 : long press disabled (pmic_en_b will not be asserted regardlessof how long BTN is asserted)
15 —	Reserved
14-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 PWR_GLITCH_EN	Power Glitch Enable By default the detection of a power glitch does not cause the pmic_en_b signal to be asserted. Setting the Power Glitch Enable bit to 1 enables the power glitch event for the PMIC. 0 - disabled 1 - enabled
6 TOP	Turn off System Power Asserting this bit causes a signal to be sent to the Power Management IC to turn off the system power. This bit will clear once power is off. This bit is only valid when the Dumb PMIC is enabled. 0b - Leave system power on. 1b - Turn off system power.
5	Dumb PMIC Enabled

Table continues on the next page...

Field	Function
DP_EN	When set, software can control the system power. When cleared, the system requires a Smart PMIC to automatically turn power off. 0b - Smart PMIC enabled. 1b - Dumb PMIC enabled.
4 —	Reserved
3 LPWUI_EN	LP Wake-Up Interrupt Enable This interrupt line should be connected to the external pin and is intended to inform the external chip about an SNVS_LP event (tamper event, MC rollover, SRTC rollover, or time alarm). This wake-up signal can be asserted only when the chip (HP section) is powered down, and the LP section is isolated. 0 LP wake-up interrupt is disabled. 1 LP wake-up interrupt is enabled.
2 MC_ENV	Monotonic Counter Enabled and Valid When set, the MC can be incremented (by write transaction to the LPSMCMR or LPSMCLR). Once MC_SL or MC_HL bit is set this bit can be changed only by LP software reset or LP POR. 0b - MC is disabled or invalid. 1b - MC is enabled and valid.
1 —	Reserved
0 —	Reserved

64.6.11 SNVS_LP Status Register (LPSR)

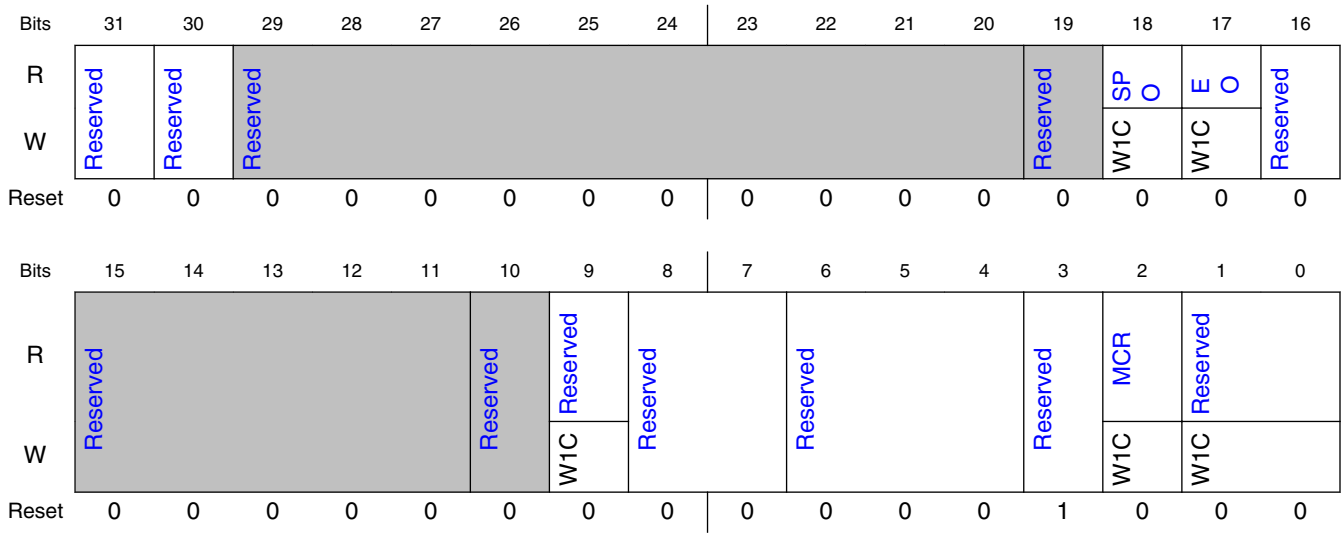
64.6.11.1 Offset

Register	Offset
LPSR	4Ch

64.6.11.2 Function

The SNVS_LP Status Register reflects the internal state and behavior of the SNVS_LP. This is a privileged write register.

64.6.11.3 Diagram



64.6.11.4 Fields

Field	Function
31 —	Reserved
30 —	Reserved
29-20 —	Reserved
19 —	Reserved
18 SPO	Set Power Off The SPO bit is set when the power button is pressed longer than the configured debounce time. Writing to the SPO bit will clear the set_pwr_off_irq interrupt. 0b - Set Power Off was not detected. 1b - Set Power Off was detected.
17 EO	Emergency Off This bit is set when a power off is requested. 0b - Emergency off was not detected. 1b - Emergency off was detected.
16 —	Reserved
15-11 —	Reserved

Table continues on the next page...

Field	Function
10 —	Reserved
9 —	Reserved
8-7 —	Reserved
6-4 —	Reserved
3 —	Reserved
2 MCR	Monotonic Counter Rollover 0b - MC has not reached its maximum value. 1b - MC has reached its maximum value.
1-0 —	Reserved

64.6.12 SNVS_LP Secure Monotonic Counter MSB Register (LPSMCMR)

64.6.12.1 Offset

Register	Offset
LPSMCMR	5Ch

64.6.12.2 Function

The SNVS_LP Secure Monotonic Counter MSB Register contains the monotonic counter era bits and the most-significant 16 bits of the monotonic counter. The monotonic counter is incremented by one if there is a write command to the LPSMCMR or LPSMCLR register. This is a non-privileged read-only register.

64.6.12.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MC_ERA_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MON_COUNTER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.12.4 Fields

Field	Function
31-16 MC_ERA_BITS	Monotonic Counter Era Bits These bits are inputs to the module and typically connect to fuses. When the Monotonic Counter is in use (i.e. enabled and valid and powered by an uninterrupted power source, e.g. a coin cell battery), and the boot software detects that the Monotonic Counter most-significant 16 Bits and Monotonic Counter LSB Register have been reset (MC_ENV=0), the boot software can take action to ensure that the value in the monotonic counter remains monotonic (i.e. never decreasing). The action is to blow an additional MC_ERA_BITS fuse. Since the MC_ERA_BITS field forms the most-significant field of the monotonic counter, blowing an additional fuse guarantees that the new monotonic counter value is higher than any previous value. Typically this would be necessary only when the coin cell battery is changed. Since the Monotonic Counter is reset on an LP Software Reset, an excessive number of MC_ERA_BITS fusez may be consumed if LP Software Reset is used repeatedly.
15-0 MON_COUNTER	Monotonic Counter most-significant 16 Bits **Note that writing to this register does *not* change the value of this field to the value that was written.** **The 48-bit monotonic counter value (consisting of LPSMCMR[MON_COUNTER] prepended to LPSMCLR[MON_COUNTER]) is incremented by one when:** • A write transaction to the LPSMCMR or LPSMCLR register is detected. • The MC_ENV bit is set. • MC_SL and MC_HL bits are not set. This value can be reset only by LP software reset or LP POR.

64.6.13 SNVS_LP Secure Monotonic Counter LSB Register (LPSMCLR)

64.6.13.1 Offset

Register	Offset
LPSMCLR	60h

64.6.13.2 Function

The SNVS_LP Secure Monotonic Counter LSB Register contains the 32 least-significant bits of the monotonic counter. The MC is incremented by one if there is a write command to the LPSMCMR or LPSMCLR register. This is a non-privileged read-only register.

64.6.13.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MON_COUNTER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MON_COUNTER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.13.4 Fields

Field	Function
31-0 MON_COUNTER	Monotonic Counter bits **Note that writing to this register does *not* change the value of this field to the value that was written.** **The 48-bit monotonic counter value (consisting of LPSMCMR[MON_COUNTER] prepended to LPSMCLR[MON_COUNTER]) is incremented by one when:** • A write transaction to the LPSMCMR or LPSMCLR register is detected. • The MC_ENV bit is set. • MC_SL and MC_HL bits are not set. This value can be reset only by LP software reset or LP POR.

64.6.14 SNVS_LP Power Glitch Detector Register (LPPGDR)

64.6.14.1 Offset

Register	Offset
LPPGDR	64h

64.6.14.2 Function

The SNVS_LP Power Glitch Detector Register is a 32-bit read/write register that is used for storing the power glitch detector value, as described in Power glitch detector (PGD). This is a privileged write register.

64.6.14.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PGD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PGD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.14.4 Fields

Field	Function
31-0 PGD	Power Glitch Detector Value

64.6.15 SNVS_LP General Purpose Register 0 (legacy alias) (LPGPR0_legacy_alias)

64.6.15.1 Offset

Register	Offset
LPGPR0_legacy_alias	68h

64.6.15.2 Function

See register SNVS_LP General Purpose Registers 0 .. 7 (LPGPR0 - LPGPR7).

64.6.15.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.15.4 Fields

Field	Function
31-0 GPR	General Purpose Register When GPR_SL or GPR_HL bit is set, the register cannot be programmed.

64.6.16 SNVS_LP General Purpose Registers 0 .. 3 (LPGPR0_alias - LPGPR3_alias)

64.6.16.1 Offset

Register	Offset
LPGPR0_alias	90h
LPGPR1_alias	94h
LPGPR2_alias	98h
LPGPR3_alias	9Ch

64.6.16.2 Function

See register SNVS_LP General Purpose Registers 0 .. 7 (LPGPR0 - LPGPR7).

64.6.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.16.4 Fields

Field	Function
31-0 GPR	General Purpose Register When GPR_SL or GPR_HL bit is set, the register cannot be programmed.

64.6.17 SNVS_LP General Purpose Registers 0 .. 7 (LPGPR0 - LPGPR7)

64.6.17.1 Offset

For a = 0 to 7:

Register	Offset
LPGPRa	100h + (a × 4h)

64.6.17.2 Function

The SNVS_LP General Purpose Register is a 256-bit read/write register located in SNVS_LP, which can be used by any application for retaining data during an SoC power-down mode. This is a privileged read/write register. The full GPR register is accessed as 8 32-bit registers located in successive word addresses starting at offset 100h. For backward compatibility with earlier versions of SNVS, LPGPR0..LPGPR3 are aliased at the earlier offset of 90h and LPGPR0 is also aliased at its original offset of 68h. New software should access the GPR register at the preferred offset of 100h. The GPR will be automatically zeroized when an enabled security event occurs, unless GPR zeroization is disabled via the GPR_Z_DIS bit in the LP Control Register.

64.6.17.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.17.4 Fields

Field	Function
31-0 GPR	General Purpose Register When GPR_SL or GPR_HL bit is set, the register cannot be programmed.

64.6.18 SNVS_HP Version ID Register 1 (HPVIDR1)

64.6.18.1 Offset

Register	Offset
HPVIDR1	BF8h

64.6.18.2 Function

The SNVS_HP Version ID Register 1 is a non-privileged read-only register that contains the current version of the SNVS. The version consists of a module ID, a major version number, and a minor version number.

64.6.18.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IP_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR_REV								MINOR_REV							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0

64.6.18.4 Fields

Field	Function
31-16 IP_ID	SNVS block ID
15-8 MAJOR_REV	SNVS block major version number
7-0 MINOR_REV	SNVS block minor version number

64.6.19 SNVS_HP Version ID Register 2 (HPVIDR2)

64.6.19.1 Offset

Register	Offset
HPVIDR2	BFCh

64.6.19.2 Function

The SNVS_HP Version ID Register 2 is a non-privileged read-only register that indicates the current version of the SNVS. Version ID register 2 consists of the following fields: integration options, ECO revision, and configuration options.

64.6.19.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IP_ERA								INTG_OPT							
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ECO_REV								CONFIG_OPT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

64.6.19.4 Fields

Field	Function
31-24	IP Era
IP_ERA	00h - Era 1 or 2
	03h - Era 3
	04h - Era 4

Table continues on the next page...

Field	Function
	05h - Era 5
23-16 INTG_OPT	SNVS Integration Options
15-8 ECO_REV	SNVS ECO Revision
7-0 CONFIG_OPT	SNVS Configuration Options

Chapter 65
12-bit Analog to Digital Converter (ADC)

65.1 Chip-specific ADC information

Table 65-1. Reference links to related information

Topic	Related module	Reference
Full description	ADC	ADC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

65.1.1 Analog to Digital Converter (ADC)

The ADC is a 12-bit resolution, successive approximation ADC designed for operation within an integrated microcontroller. The ADC can support up to 16 single-ended external analog inputs. The ADC supports only 12-bit non-differential and 13-bit differential formatting option. The ADC can achieve 1 microsecond conversion rate. The following table shows the configuration of the ADC.

Table 65-2. ADC configuration

Parameter	Description
Name	Analog to Digital Converter (ADC)
Instances	2
Configurable features	• FIFO Size: 16 • Command Buffers: 15 • Compare Values: 4 • Trigger Sources: 2
Interface speed	NA
External I/O pins	ADCx_CHnA, ADCx_CHnB (n = [7:0]). See the attached IOMUX spreadsheet for input sources for all the channels in ADC0 and ADC1.

65.1.2 ADC triggers

On i.MX 7ULP, ADC Trigger sources get routed through Trigger MUX. See the Table 38-2 for ADC0 and ADC1 trigger sources.

This device does not support ADC to ADC, and DAC to ADC type chained conversions but the regular time based conversions should partly cover the ADC Synchronous Operation. The solution consists of a timer based trigger, which can trigger the conversion at regular intervals. The hardware trigger will be associated to a command, so it will switch between ADC 0 and 1 for sampling at every trigger. There are also software triggers in each ADC. So, if multiple ADCs begin sampling with some fixed timing relationship, the user can either connect their hardware trigger sources together, or trigger them in software and program some startup delay on each trigger.

65.1.3 ADC channel input options

To know about the input source options of all the implemented channels (CH0-CH15) of ADC0 and ADC1 and temperature sensor option, see the " ADC Sources" tab in the attached IOMUXC Pinout spreadsheet.

65.1.4 ADC voltage reference sources

The table below shows voltage reference options that can be selected through CFG[REFSEL] field in ADC0 and ADC1.

Table 65-3. ADC0/1 voltage reference sources

ADC instance	Pin name	REFSEL bit value	Voltage reference source
ADC0	vrefh1	REFSEL = 00b	Voltage reference VREFH_ANA18
	vrefh2	REFSEL = 01b	1.8V supply VDD_ANA18
	vrefh3	REFSEL = 10b	1.8V supply VDD_ANA18
	vrefl_1p8		Voltage reference VREFL_ANA
ADC1	vrefh1	REFSEL = 00b	Voltage reference VREFH_ANA18
	vrefh2	REFSEL = 01b	1.8V supply VDD_ANA18
	vrefh3	REFSEL = 10b	1.8V supply VDD_ANA18

Table continues on the next page...

Table 65-3. ADC0/1 voltage reference sources (continued)

ADC instance	Pin name	REFSEL bit value	Voltage reference source
	vrefl_1p8		Voltage reference VREFL_ANA

65.2 Introduction

The 12-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

NOTE

For the chip specific modes of operation, see the power management information of the device.

Note

Hexadecimal values are designated by a preceding 0x, binary values designated by a preceding 0b, and decimal values have no preceding character.

65.2.1 Features

Following are the features of the ADC module.

- Linear successive approximation algorithm
  - differential operation with 13-bit resolution
  - single-ended operation with 12-bit resolution
- Channel support for up to 64 analog input channels for conversion of external pins and from internal sources
  - Select external pin inputs paired for conversion as differential channel input
  - Measurement of on-chip analog sources such as DAC, temperature sensor or bandgap
- Channel scaling allows input voltage levels higher than the ADC reference voltage
- Configurable analog input sample time
- Configurable speed options to accommodate operation in low power modes of the device.
- Trigger detect with up to 2 trigger sources with priority level configuration. Software or hardware trigger option for each.
- 15 command buffers allow independent options selection and channel sequence scanning.

- Automatic compare for less-than, greater-than, within range, or out-of-range with "store on true" and "repeat until true" options
- 16-entry conversion result data FIFO with configurable watermark and overflow detection
- Interrupt, DMA or polled operation

65.2.2 Block diagram

The following figure is the ADC module block diagram.

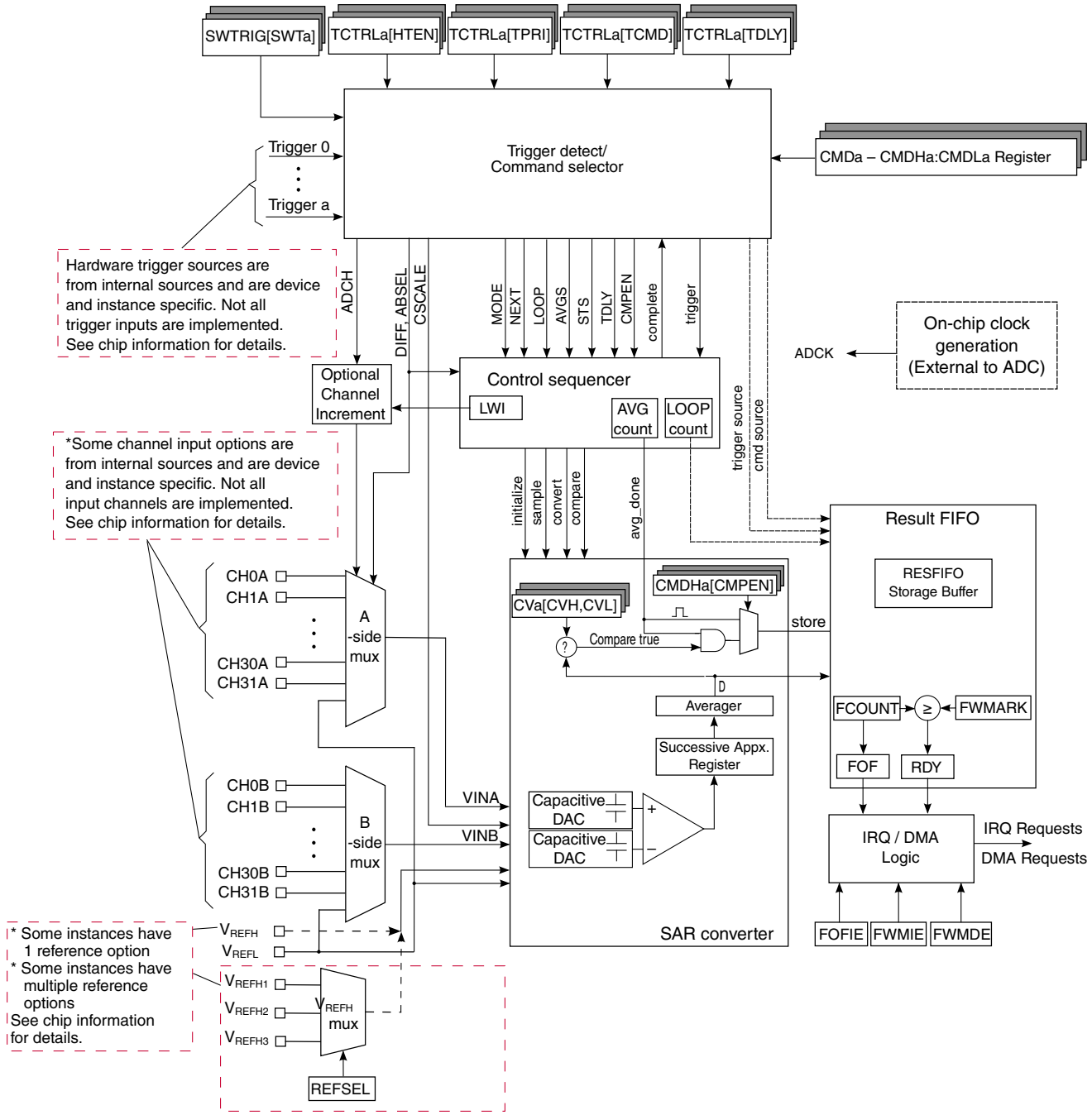


Figure 65-1. ADC block diagram

65.2.3 Modes of operation

The table below shows the operation of ADC module in various chip modes.

Table 65-4. Chip modes supported by the ADC module

Chip mode	ADC Operation
Run	Normal operation
Stop/Wait	Can continue operating provided the Doze Enable bit (CTRL[DOZEN]) is clear and the module is using an external or internal clock source which remains operating during stop/wait modes. When the CTRL[DOZEN] bit is set, the module will wait for the current averaging iteration/FIFO storage to complete before acknowledging stop or wait mode entry.
Low Leakage Stop	The Doze Enable (CTRL[DOZEN]) bit is ignored and the ADC will wait for the current transfer to complete any pending operation before acknowledging low-leakage mode entry.

65.3 Signal descriptions

The ADC module supports up to 64 analog channel inputs with differential and single-ended conversion options for all channels. The module also requires supply and ground connections.

Table 65-5. ADC signal descriptions

Signal	Description	I/O
V_{DDA}	Analog Power Supply	I
V_{SSA}	Analog Ground	I
CH31A–CH0A	A-side Analog Channel Inputs	I
CH31B–CH0B	B-side Analog Channel Inputs	I

NOTE

The voltage reference high (V_{REFH}) used by the ADC is supplied from either an on-chip voltage reference source or from an off-chip source supplied through external pins. V_{REFL} is always from an external pin and must be at the same voltage potential as V_{SSA}. See the chip configuration information on the voltage reference options specific to this packaged device.

NOTE

This block supports a programmable selection of the Voltage Reference High used for ADC conversions (via the CFG[REFSEL] field). See the chip configuration information on the voltage reference options specific to this packaged device.

65.3.1 Analog Power (V_{DDA})

The ADC analog portion uses V_{DDA} as its power connection. In some packages, V_{DDA} is connected internally to V_{DD}. If externally available, connect the V_{DDA} pin to the same voltage potential as V_{DD}. External filtering may be necessary to ensure clean V_{DDA} for good results.

65.3.2 Analog Ground (V_{SSA})

The ADC analog portion uses V_{SSA} as its ground connection. In some packages, V_{SSA} is connected internally to V_{SS}. If externally available, connect the V_{SSA} pin to the same voltage potential as V_{SS}.

65.3.3 Analog Channel Inputs (CH*n*A and CH*n*B)

The CMDL*a*[ADCH], CMDL*a*[DIFF] and CMDL*a*[ABSEL] bitfields control selection of paired or individual input channels. Each ADC command independently makes a channel selection. Each ADCH channel selection has an associated A side and an associated B side input. Each ADCH pair can optionally be converted in a differential mode but only limited pairs are intended to be converted as differential channels (i.e., adjacent pins that have been designed with matched impedance). For the pin pairings available for differential conversions for your device, see the Chip Configuration details.

NOTE

Some of the input channels are from on-chip sources such as temperature sensors and reference voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions may not be available for your device. Refer to ADC configuration section in the chip configuration chapter for the channels supported on this device.

65.4 ADC register descriptions

This section describes the ADC registers.

65.4.1 ADC Memory map

ADC0 base address: 4104_1000h

ADC1 base address: 410A_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_001Ah
4h	Parameter Register (PARAM)	32	RO	0F04_1002h
10h	ADC Control Register (CTRL)	32	RW	0000_0000h
14h	ADC Status Register (STAT)	32	W1C	0000_0000h
18h	Interrupt Enable Register (IE)	32	RW	0000_0000h
1Ch	DMA Enable Register (DE)	32	RW	0000_0000h
20h	ADC Configuration Register (CFG)	32	RW	0080_0000h
24h	ADC Pause Register (PAUSE)	32	RW	0000_0000h
30h	ADC FIFO Control Register (FCTRL)	32	RW	0000_0000h
34h	Software Trigger Register (SWTRIG)	32	WORZ	0000_0000h
C0h - C4h	Trigger Control Register (TCTRL0 - TCTRL1)	32	RW	0000_0000h
100h	ADC Command Low Buffer Register (CMDL1)	32	RW	0000_2000h
104h	ADC Command High Buffer Register (CMDH1)	32	RW	0000_0000h
108h	ADC Command Low Buffer Register (CMDL2)	32	RW	0000_2000h
10Ch	ADC Command High Buffer Register (CMDH2)	32	RW	0000_0000h
110h	ADC Command Low Buffer Register (CMDL3)	32	RW	0000_2000h
114h	ADC Command High Buffer Register (CMDH3)	32	RW	0000_0000h
118h	ADC Command Low Buffer Register (CMDL4)	32	RW	0000_2000h
11Ch	ADC Command High Buffer Register (CMDH4)	32	RW	0000_0000h
120h	ADC Command Low Buffer Register (CMDL5)	32	RW	0000_2000h
124h	ADC Command High Buffer Register (CMDH5)	32	RW	0000_0000h
128h	ADC Command Low Buffer Register (CMDL6)	32	RW	0000_2000h
12Ch	ADC Command High Buffer Register (CMDH6)	32	RW	0000_0000h
130h	ADC Command Low Buffer Register (CMDL7)	32	RW	0000_2000h
134h	ADC Command High Buffer Register (CMDH7)	32	RW	0000_0000h
138h	ADC Command Low Buffer Register (CMDL8)	32	RW	0000_2000h
13Ch	ADC Command High Buffer Register (CMDH8)	32	RW	0000_0000h
140h	ADC Command Low Buffer Register (CMDL9)	32	RW	0000_2000h
144h	ADC Command High Buffer Register (CMDH9)	32	RW	0000_0000h
148h	ADC Command Low Buffer Register (CMDL10)	32	RW	0000_2000h
14Ch	ADC Command High Buffer Register (CMDH10)	32	RW	0000_0000h
150h	ADC Command Low Buffer Register (CMDL11)	32	RW	0000_2000h
154h	ADC Command High Buffer Register (CMDH11)	32	RW	0000_0000h

Table continues on the next page...

Offset	Register	Width (In bits)	Access	Reset value
158h	ADC Command Low Buffer Register (CMDL12)	32	RW	0000_2000h
15Ch	ADC Command High Buffer Register (CMDH12)	32	RW	0000_0000h
160h	ADC Command Low Buffer Register (CMDL13)	32	RW	0000_2000h
164h	ADC Command High Buffer Register (CMDH13)	32	RW	0000_0000h
168h	ADC Command Low Buffer Register (CMDL14)	32	RW	0000_2000h
16Ch	ADC Command High Buffer Register (CMDH14)	32	RW	0000_0000h
170h	ADC Command Low Buffer Register (CMDL15)	32	RW	0000_2000h
174h	ADC Command High Buffer Register (CMDH15)	32	RW	0000_0000h
200h - 20Ch	Compare Value Register (CV1 - CV4)	32	RW	0000_0000h
300h	ADC Data Result FIFO Register (RESFIFO)	32	RO	0000_0000h

65.4.2 Version ID Register (VERID)

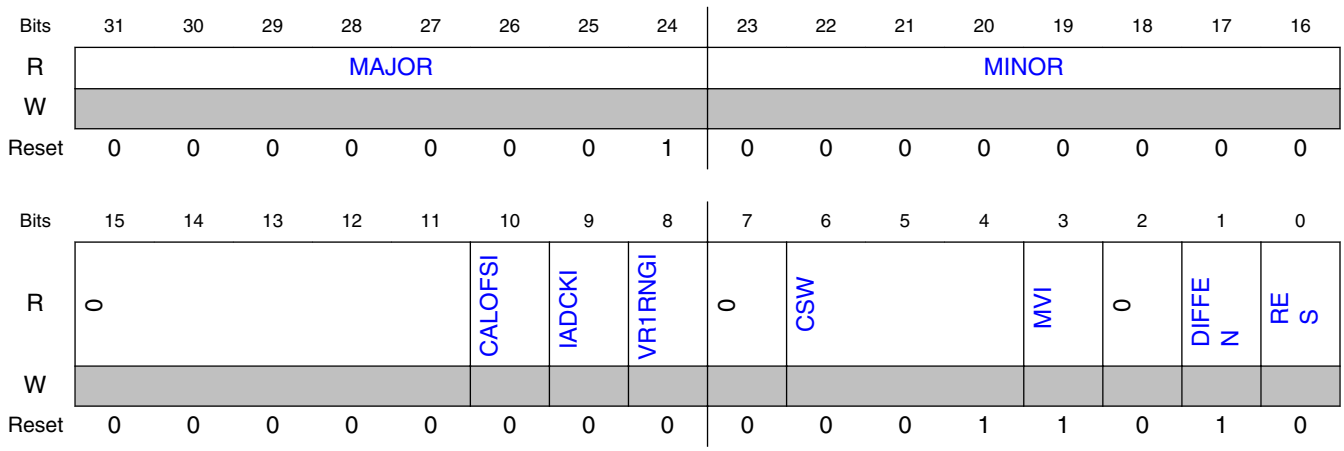
65.4.2.1 Offset

Register	Offset
VERID	0h

65.4.2.2 Function

The Version ID register indicates the version integrated for this instance on the device and also indicates inclusion/exclusion of several optional features.

65.4.2.3 Diagram



65.4.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number This read-only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number This read-only field returns the minor version number for the module specification.
15-11 —	Reserved This read-only field is reserved and always has the value 0.
10 CALOFSI	Calibration Offset Function Implemented This read-only field indicates if a calibration function and trimming control register are implemented. When supported, the CFG[CALOFS] and OFSTRIM[OFSTRIM] fields are available in support of this function. 0b - Offset calibration and offset trimming not implemented. 1b - Offset calibration and offset trimming implemented.
9 IADCKI	Internal ADC Clock implemented This read-only field indicates if this implementation of the ADC includes an internal clock source. When supported, the CFG[ADCKEN] field is available and documented for enabling the clock source. When available, this clock source is used in clock selection logic external to the module for use within the system. 0b - Internal clock source not implemented. 1b - Internal clock source (and CFG[ADCKEN]) implemented.
8 VR1RNGI	Voltage Reference 1 Range Control Bit Implemented This read-only field indicates if a control bit is implemented for selecting the input voltage range on Voltage Reference Option 1.When range control is required, the CFG[VREF1RNG] field is available and documented for controlling the Voltage Reference Option 1. 0b - Range control not required. CFG[VREF1RNG] is not implemented. 1b - Range control required. CFG[VREF1RNG] is implemented.
7	Reserved

Table continues on the next page...

Field	Function
—	This read-only field is reserved and always has the value 0.
6-4 CSW	Channel Scale Width This read-only field indicates if channel scaling is supported by this implementation. When supported, each command buffer has a control field (CMDLa[CSCALE]) for setting input scaling. 000b - Channel scaling not supported. 001b - Channel scaling supported. 1-bit CSCALE control field. 110b - Channel scaling supported. 6-bit CSCALE control field.
3 MVI	Multi Vref Implemented This read-only field indicates if multiple Voltage Reference High inputs are supported by this implementation. When multiple voltage references are supported, the CFG[REFSEL] field is available and documented for selecting the Voltage Reference High options. 0b - Single voltage reference high (VREFH) input supported. 1b - Multiple voltage reference high (VREFH) inputs supported.
2 —	Reserved This read-only field is reserved and always has the value 0.
1 DIFFEN	Differential Supported This read-only field indicates if differential operation is supported by this implementation. When supported, each command buffer has control fields (CMDLa[DIFF] and CMDLa[ABSEL]) for configuring for differential operation and expanding the number of supported channels from 32 to 64. 0b - Differential operation not supported. 1b - Differential operation supported. CMDLa[DIFF] and CMDLa[ABSEL] control fields implemented.
0 RES	Resolution This read-only field indicates the maximum accuracy supported by this implementation. When RES=1, each command buffer has a control field (CMDLa[MODE]) for selecting resolution of conversions for the associated command. 0b - Up to 13-bit differential/12-bit single ended resolution supported. 1b - Up to 16-bit differential/15-bit single ended resolution supported.

65.4.3 Parameter Register (PARAM)

65.4.3.1 Offset

Register	Offset
PARAM	4h

65.4.3.2 Function

The Parameter register indicates the size of several variable integration options for this instance on the device.

65.4.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMD_NUM								CV_NUM							
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIFOSIZE								TRIG_NUM							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0

65.4.3.4 Fields

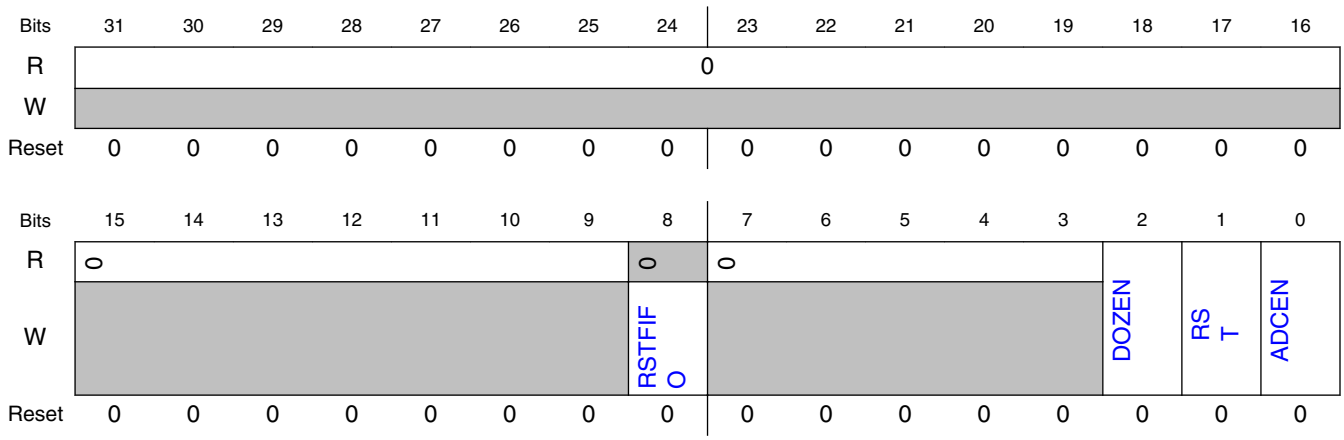
Field	Function
31-24 CMD_NUM	Command Buffer Number Number of command buffers implemented.
23-16 CV_NUM	Compare Value Number Number of compare value registers implemented.
15-8 FIFOSIZE	Result FIFO Depth The maximum number of conversion datawords that can be stored in the result FIFO before an overflow occurs. This field is read-only. 00000001b - Result FIFO depth = 1 dataword. 00000100b - Result FIFO depth = 4 datawords. 00001000b - Result FIFO depth = 8 datawords. 00010000b - Result FIFO depth = 16 datawords. 00100000b - Result FIFO depth = 32 datawords. 01000000b - Result FIFO depth = 64 datawords.
7-0 TRIG_NUM	Trigger Number Number of Triggers implemented.

65.4.4 ADC Control Register (CTRL)

65.4.4.1 Offset

Register	Offset
CTRL	10h

65.4.4.2 Diagram



65.4.4.3 Fields

Field	Function
31-9 —	Reserved This read-only field is reserved and always has the value 0.
8 RSTFIFO	Reset FIFO 0b - No effect. 1b - FIFO is reset.
7-3 —	Reserved This read-only field is reserved and always has the value 0.
2 DOZEN	Doze Enable Controls system transition to Stop and Wait power modes while ADC is converting. When DOZEN is clear, immediate entries to Wait or Stop are allowed. Note that the selected clock source provided from the on-chip clock source must be able to continue operating. When the DOZEN bit is set, the ADC waits for the current averaging iteration/FIFO storage to complete before acknowledging stop or wait mode entry. When the system is entering a Low Leakage Stop mode, the DOZEN bit is ignored and the module waits for the current transfer to complete any pending operation before acknowledging low leakage mode entry. 0b - ADC is enabled in Doze mode. 1b - ADC is disabled in Doze mode.
1 RST	Software Reset Resets all internal logic and registers, except the Control Register. Remains set until cleared by software. 0b - ADC logic is not reset. 1b - ADC logic is reset.
0 ADCEN	ADC Enable 0b - ADC is disabled. 1b - ADC is enabled.

65.4.5 ADC Status Register (STAT)

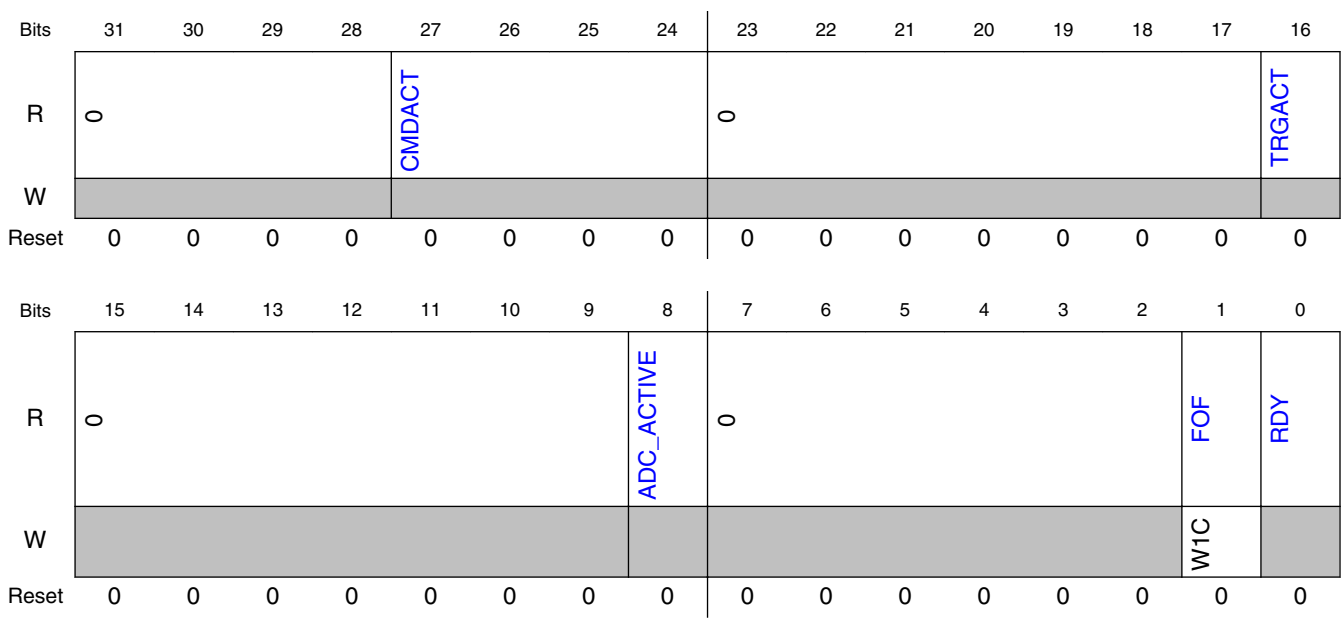
65.4.5.1 Offset

Register	Offset
STAT	14h

65.4.5.2 Function

The Status Register provides the current status of the ADC module.

65.4.5.3 Diagram



65.4.5.4 Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.

Table continues on the next page...

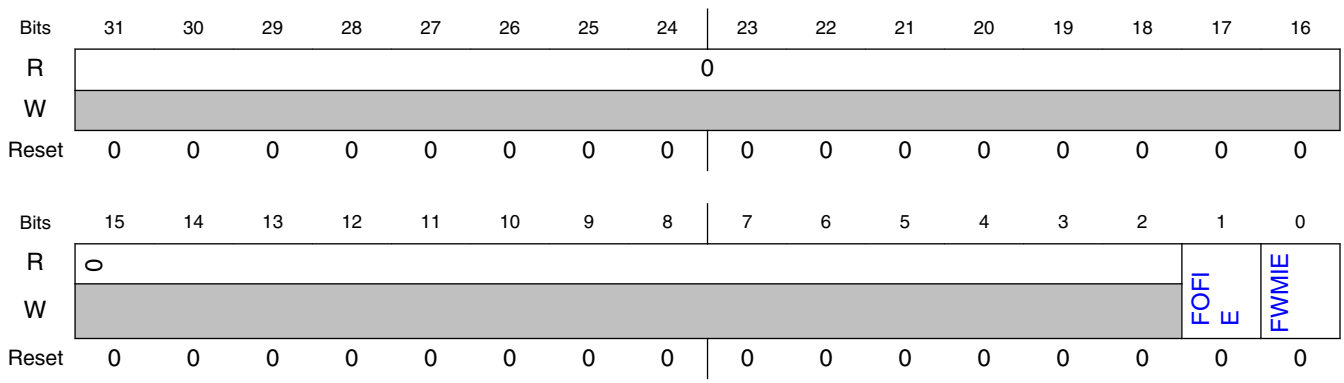
Field	Function
27-24 CMDACT	Command Active CMDACT is a read-only status field indicating the command that is actively being processed. 0000b - No command is currently in progress. 0001b - Command 1 currently being executed. 0010b - Command 2 currently being executed. 0011-1111b - Associated command number is currently being executed.
23-17 —	Reserved This read-only field is reserved and always has the value 0.
16 TRGACT	Trigger Active TRGACT is a read-only status field indicating the trigger associated with the command actively being processed. TRGACT only has meaning when CMDACT is non-zero. 0b - Command (sequence) associated with Trigger 0 currently being executed. 1b - Command (sequence) associated with Trigger 1 currently being executed.
15-9 —	Reserved This read-only field is reserved and always has the value 0.
8 ADC_ACTIVE	ADC Active This flag shows if the module is currently processing a conversion, or has pending triggers to service. When the ADC is IDLE, this bit will be set to 0b0. 0b - The ADC is IDLE. There are no pending triggers to service and no active commands are being processed. 1b - The ADC is processing a conversion, running through the power up delay, or servicing a trigger.
7-2 —	Reserved This read-only field is reserved and always has the value 0.
1 FOF	Result FIFO Overflow Flag Indicates that more data has been written to the Result FIFO than it can hold. The newer data is not stored and the FIFO remains holding the original contents. This field asserts regardless of the value of IE[FOFIE]. However, an interrupt request is issued only if IE[FOFIE] is set. This flag is cleared by writing a 1. 0b - No result FIFO overflow has occurred since the last time the flag was cleared. 1b - At least one result FIFO overflow has occurred since the last time the flag was cleared.
0 RDY	Result FIFO Ready Flag Indicates when the number of valid datawords in the result FIFO is greater than the watermark level set in the FCTRL[FWMARK] bitfield. This field asserts regardless of the value of FWMIE. However, an interrupt request or DMA request occurs only when the associated control bit (IE[FWMIE] and DE[FWMDE]) is set. This flag is cleared when the FCOUNT (which decrements on each read of the RESFIFO register) is less than or equal to the watermark level set in the FCTRL[FWMARK] bitfield. 0b - Result FIFO data level not above watermark level. 1b - Result FIFO holding data above watermark level.

65.4.6 Interrupt Enable Register (IE)

65.4.6.1 Offset

Register	Offset
IE	18h

65.4.6.2 Diagram



65.4.6.3 Fields

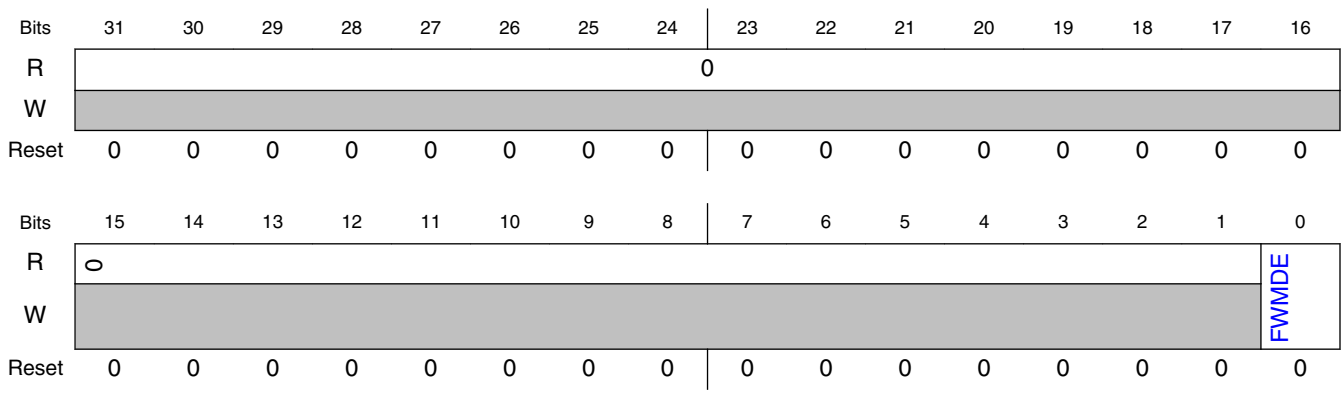
Field	Function
31-2 —	Reserved This read-only field is reserved and always has the value 0.
1 FOFIE	Result FIFO Overflow Interrupt Enable Configures ADC to generate overflow interrupt requests when FOF flag is asserted. 0b - FIFO overflow interrupts are not enabled. 1b - FIFO overflow interrupts are enabled.
0 FWMIE	FIFO Watermark Interrupt Enable Configures the module to generate watermark interrupt requests when RDY flag is asserted. 0b - FIFO watermark interrupts are not enabled. 1b - FIFO watermark interrupts are enabled.

65.4.7 DMA Enable Register (DE)

65.4.7.1 Offset

Register	Offset
DE	1Ch

65.4.7.2 Diagram



65.4.7.3 Fields

Field	Function
31-1 —	Reserved This read-only field is reserved and always has the value 0.
0 FWMDE	FIFO Watermark DMA Enable Configures ADC to generate DMA requests when RDY flag is asserted. 0b - DMA request disabled. 1b - DMA request enabled.

65.4.8 ADC Configuration Register (CFG)

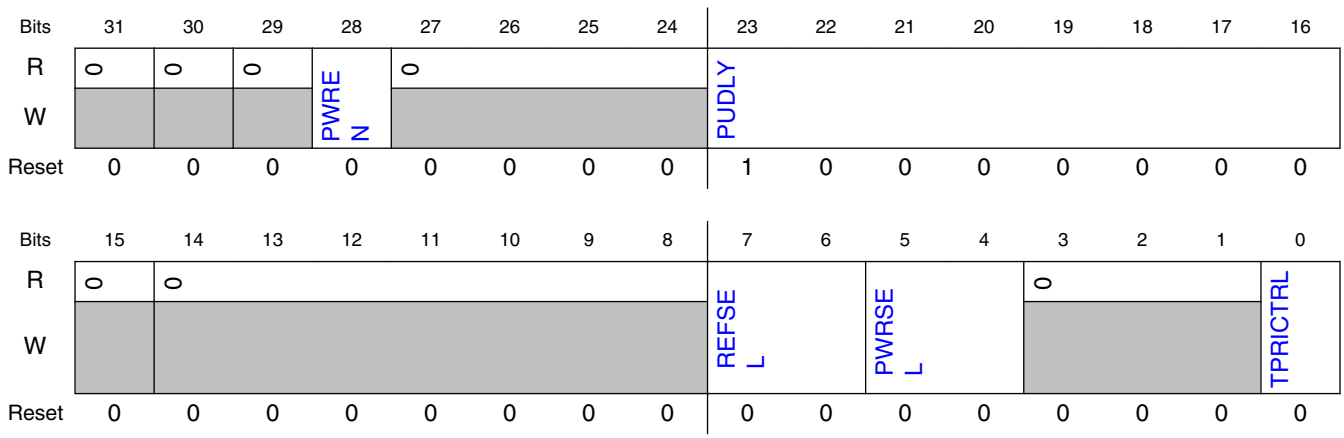
65.4.8.1 Offset

Register	Offset
CFG	20h

65.4.8.2 Function

The Configuration Register controls ADC functions that are common to all commands. The CFG cannot be changed while the CTRL[ADCEN] bit is set. Writes to CFG while CTRL[ADCEN] is set are ignored.

65.4.8.3 Diagram



65.4.8.4 Fields

Field	Function
31	Reserved
—	This read-only field is reserved and always has the value 0.
30	Reserved
—	This read-only field is reserved and always has the value 0.
29	Reserved
—	This read-only field is reserved and always has the value 0.
28 PWREN	ADC Analog Pre-Enable Enables the ADC analog circuits. When setting CFG[PWREN], user code should delay for a period exceeding the analog startup time of $t_{ADCSTUP}$ before enabling the ADC for operation. The module is still operational even when CFG[PWREN] is left clear but command execution start is delayed for a period defined by CFG[PUDLY]. Refer to the device datasheet for ADC idle and ADC active power consumption parameters. 0b - ADC analog circuits are only enabled while conversions are active. Performance is affected due to analog startup delays. 1b - ADC analog circuits are pre-enabled and ready to execute conversions without startup delays (at the cost of higher DC current consumption). When PWREN is set, the power up delay is enforced such that any detected trigger does not begin ADC operation until the power up delay time has passed.

Table continues on the next page...

Field	Function
27-24 —	Reserved This read-only field is reserved and always has the value 0.
23-16 PUDLY	Power Up Delay When CFG[PWREN]=0b0, the ADC analog circuits are only powered while the module is active and there is a counted delay defined by CFG[PUDLY] after an initial trigger transitions the ADC from its Idle state to allow time for the analog circuits to stabilize. The startup delay count of (PUDLY*4) ADCK cycles must result in a longer delay than the analog startup time of $t_{ADCSTUP}$. Accuracy of the initial conversion after activation is degraded if CFG[PUDLY] is set to too small a value. When CFG[PWREN]=0b1 prior to ADC activation, the analog circuits are pre-enabled and the activation delay defined by CFG[PUDLY] is bypassed.
15 —	Reserved This read-only field is reserved and always has the value 0.
14-8 —	Reserved This read-only field is reserved and always has the value 0.
7-6 REFSEL	Voltage Reference Selection Selects the voltage reference high used for conversions. **NOTE:** See the chip configuration information on the voltage reference options specific to this packaged device. 00b - (Default) Option 1 setting. 01b - Option 2 setting. 10b - Option 3 setting. 11b - Reserved
5-4 PWRSEL	Power Configuration Select Configures the module for power and performance. In the highest power setting, the highest conversion rates are possible. Refer to the device datasheet for power and performance capabilities for each setting. The highest power setting corresponds to 0b11 with decreasing power down to 0b00. 00b - Level 1 (Lowest power setting) 01b - Level 2 10b - Level 3 11b - Level 4 (Highest power setting)
3-1 —	Reserved This read-only field is reserved and always has the value 0.
0 TPRICTRL	ADC trigger priority control This bitfield controls how higher priority triggers are handled. See Trigger detect and command execution for a detailed explanation trigger event handling. 0b - If a higher priority trigger is detected during command processing, the current conversion is aborted and the new command specified by the trigger is started. 1b - If a higher priority trigger is received during command processing, the current conversion is completed (including averaging iterations if enabled) and stored to the RESFIFO before the higher priority trigger/command is initiated. Note that compare until true commands can be interrupted prior to resulting in a true conversion.

65.4.9 ADC Pause Register (PAUSE)

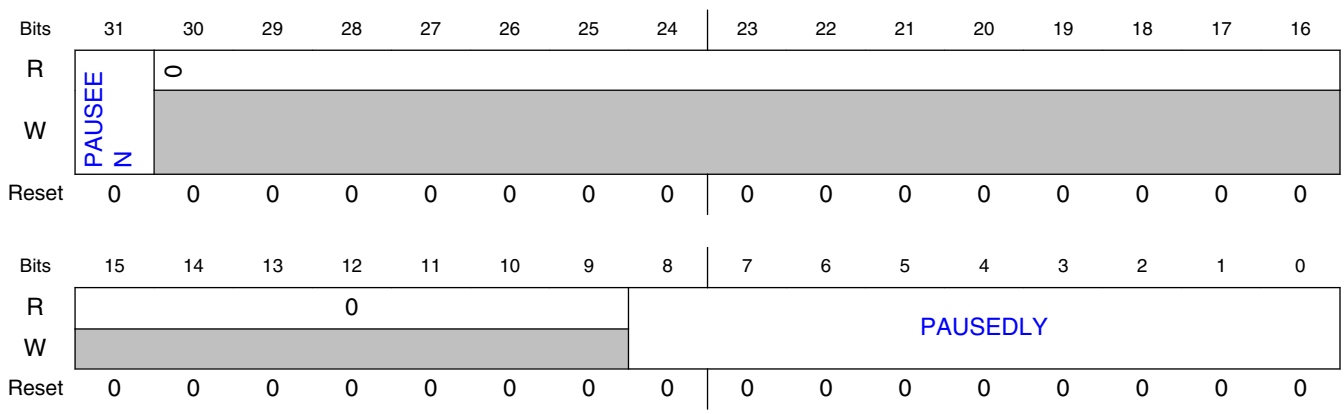
65.4.9.1 Offset

Register	Offset
PAUSE	24h

65.4.9.2 Function

The Pause Register controls an optional inserted delay between conversions. PAUSE cannot be changed while the CTRL[ADCEN] bit is set. Writes to PAUSE while CTRL[ADCEN] is set are ignored.

65.4.9.3 Diagram



65.4.9.4 Fields

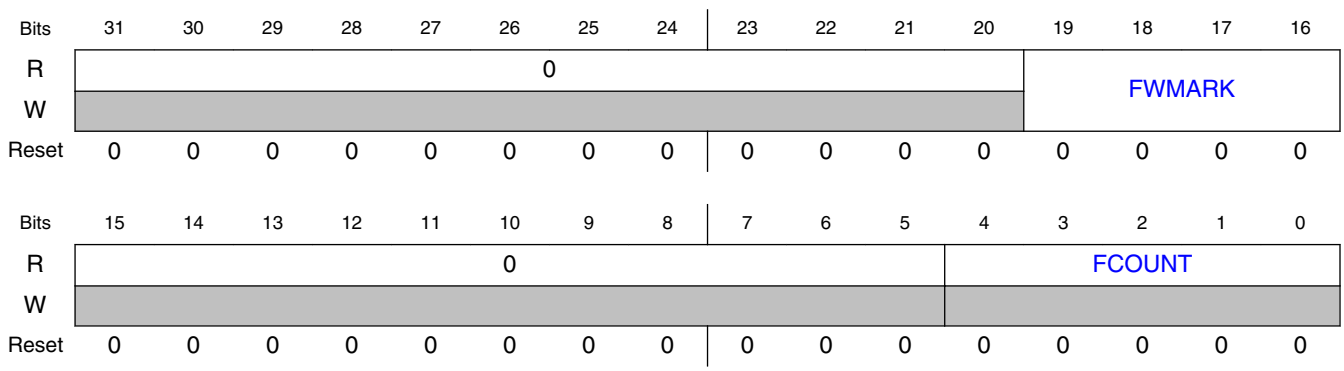
Field	Function
31 PAUSEEN	PAUSE Option Enable Enables the ADC pausing function. When enabled, a programmable delay is inserted during command execution sequencing between LOOP iterations, between commands in a sequence, and between conversions when command is executing in for "Compare Until True" configuration. 0b - Pause operation disabled 1b - Pause operation enabled
30-9 —	Reserved This read-only field is reserved and always has the value 0.
8-0 PAUSEDLY	Pause Delay When PAUSEEN is set, the PAUSEDLY field controls the duration of pausing during command exection sequencing. The pause delay is a count of (PAUSEDLY*4) ADCK cycles.

65.4.10 ADC FIFO Control Register (FCTRL)

65.4.10.1 Offset

Register	Offset
FCTRL	30h

65.4.10.2 Diagram



65.4.10.3 Fields

Field	Function
31-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 FWMARK	Watermark level selection FWMARK is a programmable threshold setting. When the number of datawords stored in the ADC Result FIFO is greater than the value in this field, the STAT[RDY] flag is asserted to indicate stored data has reached the programmable threshold. When IE[FWMIE] is set, an interrupt request is generated. When DE[FWMDE] is set, a DMA request is generated.
15-5 —	Reserved This read-only field is reserved and always has the value 0.
4-0 FCOUNT	Result FIFO counter This read-only field indicates the number of datawords that are stored in the result FIFO. This value may be used in conjunction with PARAM[FIFOSIZE] to calculate how much room is left in the result FIFO. FCOUNT is incremented with each store of new data to the result FIFO and decrements with each read of the result FIFO. The FIFO is reset by writing to the CTRL[RSTFIFO] bit, resulting in FCTRL[FCOUNT] initialized to 0x0.

65.4.11 Software Trigger Register (SWTRIG)

65.4.11.1 Offset

Register	Offset
SWTRIG	34h

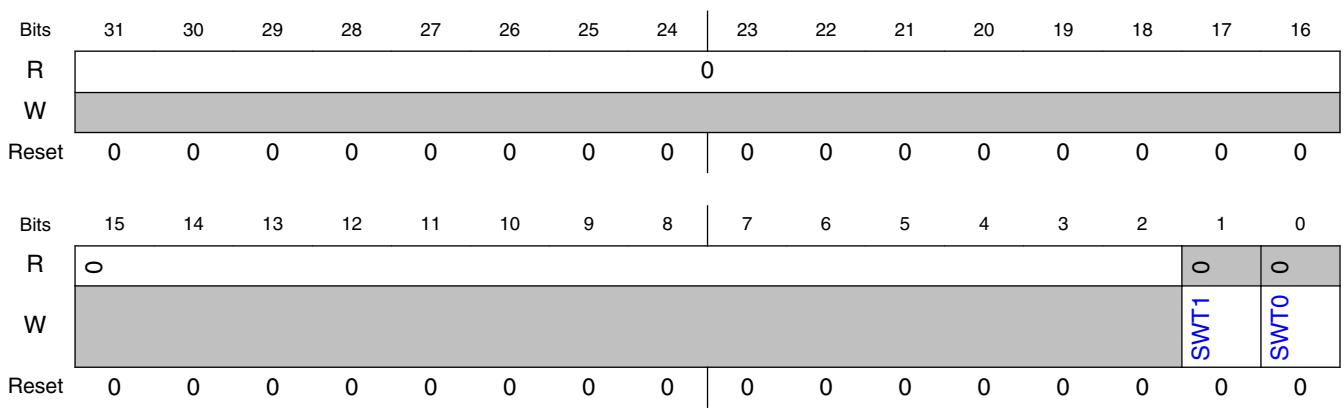
65.4.11.2 Function

The Software Trigger Register (SWTRIG) is written to initiate software triggered conversions. Writes to SWTRIG register are ignored while CTRL[ADCEN] is clear.

NOTE

There is a synchronization delay of 3 ADCK clock cycles when writing to set CTRL[ADCEN]. Writes to SWTRIG will continue to be ignored ultil CTRL[ADCEN] can propagate into the ADCK domain.

65.4.11.3 Diagram



65.4.11.4 Fields

Field	Function
31-2 —	Reserved This read-only field is reserved and always has the value 0.
1 SWT1	Software trigger 1 event Write 1 to SWT1 generates a trigger 1 event. Writing 1 to SWT1 is ignored while the trigger 1 event is being serviced or is pending. 0b - No trigger 1 event generated. 1b - Trigger 1 event generated.
0 SWT0	Software trigger 0 event Write 1 to SWT0 generates a trigger 0 event. Writing 1 to SWT0 is ignored while the trigger 0 event is being serviced or is pending. 0b - No trigger 0 event generated. 1b - Trigger 0 event generated.

65.4.12 Trigger Control Register (TCTRL0 - TCTRL1)

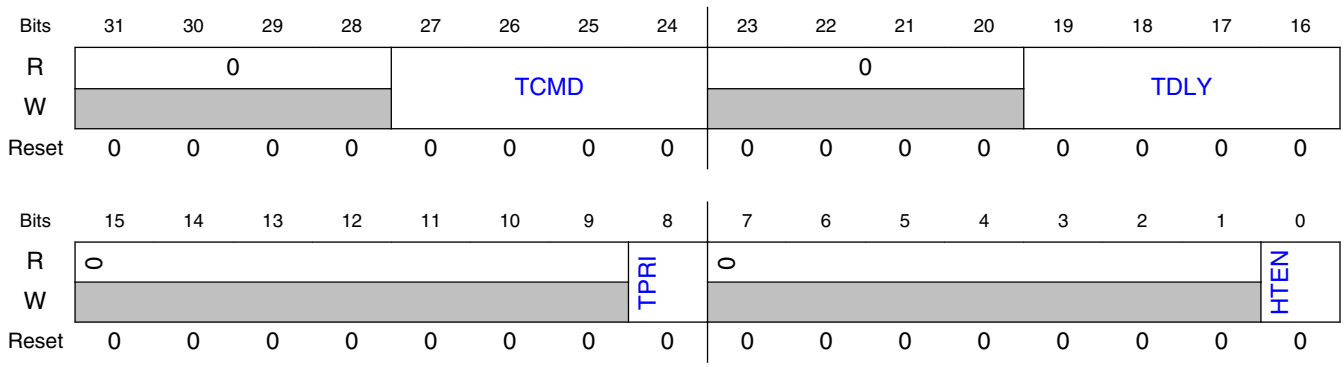
65.4.12.1 Offset

Register	Offset
TCTRL0	C0h
TCTRL1	C4h

65.4.12.2 Function

The Trigger Control (TCTRL*a*) register implements control fields associated with each implemented trigger source. When the ADC is actively executing commands, only one of the TCTRL*a* registers is actively controlling ADC conversions. The actively controlling TCTRL*a* register cannot be updated while the ADC is active. A write to a TCTRL*a* registers while that trigger control register is controlling the ADC operation is ignored.

65.4.12.3 Diagram



65.4.12.4 Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 TCMD	Trigger command select Selects the command from command buffer to execute upon detection of the associated trigger event. When a trigger is received while the ADC is busy converting and the new trigger has higher priority than trigger associated with the current command being executed, the command in this register, associated with the new trigger, is executed under control of CFG[TPRICTRL]. When CFG[TPRICTRL]=0b0, the interruption due to higher priority trigger event is immediate. When CFG[TPRICTRL]=0b1, the current conversion (including any hardware averaging) is allowed to complete and the interruption is at the next loop boundary. 0000b - Not a valid selection from the command buffer. Trigger event is ignored. 0001b - CMD1 is executed 0010-1110b - Corresponding CMD is executed 1111b - CMD15 is executed
23-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 TDLY	Trigger delay select Selects the trigger delay duration to wait at the start of servicing a trigger event. Each trigger source has an associated programmable delay prior to beginning an initial conversion. When TDLY field is clear, then no delay is incurred. When TDLY is set to a non-zero value, the duration for the delay is 2^{TDLY} ADCK cycles.
15-9 —	Reserved This read-only field is reserved and always has the value 0.
8 TPRI	Trigger priority setting This bitfield sets the priority of the associated trigger source. If two or more triggers have the same priority level setting, the lower order trigger event has the higher priority (e. g., if Trigger 0 and Trigger 1 are both pending triggers and TCTRL0[TPRI] is configured the same as TCTRL1[TPRI], then the command associated with Trigger 0 is serviced first). 0b - Set to highest priority, Level 1

Table continues on the next page...

Field	Function
	1b - Set to lower priority, Level 2
7-1 —	Reserved This read-only field is reserved and always has the value 0.
0 HTEN	Trigger enable Enables hardware trigger source to initiate conversion on the rising-edge of the input trigger source. **NOTE:** Enabling hardware trigger does not disable software triggers. 0b - Hardware trigger source disabled 1b - Hardware trigger source enabled

65.4.13 ADC Command Low Buffer Register (CMDL1 - CMDL15)

65.4.13.1 Offset

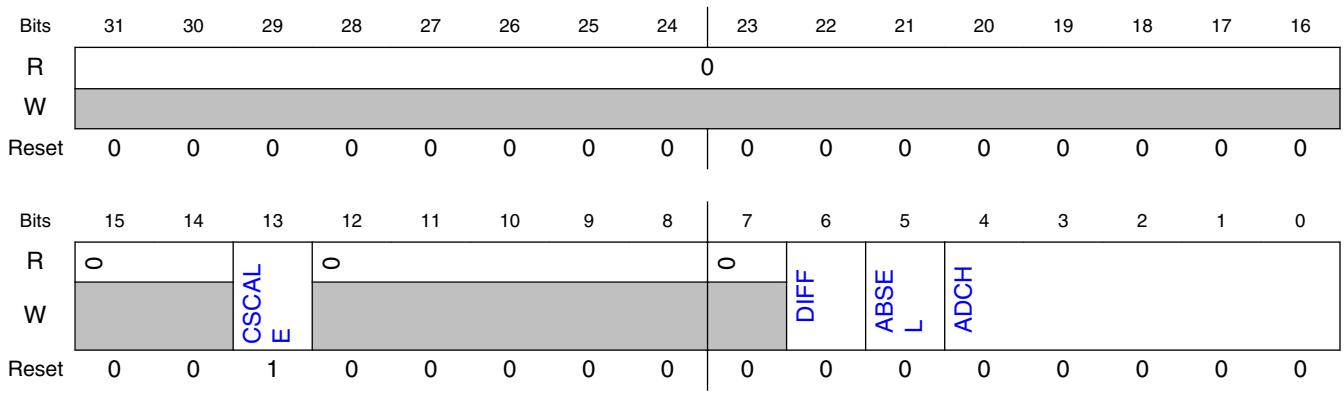
For a = 1 to 15:

Register	Offset
CMDLa	F8h + (a × 8h)

65.4.13.2 Function

There are 15 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDL*a*:CMDH*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling command by association to a trigger event via configuration of the TCTRL*a*[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer cannot be updated while the ADC is active. A write to a CMD buffer while that CMD buffer is controlling the ADC operation is ignored.

65.4.13.3 Diagram



65.4.13.4 Fields

Field	Function
31-14 —	Reserved This read-only field is reserved and always has the value 0.
13 CSCALE	Channel Scale Reduces the selected ADC analog channel input voltage level by a factor. The maximum possible voltage on the ADC channel input should be considered when selecting a CSCALE value to ensure that the reducing factor always results voltage level at or below the VREFH reference. This reducing capability allows conversion of analog inputs higher than VREFH. A-side and B-side channel inputs are both scaled using the CSCALE field. 0b - Scale selected analog channel (Factor of 30/64) 1b - (Default) Full scale (Factor of 1)
12-8 —	Reserved This read-only field is reserved and always has the value 0.
7 —	Reserved This read-only field is reserved and always has the value 0.
6 DIFF	Differential Mode Enable Configures the ADC to operate in differential mode or single-ended mode. 0b - Single-ended mode. 1b - Differential mode.
5 ABSEL	A-side vs. B-side Select When DIFF=0b0, ABSEL selects the channel from either the A-side or B-side channel inputs. When DIFF=0b1, ABSEL configures the ADC result to be (CHnA-CHnB) or (CHnB-CHnA). 0b - When DIFF=0b0, the associated A-side channel is converted as single-ended. When DIFF=0b1, the ADC result is (CHnA-CHnB). 1b - When DIFF=0b0, the associated B-side channel is converted as single-ended. When DIFF=0b1, the ADC result is (CHnB-CHnA).
4-0 ADCH	Input channel select

Field	Function
	Each ADCH channel selection has an associated A side and an associated B side input. The ADCH setting in conjunction with the DIFF and ABSEL bitfield settings selects the input from one of the channel inputs or one of the input pairs. See the DIFF and ABSEL bitfield descriptions. **NOTE:** Not all pairs are intended to be converted as differential channels. For the pin pairings available for differential conversions for your device, see the Chip Configuration details. **NOTE:** Some of the input channels are from internal resources such as temperature sensors and bandgap voltage sources and may only be connected to individual instances of the ADC module. Some of the input channel options in the bitfield-setting descriptions might not be available for your device. For the ADC channel assignments for your device, see the chip-specific information. 00000b - Select CH0A or CH0B or CH0A/CH0B pair. 00001b - Select CH1A or CH1B or CH1A/CH1B pair. 00010b - Select CH2A or CH2B or CH2A/CH2B pair. 00011b - Select CH3A or CH3B or CH3A/CH3B pair. 00100-11101b - Select corresponding channel CHnA or CHnB or CHnA/CHnB pair. 11110b - Select CH30A or CH30B or CH30A/CH30B pair. 11111b - Select CH31A or CH31B or CH31A/CH31B pair.

65.4.14 ADC Command High Buffer Register (CMDH1 - CMDH4)

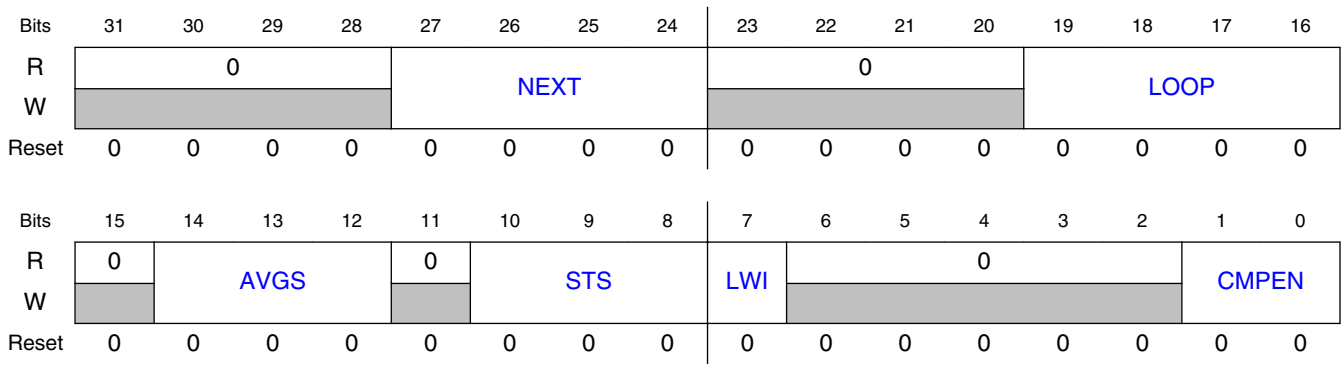
65.4.14.1 Offset

Register	Offset
CMDH1	104h
CMDH2	10Ch
CMDH3	114h
CMDH4	11Ch

65.4.14.2 Function

There are 15 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDL*a*:CMDH*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling command by association to a trigger event via configuration of the TCTRL*a*[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer cannot be updated while the ADC is active. A write to a CMD buffer while that CMD buffer is controlling ADC operation is ignored.

65.4.14.3 Diagram



65.4.14.4 Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 NEXT	Next Command Select Selects the next command to be executed after this command completes. Multiple commands can be configured in a scan configuration by linking the next command in a daisy-chain sequence. The command buffer number is not indicative of any particular order and the order of execution is strictly controlled by the NEXT field (for example, a sequence of commands could be CMD2, CMD1, CMD3). Unending circular command execution is allowed by setting the NEXT field in the last command in a sequence to the first command in the sequence. It is also allowed for a command to set the next command to itself, resulting in a continuous conversion configuration. Setting the next command to 0x0 causes conversions to terminate at the completion of the command. Lower priority trigger events cannot be serviced until a higher priority triggered command (or sequence of commands) completes. 0000b - No next command defined. Terminate conversions at completion of current command. If lower priority trigger pending, begin command associated with lower priority trigger. 0001b - Select CMD1 command buffer register as next command. 0010-1110b - Select corresponding CMD command buffer register as next command 1111b - Select CMD15 command buffer register as next command.
23-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 LOOP	Loop Count Select Selects how many times this command executes (and stores conversion result to RESFIFO) before finish and transition to the next command or Idle state. LWI field controls whether a single channel is converted on each iteration or auto channel increment results in channel scanning functionality. 0000b - Looping not enabled. Command executes 1 time. 0001b - Loop 1 time. Command executes 2 times. 0010b - Loop 2 times. Command executes 3 times. 0011-1110b - Loop corresponding number of times. Command executes LOOP+1 times. 1111b - Loop 15 times. Command executes 16 times.
15 —	Reserved This read-only field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
14-12 AVGS	Hardware Average Select Selects how many ADC conversions are averaged to create the ADC result (2^{AVGS}). An internal storage buffer is used to capture temporary results while the averaging iterations are executed. Hardware averaging is a nested loop control and does not extend across LOOP boundaries. See Functional description for more detailed description on usage of AVGS, LOOP, and NEXT bitfields in command execution sequencing. 000b - Single conversion. 001b - 2 conversions averaged. 010b - 4 conversions averaged. 011b - 8 conversions averaged. 100b - 16 conversions averaged. 101b - 32 conversions averaged. 110b - 64 conversions averaged. 111b - 128 conversions averaged.
11 —	Reserved This read-only field is reserved and always has the value 0.
10-8 STS	Sample Time Select When programmed to 000 the minimum sample time of 3 ADCK cycles is selected. When STS is programmed to a non-zero value the sample time is ($3 + 2^{STS}$) ADCK cycles. The shortest sample time maximizes conversion speed for lower impedance inputs. Extending sample time allows higher impedance inputs to be accurately sampled. Longer sample times can also be used to lower overall power consumption when command looping and sequencing is configured and high conversion rates are not required. 000b - Minimum sample time of 3 ADCK cycles. 001b - $3 + 2^1$ ADCK cycles; 5 ADCK cycles total sample time. 010b - $3 + 2^2$ ADCK cycles; 7 ADCK cycles total sample time. 011b - $3 + 2^3$ ADCK cycles; 11 ADCK cycles total sample time. 100b - $3 + 2^4$ ADCK cycles; 19 ADCK cycles total sample time. 101b - $3 + 2^5$ ADCK cycles; 35 ADCK cycles total sample time. 110b - $3 + 2^6$ ADCK cycles; 67 ADCK cycles total sample time. 111b - $3 + 2^7$ ADCK cycles; 131 ADCK cycles total sample time.
7 LWI	Loop with Increment When LWI is clear, the LOOP field selects the number of times the selected channel is converted consecutively. When LWI is set, auto channel incrementing is enabled and the LOOP field defines how many consecutive channels are converted as part of the command execution. Example 1: LOOP = 0x8, LWI = 0b0, ABSEL = 0b0, ADCH = 0xD. Convert on channel 13A 8 times. Example 2: LOOP = 0x8, LWI= 0b1, ABSEL = 0b0, ADCH = 0xD. Run channels 13A, 14A, 15A, ... 20A each one time. Maximum channel scanning using a single command buffer then is defined by the maximum value of the LOOP field (16). Note, when LWI is set, the minimum sample length must be extended to CMDHa[STS] = 0x3. 0b - Auto channel increment disabled 1b - Auto channel increment enabled
6-2 —	Reserved This read-only field is reserved and always has the value 0.
1-0 CMPEN	Compare Function Enable After an ADC channel input is sampled and converted and any averaging iterations are performed, the CMDHa[CMPEN] field guides operation of the automatic compare function to optionally only store when the compare operation is true. When compare is enabled, the conversion result is compared to the

Field	Function
	compare value registers (CV*a*[CVH] and CV*a*[CVL]). There are multiple options on command sequencing related to the compare function. See Compare function for a detailed explanation of the compare options. **NOTE:** Not all Command Buffers have implemented the CMPEN field. The CMPEN field is only available in Command Buffers that have a corresponding Compare Value register. 00b - Compare disabled. 01b - Reserved 10b - Compare enabled. Store on true. 11b - Compare enabled. Repeat channel acquisition (sample/convert/compare) until true.

65.4.15 ADC Command High Buffer Register (CMDH5 - CMDH15)

65.4.15.1 Offset

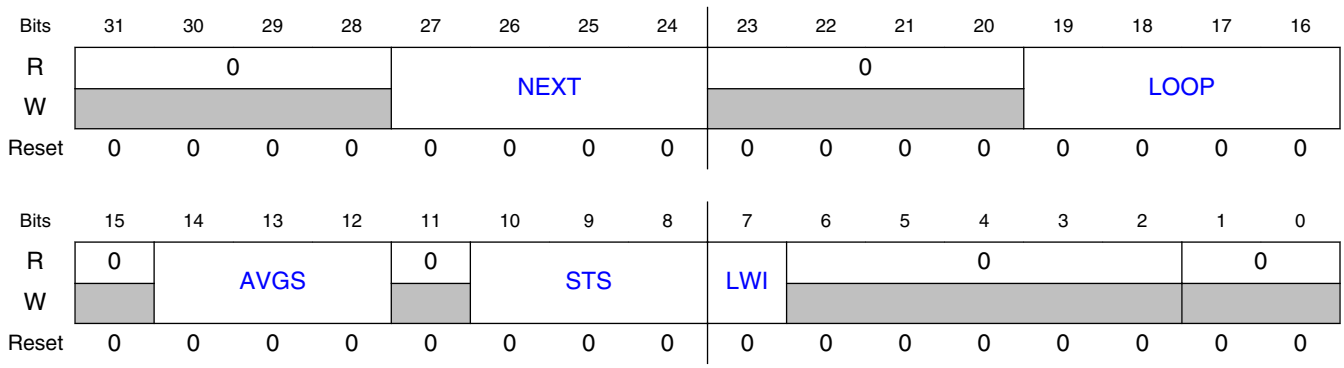
For a = 5 to 15:

Register	Offset
CMDHa	FCh + (a × 8h)

65.4.15.2 Function

There are 15 command buffers (CMD*a*), each constructed from two 32-bit registers (CMDL*a*:CMDH*a*) that can be configured for different channel select and varying conversion options. Any of the command buffers is selected and used as the controlling command by association to a trigger event via configuration of the TCTRL*a*[TCMD] bitfield. When the ADC is actively executing commands, only one of the CMD buffers is actively controlling ADC conversions. The actively controlling CMD buffer cannot be updated while the ADC is active. A write to a CMD buffer while that CMD buffer is controlling ADC operation is ignored.

65.4.15.3 Diagram



65.4.15.4 Fields

Field	Function
31-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 NEXT	Next Command Select Selects the next command to be executed after this command completes. Multiple commands can be configured in a scan configuration by linking the next command in a daisy-chain sequence. The command buffer number is not indicative of any particular order and the order of execution is strictly controlled by the NEXT field (for example, a sequence of commands could be CMD2, CMD1, CMD3). Unending circular command execution is allowed by setting the NEXT field in the last command in a sequence to the first command in the sequence. It is also allowed for a command to set the next command to itself, resulting in a continuous conversion configuration. Setting the next command to 0x0 causes conversions to terminate at the completion of the command. Lower priority trigger events cannot be serviced until a higher priority triggered command (or sequence of commands) completes. 0000b - No next command defined. Terminate conversions at completion of current command. If lower priority trigger pending, begin command associated with lower priority trigger. 0001b - Select CMD1 command buffer register as next command. 0010-1110b - Select corresponding CMD command buffer register as next command 1111b - Select CMD15 command buffer register as next command.
23-20 —	Reserved This read-only field is reserved and always has the value 0.
19-16 LOOP	Loop Count Select Selects how many times this command executes (and stores conversion result to RESFIFO) before finish and transition to the next command or Idle state. LWI field controls whether a single channel is converted on each iteration or auto channel increment results in channel scanning functionality. 0000b - Looping not enabled. Command executes 1 time. 0001b - Loop 1 time. Command executes 2 times. 0010b - Loop 2 times. Command executes 3 times. 0011-1110b - Loop corresponding number of times. Command executes LOOP+1 times. 1111b - Loop 15 times. Command executes 16 times.
15 —	Reserved This read-only field is reserved and always has the value 0.

Table continues on the next page...

Field	Function
14-12 AVGS	Hardware Average Select Selects how many ADC conversions are averaged to create the ADC result (2^{AVGS}). An internal storage buffer is used to capture temporary results while the averaging iterations are executed. Hardware averaging is a nested loop control and does not extend across LOOP boundaries. See Functional description for more detailed description on usage of AVGS, LOOP, and NEXT bitfields in command execution sequencing. 000b - Single conversion. 001b - 2 conversions averaged. 010b - 4 conversions averaged. 011b - 8 conversions averaged. 100b - 16 conversions averaged. 101b - 32 conversions averaged. 110b - 64 conversions averaged. 111b - 128 conversions averaged.
11 —	Reserved This read-only field is reserved and always has the value 0.
10-8 STS	Sample Time Select When programmed to 000 the minimum sample time of 3 ADCK cycles is selected. When STS is programmed to a non-zero value the sample time is ($3 + 2^{STS}$) ADCK cycles. The shortest sample time maximizes conversion speed for lower impedance inputs. Extending sample time allows higher impedance inputs to be accurately sampled. Longer sample times can also be used to lower overall power consumption when command looping and sequencing is configured and high conversion rates are not required. 000b - Minimum sample time of 3 ADCK cycles. 001b - $3 + 2^1$ ADCK cycles; 5 ADCK cycles total sample time. 010b - $3 + 2^2$ ADCK cycles; 7 ADCK cycles total sample time. 011b - $3 + 2^3$ ADCK cycles; 11 ADCK cycles total sample time. 100b - $3 + 2^4$ ADCK cycles; 19 ADCK cycles total sample time. 101b - $3 + 2^5$ ADCK cycles; 35 ADCK cycles total sample time. 110b - $3 + 2^6$ ADCK cycles; 67 ADCK cycles total sample time. 111b - $3 + 2^7$ ADCK cycles; 131 ADCK cycles total sample time.
7 LWI	Loop with Increment When LWI is clear, the LOOP field selects the number of times the selected channel is converted consecutively. When LWI is set, auto channel incrementing is enabled and the LOOP field defines how many consecutive channels are converted as part of the command execution. Example 1: LOOP = 0x8, LWI = 0b0, ABSEL = 0b0, ADCH = 0xD. Convert on channel 13A 8 times. Example 2: LOOP = 0x8, LWI= 0b1, ABSEL = 0b0, ADCH = 0xD. Run channels 13A, 14A, 15A, ... 20A each one time. Maximum channel scanning using a single command buffer then is defined by the maximum value of the LOOP field (16). Note, when LWI is set, the minimum sample length must be extended to CMDHa[STS] = 0x3. 0b - Auto channel increment disabled 1b - Auto channel increment enabled
6-2 —	Reserved This read-only field is reserved and always has the value 0.
1-0 —	Reserved This read-only field is reserved and always has the value 0.

65.4.16 Compare Value Register (CV1 - CV4)

65.4.16.1 Offset

Register	Offset
CV1	200h
CV2	204h
CV3	208h
CV4	20Ch

65.4.16.2 Function

The compare value registers (CV*a*) contain values used to compare the conversion result when the compare function is enabled. This register is formatted in the same way as the D field in the RESFIFO registers in different modes of operation for both bit position definition and value format using unsigned or signed 2's complement. There is a direct association of each compare value register to a specific command buffer register (i.e., CV1 is only used during execution of CMD1 command).

NOTE

Not all Command Buffers have an associated Compare Value register. The compare function is only available on Command Buffers that have a corresponding Compare Value register.

When the ADC is actively executing commands, the CV*a* register associated with the active command (CMD*a*) cannot be updated and writes to associated CV*a* register during this time are ignored.

65.4.16.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CVH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CVL															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

65.4.16.4 Fields

Field	Function
31-16 CVH	Compare Value High. The compare function can be configured to check whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. After the input is sampled and converted and any averaging iterations are performed, the compare values in CVL and CVH are optionally used in a compare operation on the result. See Compare function for a description on how CVH is used.
15-0 CVL	Compare Value Low. The compare function can be configured to check whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. After the input is sampled and converted and any averaging iterations are performed, the compare values in CVL and CVH are optionally used in a compare operation on the result. See Compare function for a description on how CVL is used.

65.4.17 ADC Data Result FIFO Register (RESFIFO)

65.4.17.1 Offset

Register	Offset
RESFIFO	300h

65.4.17.2 Function

The data result FIFO register (RESFIFO) is a 16 entry FIFO that stores the data result of ADC conversions. In addition, several tag fields of source command and trigger information are stored along with the data. FCTRL[FCOUNT] indicates how many valid datawords are stored in the RESFIFO. Reading RESFIFO provides the oldest unread dataword entry in the FIFO and decrements FCOUNT. The FIFO can be emptied by successive reads of RESFIFO. The FIFO is reset by writing 0b1 to the CTRL[RSTFIFO] bit.

The following table describes the format of data in the result FIFO in the different modes of operation. The sign bit is the (MSB) in signed 2's complement modes. For example, when configured for 12-bit single-ended mode, D[15] and D[2:0] are cleared. When configured for 13-bit differential mode, D[15] is the sign bit and D[2:0] are cleared.

Table 65-6. Data result register format description

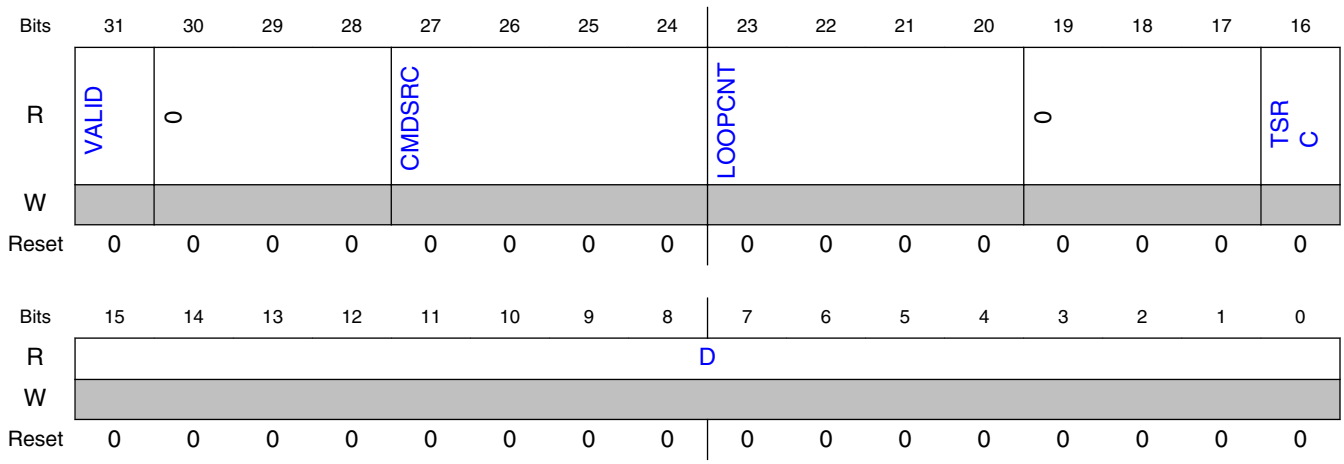
Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
13-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	Signed 2's complement, left-justified, zero extended
12-bit single-ended	0	D	D	D	D	D	D	D	D	D	D	D	D	0	0	0	Unsigned, zero in D[15] and D[2:0]

NOTE

S: Sign bit;

D: Data, which is 2's complement data if indicated

65.4.17.3 Diagram



65.4.17.4 Fields

Field	Function
31 VALID	FIFO entry is valid Indicate the FIFO entry is valid. When the FIFO is holding data the VALID bit is set. When the FIFO is empty the VALID bit is clear. 0b - FIFO is empty. Discard any read from RESFIFO. 1b - FIFO record read from RESFIFO is valid.
30-28 —	Reserved This read-only field is reserved and always has the value 0.
27-24 CMDSRC	Command Buffer Source Indicates the command buffer being executed that generated this result. 0000b - Not a valid value CMDSRC value for a dataword in RESFIFO. 0x0 is only found in initial FIFO state prior to an ADC conversion result dataword being stored to a RESFIFO buffer. 0001b - CMD1 buffer used as control settings for this conversion. 0010-1110b - Corresponding command buffer used as control settings for this conversion. 1111b - CMD15 buffer used as control settings for this conversion.
23-20 LOOPCNT	Loop count value Indicates the loop count value during command execution that generated this result. When CMDHa[LOOP] is non-zero, the command execution stores off a result multiple times during command execution (at the loop boundary). 0000b - Result is from initial conversion in command. 0001b - Result is from second conversion in command. 0010-1110b - Result is from LOOPCNT+1 conversion in command. 1111b - Result is from 16th conversion in command.
19-17 —	Reserved This read-only field is reserved and always has the value 0.
16 TSRC	Trigger Source

Table continues on the next page...

Field	Function
	Indicate the trigger source that initiated a conversion and generated this result. When multiple commands are chained together using the NEXT field, the TSRC field for all datawords stored to the RESFIFO indicate the trigger source that started the sequence of commands. 0b - Trigger source 0 initiated this conversion. 1b - Trigger source 1 initiated this conversion.
15-0 D	Data result The formatting for the data in D is summarized in Table 65-6.

65.5 Functional description

The ADC module performs analog-to-digital conversions on any of the software selectable analog input channels by a successive approximation algorithm. The module initializes to its lowest power state during reset and in Low-Power Stop mode. The ADC analog circuits can optionally be pre-enabled for faster starts to conversions at the expense of higher idle currents. Conversions are initiated by selectable trigger events from software or hardware sources. The trigger detect logic includes a configurable enable and priority scheme for the available trigger sources. The module includes multiple command buffers that provide configurable flexibility for channel scanning and independent channel selections for different trigger sources. Multiple command buffers also allow variable option selection such as input scaling factor,differential vs. single-ended, sample time and averaging on a per-channel basis.

This module optionally averages the result of multiple conversions on a channel before storing the calculated result. The hardware average function is enabled by setting CMDH*a*[AVGS] bitfield to a non-zero value and operates in any of the conversion modes and configurations.

When the conversion and averaging loops are completed, the resulting data is placed in a FIFO data buffer along with other tag information associated with the result. A configurable watermark level supports interrupt or DMA requests when the number of stored datawords exceeds the setting.

The ADC module optionally compares the result of a conversion with the contents of two value registers for less-than, greater-than, inside-range or outside-range detection. The compare function operates in any of the conversion modes and configurations.

The sequencing of an ADC command is summarized in the flow diagram shown below.

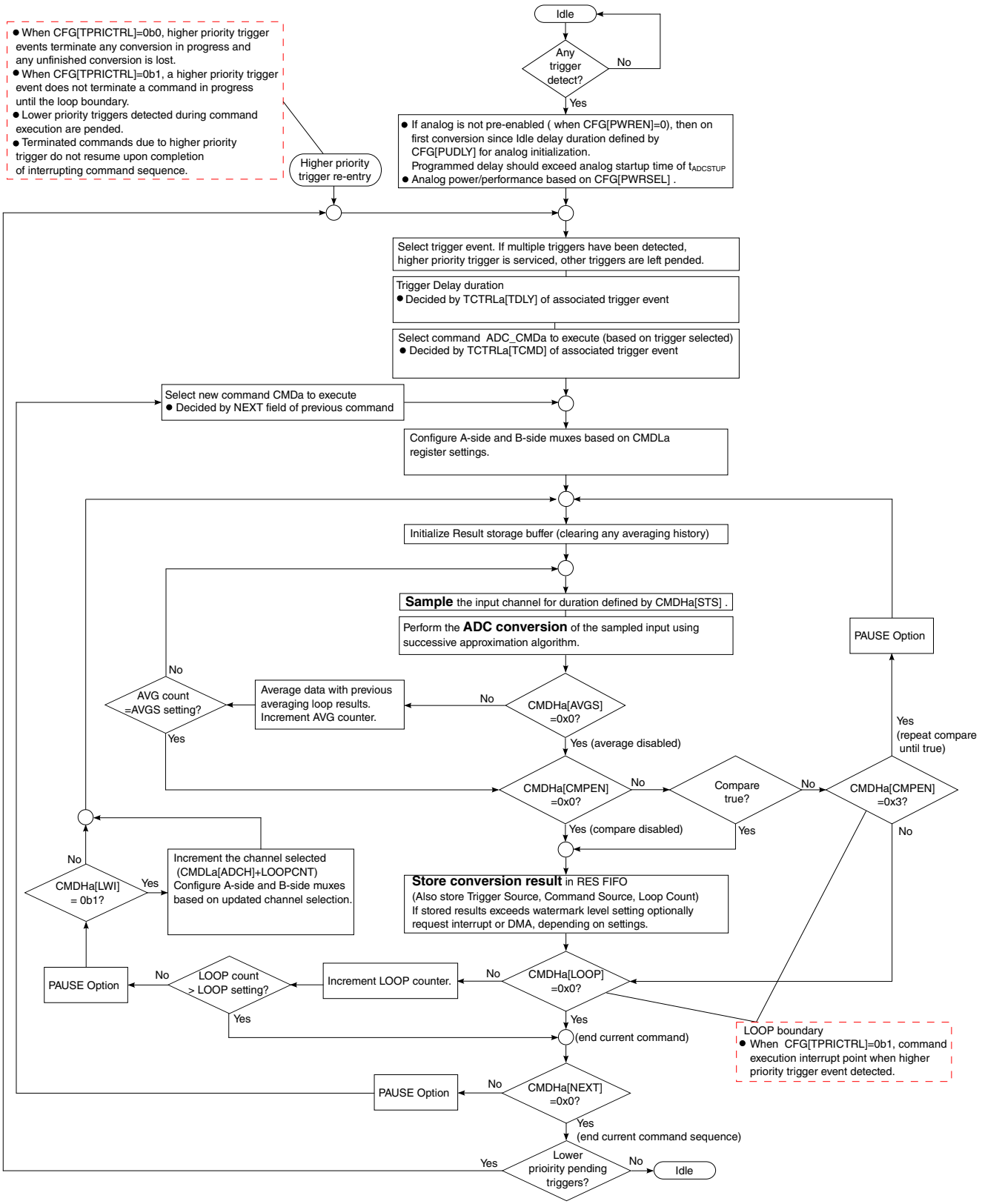


Figure 65-2. ADC command execution flow diagram

65.5.1 Voltage reference

The voltage reference high (V_{REFH}) used by the ADC is supplied from either an on-chip voltage reference source or from an off-chip source supplied through external pins. V_{REFL} is always from an external pin and must be at the same voltage potential as V_{SSA}. See the chip configuration information on the voltage reference options specific to this packaged device.

This block supports a programmable selection of the Voltage Reference High used for ADC conversions (via the CFG[REFSEL] field). See the chip configuration information on the voltage reference options specific to this packaged device.

65.5.2 Power control

The default setting for the ADC analog circuits is disabled while the ADC is in its Idle state. When a trigger is detected and ADC command processing is initiated, the analog circuits are enabled and require a period of initialization before the first conversion cycle. The CFG[PUDLY] should be programmed such that a delay duration longer than $t_{ADCSTUP}$ is incurred. Accuracy of the initial conversion(s) after activation is degraded if CFG[PUDLY] is set to too small a value.

Faster conversion startup times can be achieved by optionally setting the CFG[PWREN] field to pre-enable the analog circuits of the ADC at the expense of power consumption even while the module is in an idle state. When CFG[PWREN] is set, the Power Enable timer is activated and enforces the minimum startup (controlled by the PUDLY register field) before detected triggers are allowed to initiate ADC conversions.

This module also has configuration options for controlling power and performance summarized in the table below. See the device datasheet for specification of the available power modes.

Table 65-7. Power option settings

CFG[PWRSEL]	Description
0b00 (Default)	Level 1. Slowest speed/Lowest power setting.
0b01	Level 2.
0b10	Level 3.
0b11	Level 4. Fastest speed/highest power setting.

In the two highest power mode settings (CFG[PWRSEL]=0x2 or CFG[PWRSEL]=0x3) the ADC will delay start of sampling for 16 ADC clock cycles on the initial conversion since being in its Idle state. This delay is in addition to any configured Power Up delay or trigger delay.

65.5.3 Clock operation

The ADC operates from the ADCK clock input provided from an on-chip clock select block and is used by the SAR conversion control sequencing logic and the FIFO storage buffer. The ADCK frequency must fall within the specified frequency range for ADCK and will vary based on configuration of CFG[PWRSEL]. See the device datasheet for supported frequency ranges.

The ADC continues operating in stop and wait modes provided the Doze Enable bit (CTRL[DOZEN]) is clear and the on-chip clock select block continues to supply an ADCK clock source. When the CTRL[DOZEN] bit is set, the module will wait for the current averaging iteration/FIFO storage to complete before acknowledging stop or wait mode entry.

65.5.4 Trigger detect and command execution

See Figure 65-2 for a flow diagram of command execution sequencing.

ADC command execution is initiated from up to 2 trigger sources. Each trigger can be software generated by writing 0b1 to the corresponding SWTRIG[SWT*n*] bitfield. Alternatively, hardware triggers can be generated from asynchronous input sources at the periphery of the module. The number and sources of hardware triggers implemented is device-specific. See the chip configuration information for description of available hardware trigger sources for this device.

Each hardware trigger source is enabled by setting the associated enable bit (TCTRL*a*[HTEN]). Each trigger source is assigned a priority via the associated priority control field (TCTRL*a*[TPRI]). Each of the trigger sources is configurably associated with a command in the command buffer via the associated command select field (TCTRL*a*[TCMD]).

When a hardware trigger input is enabled, hardware trigger events are detected on the rising-edge of the associated hardware trigger source.

If a trigger event occurs that is configured for a higher priority than the trigger source associated with the currently executing command, the current command is handled in one of two ways depending on the CFG[TPRICTRL] bitfield setting. When

CFG[TPRICTRL]=0b0, a higher priority trigger causes an immediate command abort and the new command specified by the trigger is immediately started. When CFG[TPRICTRL]=0b1, a higher priority triggers allows the current conversion to complete (including averaging and compare functions) before the higher priority command is initiated.

If a lower priority trigger occurs (i.e., a trigger event occurs that is configured for a lower priority than the trigger source associated with the currently executing command), the trigger detect is pended until completion of the current command sequence.

At the completion of a command, the ADC does not resume a previously interrupted command (or series of commands). The module begins command execution associated with the processing of lower priority pending triggers. If no trigger events are pending, the ADC transitions to its idle state.

When a conversion is completed (including hardware averaging when AVGS is non-zero), the result is placed in the RESFIFO buffer. When an ADC command selects looping (when LOOP is non-zero) a command will store multiple conversion results to the FIFO during execution of that command.

At the end of command execution, the NEXT field of the command selects the next command to be executed. Multiple commands can be executed sequentially by configuration of each command's NEXT field. Setting the next command to 0x0 causes conversions to terminate at the completion of the command. Unending circular command execution is allowed by setting the NEXT field in the last command in a sequence to the first command in the sequence. It is also allowed for a command to set the next command to itself, resulting in a continuous conversion configuration. Lower priority trigger events cannot be serviced until a higher priority triggered command (or command sequence) completes.

Disabling the ADC by writing 0b0 to the CTRL[ADCEN] bitfield terminates any active ADC command processing. Writing 0b0 to the CTRL[ADCEN] bitfield causes the current command (or command sequence) to terminate, clears any pending triggers, and sends the module to an Idle state.

65.5.5 Pause option

When the maximum conversion rate is not required by an application the effective conversion rate can be reduced by implementing periodic trigger events to initiate ADC conversions or by selecting a reduced frequency clock as the ADACK source. Both of these options are chip specific and are dependent on ADC triggering and clocking options external to the ADC module. The latency associated with ADC analog power up delays results in a limit on the maximum conversion rate when using periodic triggering.

Another means of reducing conversion rates is by inserting a pause of a programmable duration between LOOP iterations, between commands in a sequence, and between conversions when command is executing in the "Compare Until True" configuration. When PAUSE[PAUSEEN] is set, the PAUSE[PAUSEDLY] field controls the duration of pausing during command exection sequencing. The pause delay is a count of (PAUSEDLY*4) ADCK cycles. The PAUSE register cannot be changed while the CTRL[ADCEN] bit is set. Writes to the PAUSE register while CTRL[ADCEN] is set are ignored.

See Figure 65-2 for the places during command execution sequencing where the pause is optionally inserted.

65.5.6 Result FIFO operation

The ADC includes a 16 entry FIFO in which the result of ADC conversions are stored. In addition, a valid indicator bit, the trigger source, the source command and the loop count are also stored along with the data. FCTRL[FCOUNT] indicates how many valid datawords are stored in the RESFIFO.

A programmable watermark threshold supports configurable notification of data availability. When FCOUNT is greater than FCTRL[FWMARK], the STAT[RDY] flag is asserted. When IE[FWMIE] is set, a watermark interrupt request is issued. When DE[FWMDE] is set, a DMA request is issued. Reading RESFIFO provides the oldest unread dataword entry in the FIFO and decrements FCOUNT. When FCOUNT falls equal to or below FWMARK, the RDY flag is cleared.

The FIFO can be emptied by successive reads of RESFIFO. When the RESFIFO[VALID] bit is 1, the associated FIFO entry is valid. Reading RESFIFO when the FIFO is empty (when RESFIFO[VALID] is clear and FCOUNT=0x0) provides an undefined dataword. The FIFO is reset by writing 0b1 to the CTRL[RSTFIFO] bit.

If the ADC attempts to store a dataword to the FIFO when the FIFO is full, the FIFO overflow flag (FCTRL[FOF]) is set. When IE[FOFIE] is set, an overflow interrupt request is issued. The STAT[FOF] flag is cleared by writing 1 to FOF. On overflow events, no new data is stored and the data associated with the store that triggered the overflow is lost, the current ADC command (sequence) is aborted and all pending trigger events are discarded. No new triggers are detected until the overflow flag is cleared.

65.5.7 Compare function

After the input is sampled and converted and any averaging iterations are performed, the CMDH*a*[CMPEN] field guides operation of the automatic compare function to optionally only store when the compare operation is true. There are multiple options on command sequencing related to the compare function as summarized in the table below.

NOTE

Not all Command Buffers have an associated Compare Value register. The compare function is only available on Command Buffers that have a corresponding Compare Value register.

Table 65-8. Compare modes

CMDH*a*[CMPEN]	Compare Function	Description
0b00	Compare disabled	Do not perform compare operation. Always store the conversion result to the FIFO.
0b01	Reserved	
0b10	Store on true	Perform compare operation. Store conversion result to FIFO at end of averaging only if compare is true. If compare is false, do not store the result to the FIFO. In either the true or false condition, the LOOP setting is considered and increments the LOOP counter before deciding whether the current command has completed or additional LOOP iterations are required.
0b11	Repeat compare until true	Perform compare operation. Store conversion result to FIFO at end of averaging only if compare is true. Once the true condition is found the LOOP setting is considered and increments the LOOP counter before deciding whether the current command has completed or additional LOOP iterations are required. If the compare is false do not store the result to the FIFO. The conversion is repeated without consideration of LOOP setting and does not increment the LOOP counter.

Depending on CV*a*[CVH] and CV*a*[CVL] values programmed, the compare operation checks whether the result is less than, greater than, or if the result falls within or outside a range determined by two compare values. The compare values are used as described in the following table.

Table 65-9. Compare operations

CV*a*[CVL] vs. CV*a*[CVH]	Operation	Description
set CVL < CVH	Outside range (General form)	Compare true if the result is less than CVL value or greater than CVH value.
set CVH to max value set CVL to compare point	Less than	Compare true if the result is less than CVL value.
set CVL to min value set CVH to compare point	Greater than	Compare true if the result is greater than CVH value.

Table continues on the next page...

Table 65-9. Compare operations (continued)

CV*a*[CVL] vs. CV*a*[CVH]	Operation	Description
set CVL > CVH	Inside range	Compare true if the result is less than CVL value and greater than CVH value.

Note

In low power modes where the ADC continues to operate, the compare function can monitor the voltage and only wake the device when the compare condition is met.

Chapter 66
Digital to Analog Converter (DAC)

66.1 Chip-specific DAC information

Table 66-1. Reference links to related information

Topic	Related module	Reference
Full description	DAC	DAC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

66.1.1 Digital to Analog Converter (DAC)

The DAC is the 12-bit resolution digital-to-analog converters with programmable reference generator output. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator or ADC. The DAC is capable of achieving 1 microsecond conversion rate for high speed signal and 2 microsecond conversion rate for low speed signal. The following table shows the configuration of the DAC.

Table 66-2. DAC configuration

Parameter	Description
Name	Digital to Analog Converter (DAC)
Instances	2
Configurable features	NA
Interface speed	NA
External I/O pins	DAC_OUT

66.1.2 DAC reference sources

The CR[DACRFS] bit selects the two possible reference sources, DACREF_1 or DACREF_2.

- When DACRFS =0b (DACREF_1), the reference voltage is VREFH_ANA18.
- When DACRFS =1b (DACREF_2), the reference voltage, is VDD_ANA18.

See the register section for details.

NOTE

If DAC is required to have conversion in VLPS or VLLS mode, the bias from PMC 0 needs to be enabled. Bias are enabled from PMC by setting VLPS[FBGHP] bitfield for VLPS mode and VLLS[FBGHP] bitfield for VLLS mode in the PMC 0 register. See PMC register descriptions for details.

66.2 Introduction

The 12-bit digital-to-analog converter (DAC) is a low-power, general-purpose DAC. The output of the DAC can be placed on an external pin or set as one of the inputs to the analog comparator, op-amps, or ADC.

66.3 Features

The features of the DAC module include:

- On-chip programmable reference generator output. The voltage output range is from 1⁄4096 V_{in} to V_{in}, and the step is 1⁄4096 V_{in}, where V_{in} is the input voltage.
- V_{in} can be selected from two reference sources
- 16 -word data buffer supported with configurable watermark and multiple operation modes
- DMA support

66.4 Block diagram

The block diagram of the DAC module is as follows:

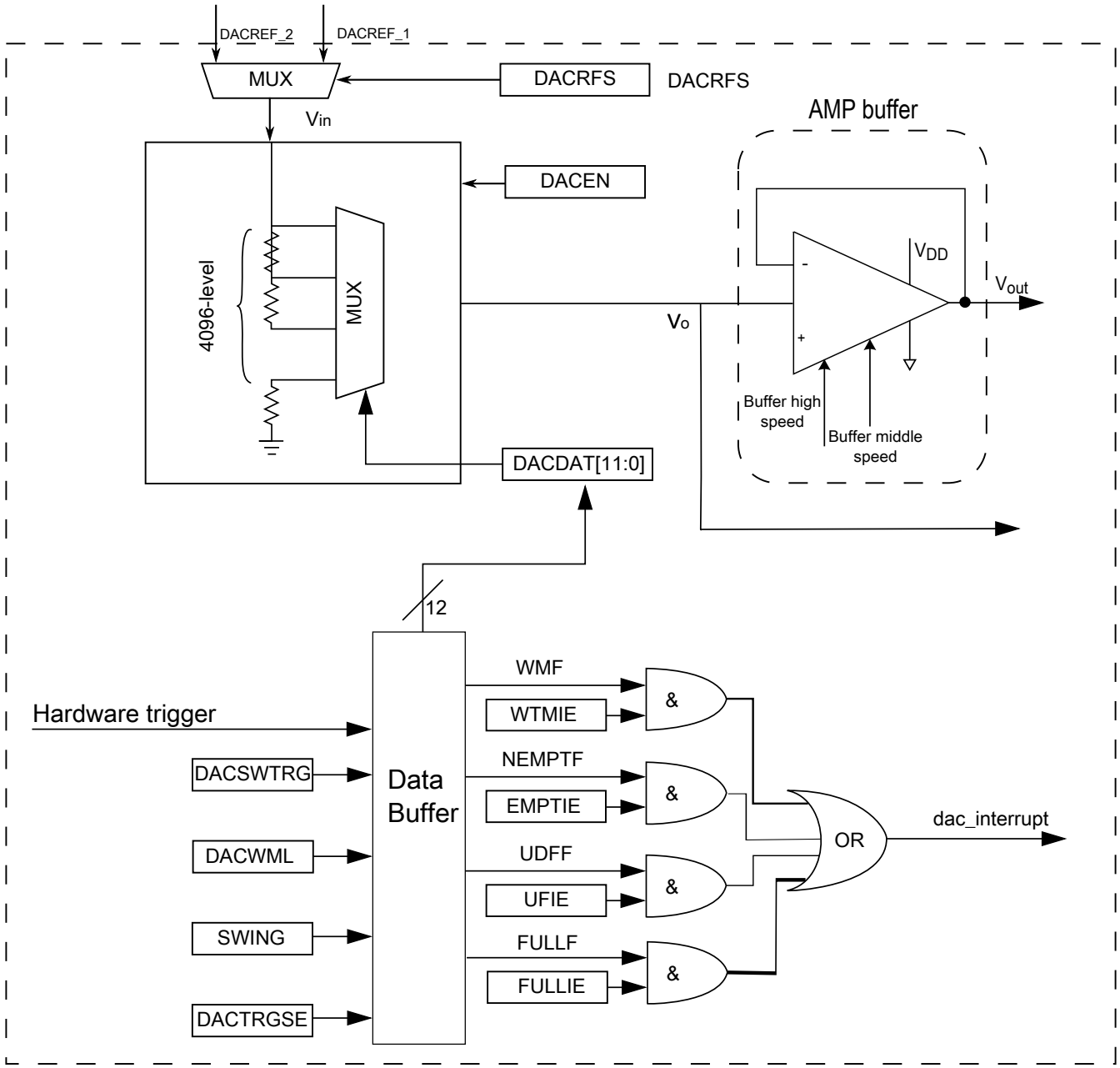


Figure 66-1. DAC block diagram

66.5 Memory map/register definition

The DAC has registers to control analog comparator and programmable voltage divider to perform the digital-to-analog functions.

66.5.1 DAC register descriptions

66.5.1.1 DAC Memory map

DAC0 base address: 4104_4000h

DAC1 base address: 4104_5000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version Identifier Register (VERID)	32	RO	0100_0000h
4h	Parameter Register (PARAM)	32	RO	0000_0003h
8h	DAC Data Register (DATA)	32	WO	0000_0000h
Ch	DAC Status and Control Register (CR)	32	RW	0000_0002h
10h	DAC FIFO Pointer Register (PTR)	32	RO	0000_0000h
14h	DAC Status and Control Register 2 (CR2)	32	RW	0000_0000h
18h	Internal Current Reference Trim Register (ITRM)	32	RW	0000_0004h

66.5.1.2 Version Identifier Register (VERID)

66.5.1.2.1 Offset

Register	Offset
VERID	0h

66.5.1.2.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

66.5.1.2.3 Fields

Field	Function
31-24 MAJOR	Major version number This read-only field returns the major version number for the module specification.
23-16 MINOR	Minor version number This read-only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Identification Number This read-only field returns the feature set number. 0000000000000000b - Standard feature set 0000000000000001b - C40 feature set 0000000000000010b - 5V DAC feature set 0000000000000100b - ADC BIST feature set

66.5.1.3 Parameter Register (PARAM)

66.5.1.3.1 Offset

Register	Offset
PARAM	4h

66.5.1.3.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														FIFOSZ	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

66.5.1.3.3 Fields

Field	Function
31-3	Reserved

Table continues on the next page...

Field	Function
—	
2-0 FIFOSZ	FIFO size The number of words in the DAC FIFO is 2^(FIFOSZ+1). 000b - FIFO depth is 2 001b - FIFO depth is 4 010b - FIFO depth is 8 011b - FIFO depth is 16 100b - FIFO depth is 32 101b - FIFO depth is 64 110b - FIFO depth is 128 111b - FIFO depth is 256

66.5.1.4 DAC Data Register (DATA)

66.5.1.4.1 Offset

Register	Offset
DATA	8h

66.5.1.4.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W					DATA0											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

66.5.1.4.3 Fields

Field	Function
31-12 —	Reserved
11-0 DATA0	FIFO DATA0

66.5.1.5 DAC Status and Control Register (CR)

66.5.1.5.1 Offset

Register	Offset
CR	Ch

66.5.1.5.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WML								DMAEN	0	0	0		UVIE	SWMD	FIFOEN
W										SWRST	FIFORST					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DACEN	DACRFS	TRGSEL	0	0	WTMIE	EMPTIE	FULLIE	0			OVFF	UDFF	WMF	NEMPTF	FULLF
W				SWTRG												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

66.5.1.5.3 Fields

Field	Function
31-24 WML	Watermark Level Select MAX is the FIFO size. **NOTE:** If the FIFO depth is 2, then all settings are treated the same. That is, if the FIFO is not full, the watermark status bit and NEMPTF will be set. One of the status bits maybe needed for DMA or IRQ request.
23 DMAEN	DMA Enable Select 0b - DMA is disabled. 1b - DMA is enabled. When DMA is enabled, the DMA request will be generated by original interrupts. The interrupts will not be presented on this module at the same time.
22 SWRST	Software reset Resets all internal logic and registers, in case of any failure during the Low-power mode. This field is written to only to trigger reset, self clear, or read 0.

Table continues on the next page...

Field	Function
21 FIFORST	FIFO Reset Resets the FIFO pointers. Reading this field will always return zero. FIFO pointers should only be reset when the DAC is disabled. This field can be used to configure both pointers to the same address to reset the FIFO as empty. This field is written to only to trigger reset, self clear, or read zero. 0b - No effect 1b - FIFO reset
20-19 —	Reserved
18 UVIE	Underflow and overflow interrupt enable 0b - Underflow and overflow interrupt is disabled. 1b - Underflow and overflow interrupt is enabled.
17 SWMD	DAC FIFO Mode Select In Swing back mode, the FIFO must be set up such that the FIFO is full. In Swing back mode, a trigger changes the read pointer to make it swing between the FIFO Full and Nearly Empty state. That is, the trigger increases the read pointer till FIFO is nearly empty and decreases the read pointer till the FIFO is full, and so on. 0b - Normal mode 1b - Swing back mode
16 FIFOEN	FIFO Enable 0b - FIFO is disabled and only one level buffer is enabled. Any data written from this buffer goes to conversion. 1b - FIFO is enabled. Data will first read from FIFO to buffer then go to conversion.
15 DACEN	DAC Enable Starts the Programmable Reference Generator operation. 0b - The DAC system is disabled. 1b - The DAC system is enabled.
14 DACRFS	DAC Reference Select 0b - The DAC selects DACREF_1 as the reference voltage. 1b - The DAC selects DACREF_2 as the reference voltage.
13 TRGSEL	DAC Trigger Select 0b - The DAC hardware trigger is selected. 1b - The DAC software trigger is selected.
12 SWTRG	DAC Software Trigger Active high. This is a write-only field, which always reads 0. If DAC software trigger is selected and FIFO is enabled, writing 1 to this field will advance the FIFO read pointer once. **NOTE:** It is not suggested to adopt hardware trigger and software trigger in mixed use, during continuous conversions. 0b - The DAC soft trigger is not valid. 1b - The DAC soft trigger is valid.
11 —	Reserved
10 WTMIE	Watermark Interrupt Enable 0b - Watermark interrupt is disabled. 1b - Watermark interrupt is enabled.
9 EMPTIE	Nearly Empty Interrupt Enable 0b - FIFO Nearly Empty interrupt is disabled. 1b - FIFO Nearly Empty interrupt is enabled.
8	Full Interrupt Enable

Table continues on the next page...

Field	Function
FULLIE	0b - FIFO Full interrupt is disabled. 1b - FIFO Full interrupt is enabled.
7-5 —	Reserved
4 OVFF	Overflow Flag This is the FIFO overflow status flag. This flag indicates that more data has been written into FIFO than it can hold. This field asserts regardless of the value of UVIE. However, an interrupt is issued to the host only if UVIE is set. This flag is cleared by writing a 1 to it. 0b - No overflow has occurred since the last time the flag was cleared. 1b - At least one FIFO overflow has occurred since the last time the flag was cleared.
3 UDFF	Underflow Flag This is the FIFO underflow status flag. If this field is set, it means that there is a new trigger after NEMPTF is set. This field asserts regardless of the value of UVIE. However, an interrupt is issued to the host only if UVIE is set. This flag is cleared by writing a 1 to it. 0b - No underflow has occurred since the last time the flag was cleared. 1b - At least one trigger underflow has occurred since the last time the flag was cleared.
2 WMF	FIFO Watermark Status Flag This field is set if the remaining FIFO data is less than or equal to the watermark setting. It is cleared automatically by writing data into FIFO by DMA or CPU. Write to this bit is ignored in FIFO mode. 0b - The DAC buffer read pointer has not reached the watermark level. 1b - The DAC buffer read pointer has reached the watermark level.
1 NEMPTF	Nearly Empty Flag This is the FIFO nearly empty flag. It is set when only one data remains in FIFO. 0b - More than one data is available in the FIFO. 1b - One data is available in the FIFO.
0 FULLF	Full Flag This is the FIFO Full status bit. This means that the FIFO read pointer equals the write pointer, because of the write pointer increase. This field is cleared if any DAC trigger is making the DAC read pointer increase. Any write to this field is ignored in FIFO mode. 0b - FIFO is not full. 1b - FIFO is full.

66.5.1.6 DAC FIFO Pointer Register (PTR)

66.5.1.6.1 Offset

Register	Offset
PTR	10h

66.5.1.6.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								DACRFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DACWFP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

66.5.1.6.3 Fields

Field	Function
31-24 —	Reserved
23-16 DACRFP	DACRFP This is the FIFO read pointer. **NOTE:** The read pointer is later than the trigger for one cycle. After the conversion finishes and the next trigger comes, then the read pointer updates in the hardware trigger mode.
15-8 —	Reserved
7-0 DACWFP	DACWFP This is the FIFO write pointer. It's value is initially set to equal the read pointer value automatically, and the FIFO status is empty. Both write pointer and read pointer in FIFO mode do not change by IPS read access.

66.5.1.7 DAC Status and Control Register 2 (CR2)

66.5.1.7.1 Offset

Register	Offset
CR2	14h

66.5.1.7.2 Function

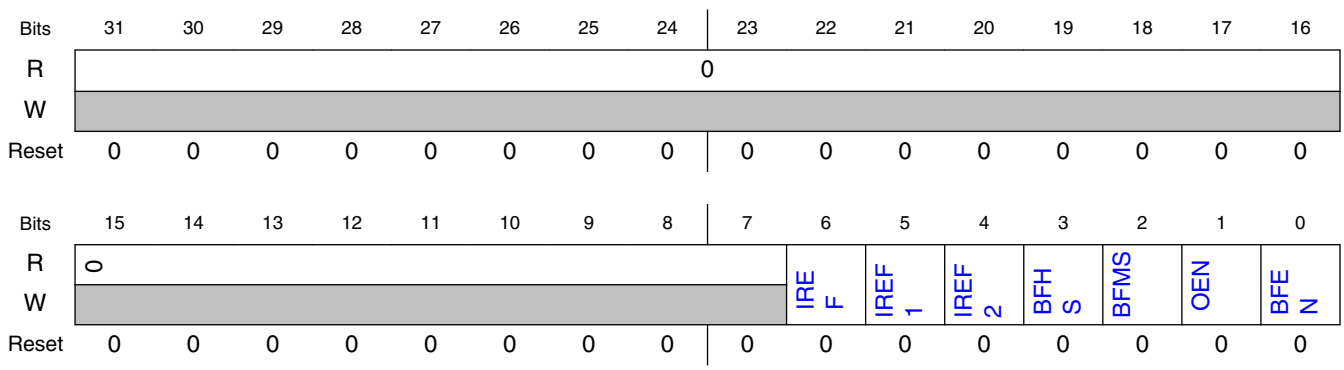
NOTE

For CR2[6:4] (IREF, IREF1 and IREF2), it is suggested to enable one of the bits to reduce power. If more than one bit is selected, the power consumption will increase but the conversion rate will also speed up.

NOTE

IREF works in all modes, but the operation mode of IREF1 or IREF2 depends on PMC.

66.5.1.7.3 Diagram



66.5.1.7.4 Fields

Field	Function
31-7 —	Reserved
6 IREF	Internal Current Reference Select 0b - Internal Current Reference not selected 1b - Internal Current Reference selected
5 IREF1	Internal ZTC (Zero Temperature Coefficient) Current Reference Select 0b - Internal ZTC Current Reference not selected 1b - Internal ZTC Current Reference selected
4 IREF2	Internal PTAT (Proportional To Absolute Temperature) Current Reference Select 0b - Internal PTAT Current Reference not selected 1b - Internal PTAT Current Reference selected
3 BFHS	Buffer High Speed Select 0b - Buffer high speed not selected 1b - Buffer high speed selected
2 BFMS	Buffer Middle Speed Select 0b - Buffer middle speed not selected 1b - Buffer middle speed selected

Table continues on the next page...

Field	Function
1 OEN	Optional Enable 0b - Output buffer is not bypassed 1b - Output buffer is bypassed
0 BFEN	Buffer Enable 0b - Opamp is not used as buffer 1b - Opamp is used as buffer

66.5.1.8 Internal Current Reference Trim Register (ITRM)

66.5.1.8.1 Offset

Register	Offset
ITRM	18h

66.5.1.8.2 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													TRIM		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

66.5.1.8.3 Fields

Field	Function
31-3 —	Reserved
2-0 TRIM	Internal Current Trim Register

66.6 Functional description

The 12-bit DAC module can select one of the two reference inputs—DACREF_1 and DACREF_2 as the DAC reference voltage, V_{in} by STATCTRL [DACRFS]. See the chip-specific DAC information to determine the source options for DACREF_1 and DACREF_2.

When the DAC is enabled, it converts the data in DACDAT0[11:0] or the data from the DAC data buffer to a stepped analog output voltage. The output voltage range is from V_{in} to $V_{in}/4096$, and the step is $V_{in}/4096$.

66.6.1 FIFO mode

In FIFO mode, the buffer is organized as a FIFO. For a valid write to any DATA register, the data is put into the FIFO, and the write pointer is automatically incremented. The module is connected internally to a 32-bit interface. For any 16-bit or 8-bit FIFO access, address bit[1] needs to be 0; otherwise, the write is ignored. A successful 32-bit FIFO write increases the write pointer by one.

For an 8-bit write, the LSB determines which byte lane is written to. Only if both the byte lanes are written to, will the write pointer increase. The user needs to ensure that an 8-bit access is done in pair, and both the upper and lower bytes are written. There is no priority for which byte is written to first.

FIFO mode needs to be enabled with DACCR[FIFOEN].

66.6.2 DAC data buffer operation

When the DAC FIFO is disabled, and the one entry buffer is enabled, the DAC converts the data in the buffer to analog output voltage. Any write to the DATA register will replace the data in the buffer and push data to analog conversion without trigger support.

When the IP interface bus access (IPS) is in write state, the write speed of DATA[DATA0] is high. However, the analog part of the DAC cannot convert data at the same speed. Therefore, the multi write speed limit needs to be defined in the firmware.

66.6.3 DAC interrupts

There are several interrupts and associated flags that can be configured for the DAC buffer.

FIFO request flag

The FIFO request flag is set based on the number of entries in the FIFO and the FIFO watermark configuration. This flag is set when the number of entries in any of the enabled FIFOs is less than or equal to the FIFO watermark configuration and is automatically cleared when the number of entries in FIFO is greater than the FIFO watermark configuration. The FIFO request flag can generate an interrupt or a DMA request.

FIFO warning flag

The FIFO warning flag is set based on the number of entries in the FIFO. The empty flag and full flags are both warning flags. The FIFO warning flag can generate an interrupt (for empty flag only).

FIFO error flag

The FIFO error flag is set when overflow and underflow happens.

66.6.4 DMA operation

When DMA is enabled, DMA requests are generated instead of interrupt requests. The DMA Done signal clears the DMA request.

The status register flags are still set and are cleared automatically when the DMA completes.

66.6.5 Resets

During reset, the DAC is configured in the default mode and is disabled.

66.6.6 Low-Power mode operation

The following table shows the wait mode and the stop mode operation of the DAC module.

Table 66-3. Modes of operation

Modes of operation	Description
Wait mode	The DAC will operate normally, if enabled.
Stop mode	If enabled, the DAC module continues to operate normally, with the hardware trigger initiating the conversion.

NOTE

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.

66.6.7 Background Debug Mode Operation

When the microcontroller is in active background debug mode, the CMP continues to operate normally.

Chapter 67
Analog Comparator (ACMP)

67.1 Chip-specific ACMP information

Table 67-1. Reference links to related information

Topic	Related module	Reference
Full description	ACMP	ACMP
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

67.1.1 Comparator (CMP)

The Comparator module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage (rail-to-rail operation). The following table shows the configuration of the CMP. The CMP digital wrapper is in the always-on DGO power domain.

Table 67-2. CMP configuration

Parameter	Description
Name	Comparator (CMP)
Instances	2
Configurable features	NA
Interface speed	NA
External I/O pins	See the IOMUXC sheet for pin details

67.1.2 CMP trigger options

On i.MX 7ULP, CMP trigger sources get routed through TRIGMUX module. To know about the input and output trigger options for CMP0 and CMP1, see Table 38-2

67.1.3 CMP voltage reference options

The table below shows supply voltage reference options that can be selected through C1[VRSEL] field in CMP0 and CMP1.

CMP instance	Pin name	VRSEL bit value	Voltage reference source
CMP0	vref1_ana18	VRSEL = 1b	PMC's 1.8V supply VDD_PMC18
	vref0_ana18	VRSEL = 0b	PMC's 1.3V reference
	vrefl_1p8		Global Ground
CMP1	vref1_ana18	VRSEL = 1b	PMC's 1.8V supply VDD_PMC18
	vref0_ana18	VRSEL = 0b	PMC's 1.3V reference
	vrefl_1p8		Global Ground

NOTE

The comparator internal reference needs to be enabled by setting PMC0_ACTRL[BUFEN] bitfield. Before use, in VLPS or VLLS mode, the reference voltage needs to be enabled in PMC 0 by setting VLPS[FBGHP] bitfield for VLPS mode and VLLS[FBGHP] bitfield for VLLS mode. See register description in the PMC chapter.

67.1.4 CMP input connections

To know about the input connections for CMP0 and CMP1, see the "CMP Sources" tab of the IOMUXC spreadsheet attached with this reference manual.

67.2 Introduction

The comparator (CMP) module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage, known as rail-to-rail operation.

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal is provided by the 8-bit digital-to-analog converter (DAC). The mux circuit is designed to operate across the full range of the supply voltage.

The DAC is a 256-tap resistor ladder network that provides a selectable voltage reference for applications requiring a voltage reference. The 256-tap resistor ladder network divides the supply reference V_{in} into 256 voltage levels. A 8-bit digital signal input selects the output voltage level, which varies from V_{in} to $V_{in}/256$. V_{in} can be selected from two voltage sources, V_{in1} and V_{in2}.

67.3 Features

The following subsections list the features of the CMP, the DAC, and the ANMUX.

67.3.1 CMP features

The CMP has the following features:

- Operational over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control
- Selectable interrupt on rising-edge, falling-edge, or both rising or falling edges of the comparator output
- Selectable inversion on comparator output
- Capability to produce a wide range of outputs such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally filtered:
    - Filter can be bypassed
    - Can be clocked via external SAMPLE signal or scaled bus clock
- External hysteresis can be used at the same time that the output filter is used for internal functions
- Two software selectable performance levels:

    - Shorter propagation delay at the expense of higher power
    - Low power, with longer propagation delay
- DMA transfer support
    - A comparison event can be selected to trigger a DMA transfer
- Functional in all MCU power modes
- The window and filter functions are not available in STOP modes
- The comparator can be triggered by other peripherals to work for only a small fraction of the time

67.3.2 8-bit DAC key features

The 8-bit DAC has the following features:

- 8-bit resolution
- Selectable supply reference source
- Configurable low power mode or high speed mode
- Power Down mode to conserve power when not in use
- Option to route the output to internal comparator input

67.3.3 ANMUX key features

The ANMUX has the following features:

- Two 8-to-1 channel MUXes
- Operational over the entire supply range

67.4 CMP, DAC, and ANMUX diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

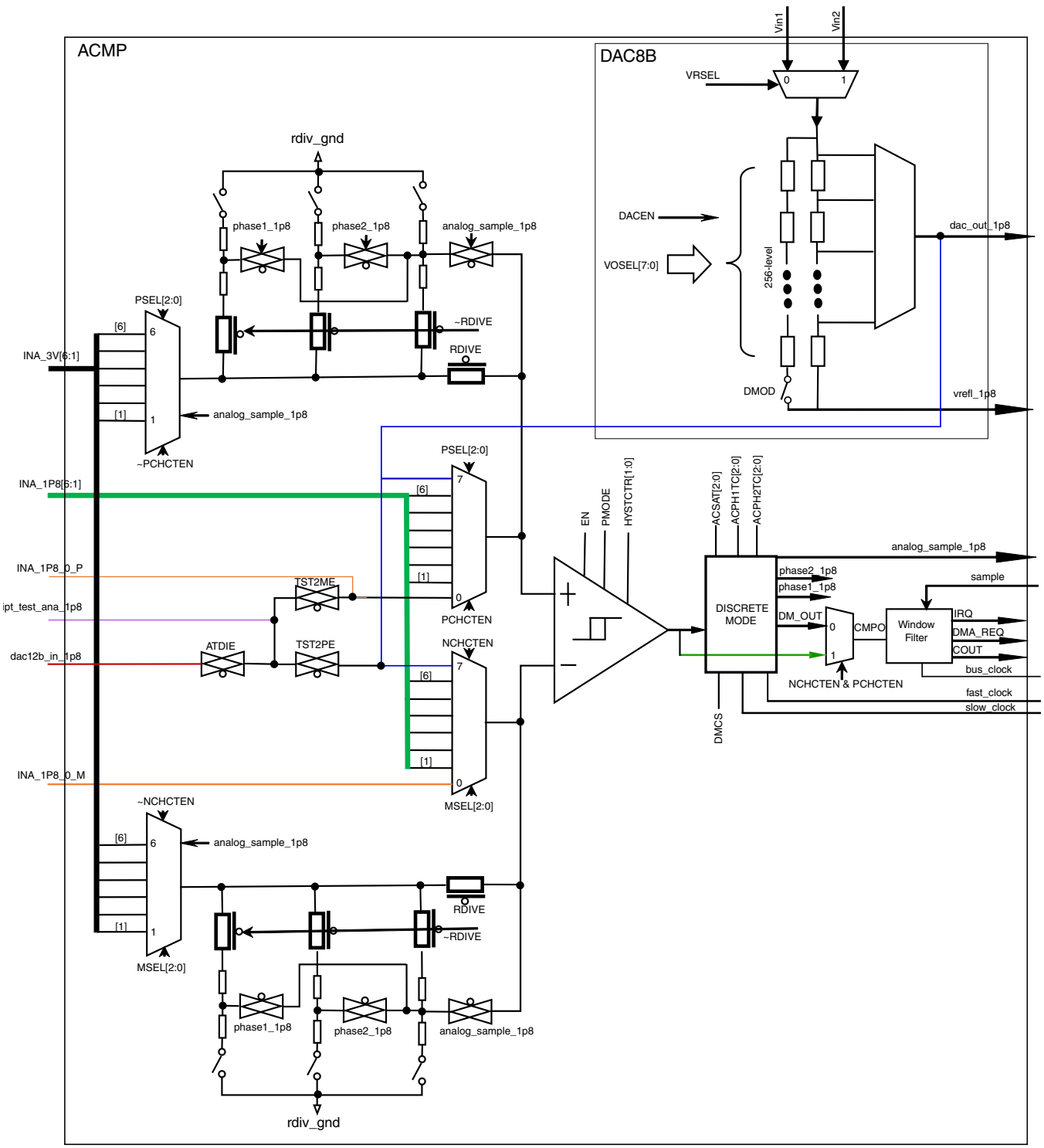


Figure 67-1. CMP high level diagram

67.5 CMP block diagram

The following figure shows the block diagram for the CMP module.

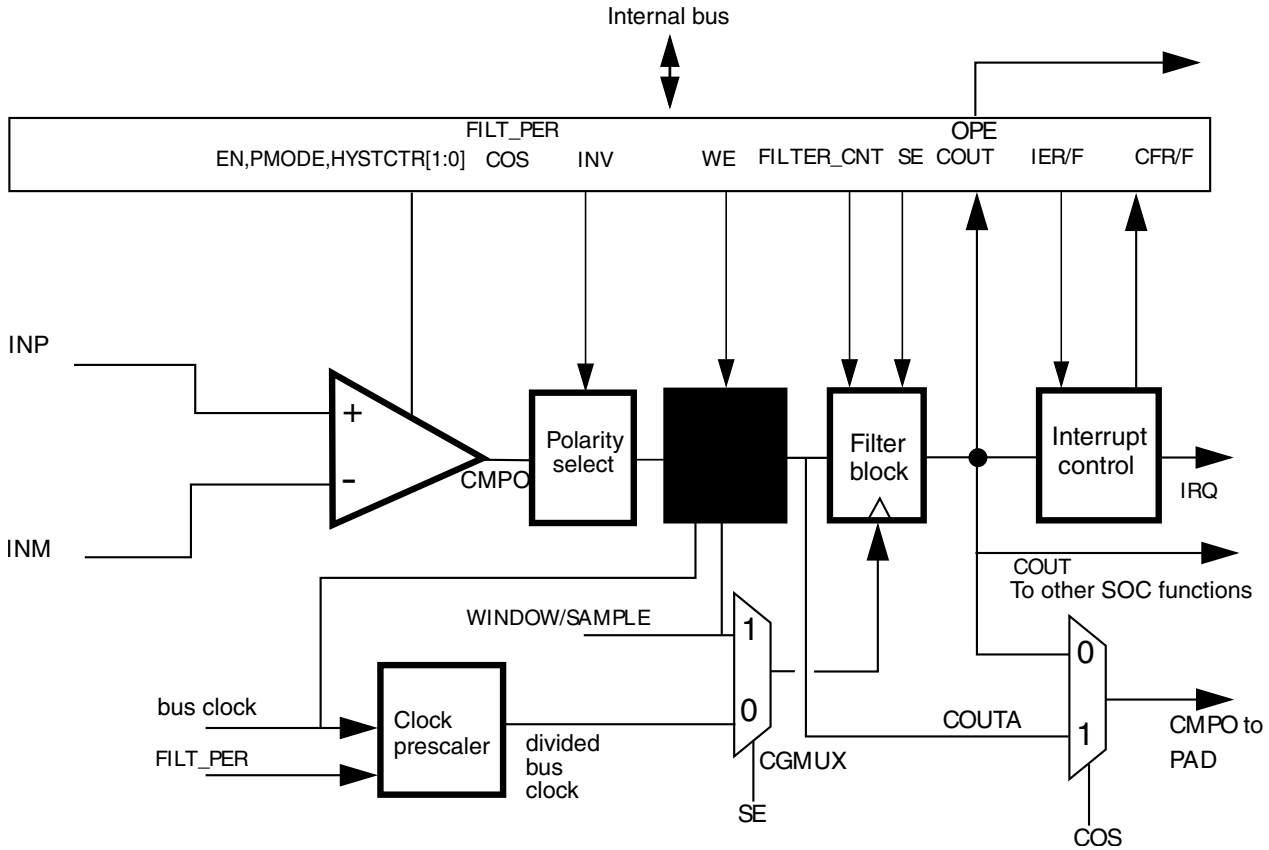


Figure 67-2. Comparator module block diagram

In the CMP block diagram:

- The Window Control block is bypassed when C0[WE] = 0.
- If C0[WE] = 1, the comparator output is sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter block is bypassed when not in use.

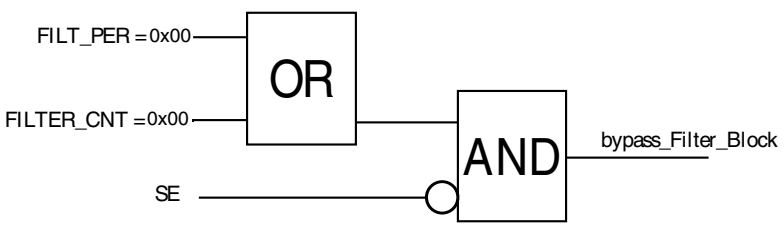


Figure 67-3. Filter block bypass logic

- The Filter block acts as a simple sampler if the filter is bypassed and C0[FILTER_CNT] is set to 0x01.
- The Filter block filters based on multiple samples when the filter is bypassed and C0[FILTER_CNT] is set greater than 0x01.

  - If C0[SE] = 1, the external SAMPLE input is used as the sampling clock.
  - IF C0[SE] = 0, the divided bus clock is used as the sampling clock.
- If enabled, the Filter block will incur up to one bus clock additional latency penalty on COUT due to the fact that COUT, which crosses clock domain boundaries, must be resynchronized to the bus clock.
- C0[WE] and C0[SE] are mutually exclusive.
- If enabled, the filter clock and the sample period must be at least 4 times slower than the system clock to the comparator.

67.6 CMP pin descriptions

This section provides the comparator pin descriptions. The external 3V domain inputs INA_3V[6:1] and 1.8V domain INA_1P8[6:1], INA_1P8_0_P, INA_1P8_0_M are muxed by CMP_C1[PSEL] and CMP_C1[MSEL] before going to the positive and negative ports of the comparator respectively.

Table 67-3. CMP signal descriptions

Signal	Description	I/O
INA_3V[6:1]	The 3V domain analog inputs[6:1]	I
INA_1P8[6:1]	Analog 1.8V input channels [6:1].	I
INA_1P8_0_P	Analog 1.8V input channel 0 positive.	I
INA_1P8_0_m	Analog 1.8V input channel 0 minus.	I

NOTE

It is not recommended to compare 3V domain channel with 1.8V domain channel. If the 3V domain channel's voltage is smaller than 1.8V, in this case, it can be compared with 1.8V domain channel by setting C3[RDIVE] to 0.

67.6.1 External pins

The CMP has two analog inputs: INP and INM. Each of these pins can accept an input voltage that varies across the full operating range of the MCU. If the module is not enabled, each pin can be used as a digital input or output. Consult the specific MCU documentation to determine what functions are shared with these analog inputs.

The user can select either filtered or unfiltered comparator outputs for use on an external I/O pad.

67.7 CMP functional modes

There are three main sub-blocks to the CMP module:

- The comparator itself
- The window function
- The filter function

The filter, CMP_C0[FILTER_CNT], can be clocked from an internal or external clock source. The filter is programmable with respect to the number of samples that must agree before a change in the output is registered. In the simplest case, only one sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CMP_C0[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting CMP_C0[WE]. When set, the comparator output is sampled only when WINDOW=1. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 67-4. Comparator sample/filter controls

Mode #	CMP_C0[EN]	CMP_C0[WE]	CMP_C0[SE]	CMP_C0[FILTER_CNT]	CMP_C0[FPR]	Operation
1	0	X	X	X	X	**Disabled** See the Disabled mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode** See the Continuous mode (#s 2A & 2B).
2B	1	0	0	X	0x00	
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode** See the Sampled, Non-Filtered mode (#s 3A & 3B).
3B	1	0	0	0x01	> 0x00	
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode**

Table continues on the next page...

Table 67-4. Comparator sample/filter controls (continued)

Mode #	CMP_C0[EN]	CMP_C0[WE]	CMP_C0[SE]	CMP_C0[FILTER_CNT]	CMP_C0[FPR]	Operation
4B	1	0	0	> 0x01	> 0x00	See the Sampled, Filtered mode (#s 4A & 4B).
5A	1	1	0	0x00	X	**Windowed mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA. See the Windowed mode (#s 5A & 5B).
5B	1	1	0	X	0x00	
6	1	1	0	0x01	0x01–0xFF	**Windowed/Resampled mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled on an interval determined by CMP_C0[FPR] to generate COUT. See the Windowed/Resampled mode (# 6).
7	1	1	0	> 0x01	0x01–0xFF	**Windowed/Filtered mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT. See the Windowed/Filtered mode (#7).
8	1	1	1	X	X	**Trigger mode** In RUN mode Comparator is configured to the modes same with these Modes #5A, #5B, #6 and #7 as elaborated in above rows depended on the CMP_C0[FILT_CNT] and CMP_C0[FPR] settings. In STOP mode, the comparator and possible the 8bit DAC is triggered work periodically and the active channels follow the round robin cycling scheme. See the Trigger mode.
All other combinations of CMP_C0[EN], CMP_C0[WE], CMP_C0[SE], CMP_C0[FILTER_CNT], and CMP_C0[FPR] are illegal.						

For cases where a comparator is used to drive a fault input, for example, for a motor-control module such as FTM, it must be configured to operate in Continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

Note

Filtering and sampling settings must be changed only after setting CMP_C0[SE]=0, CMP_C0[FPR] =0 and CMP_C0[FILTER_CNT]=0x00. This resets the filter to a known state.

67.7.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

67.7.2 Continuous mode (#s 2A & 2B)

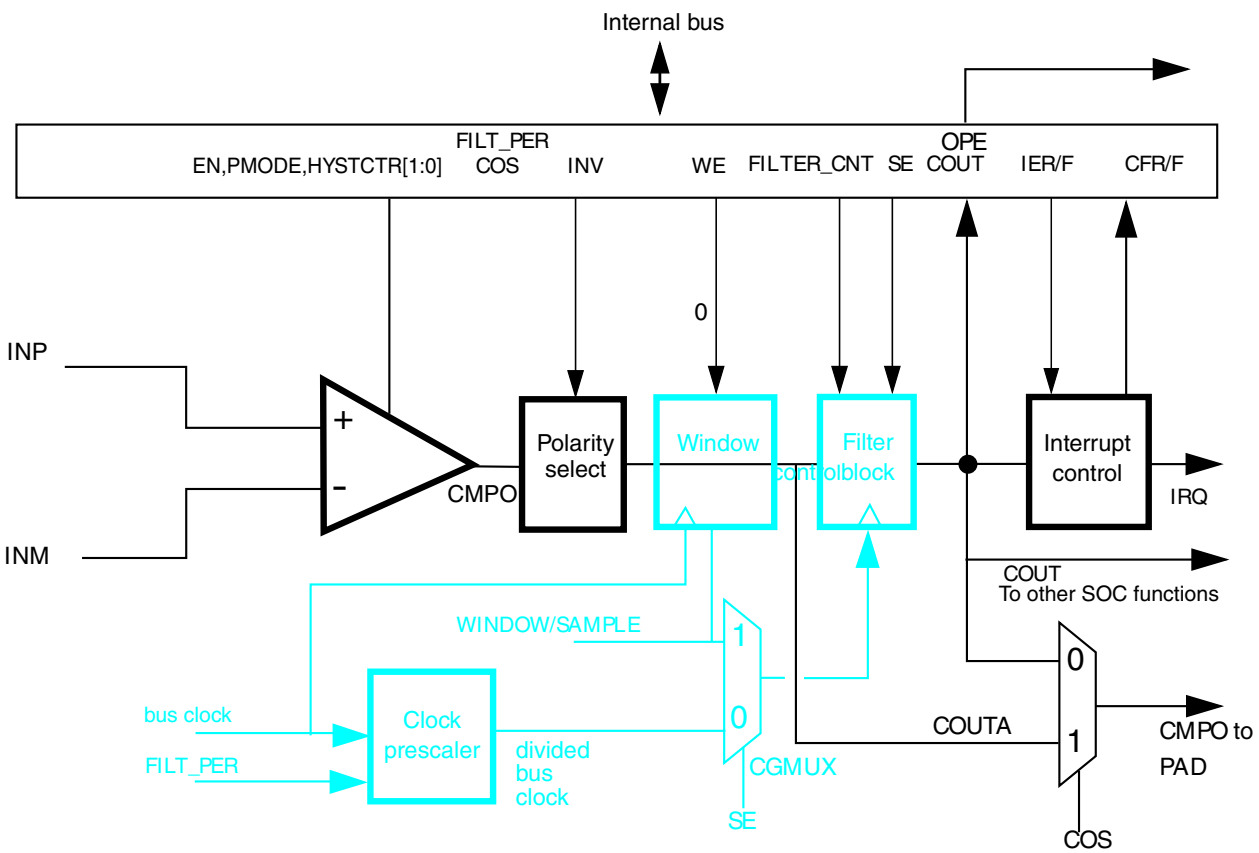


Figure 67-4. Comparator operation in Continuous mode

NOTE

See the chip configuration section for the source of sample/ window input.

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both window control and filter blocks are completely bypassed. CMP_C0[COUT] is updated continuously. The path from comparator input pins to output pin is operating in combinational unclocked mode. COUT and COUTA are identical.

For control configurations that result in disabling the filter block, see Figure 67-3.

67.7.3 Sampled, Non-Filtered mode (#s 3A & 3B)

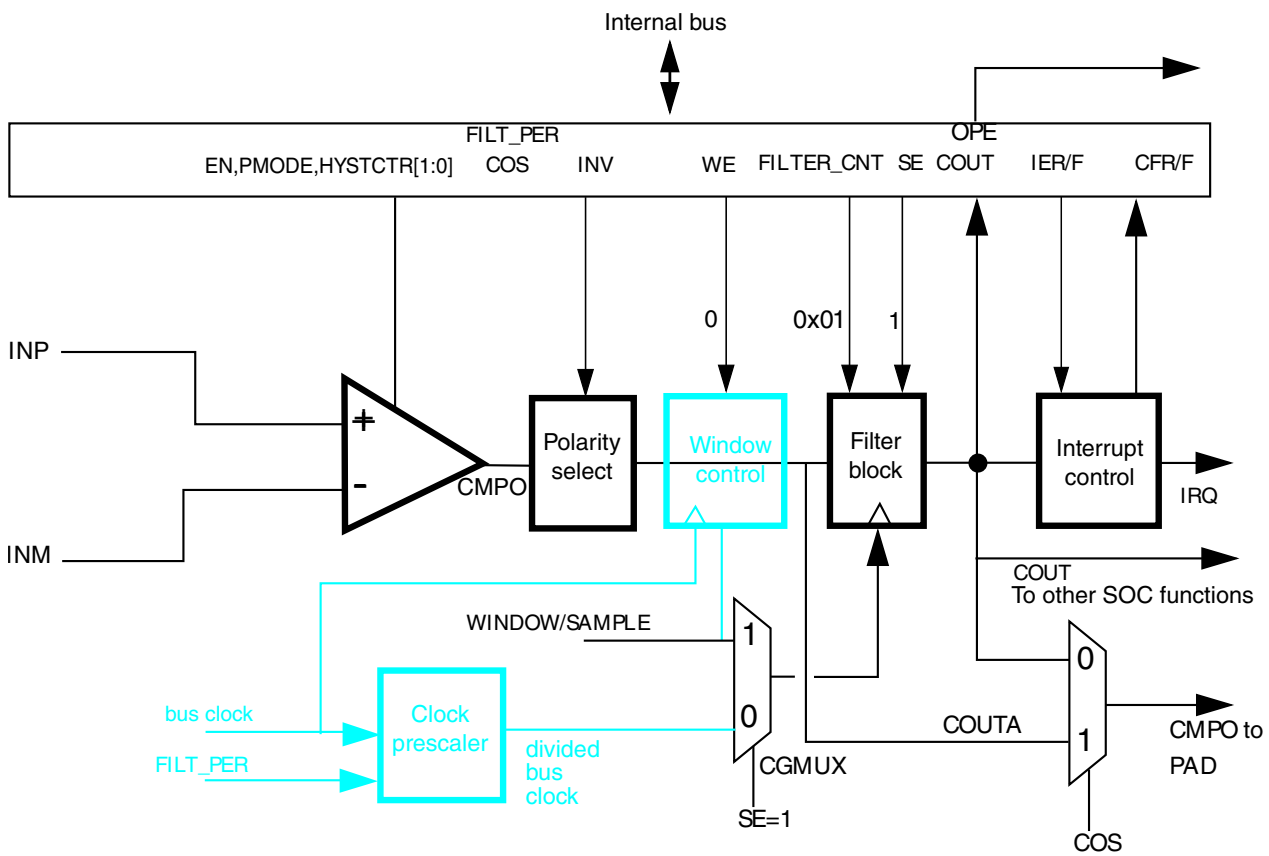


Figure 67-5. Sampled, Non-Filtered (# 3A): sampling point externally driven

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the filter block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

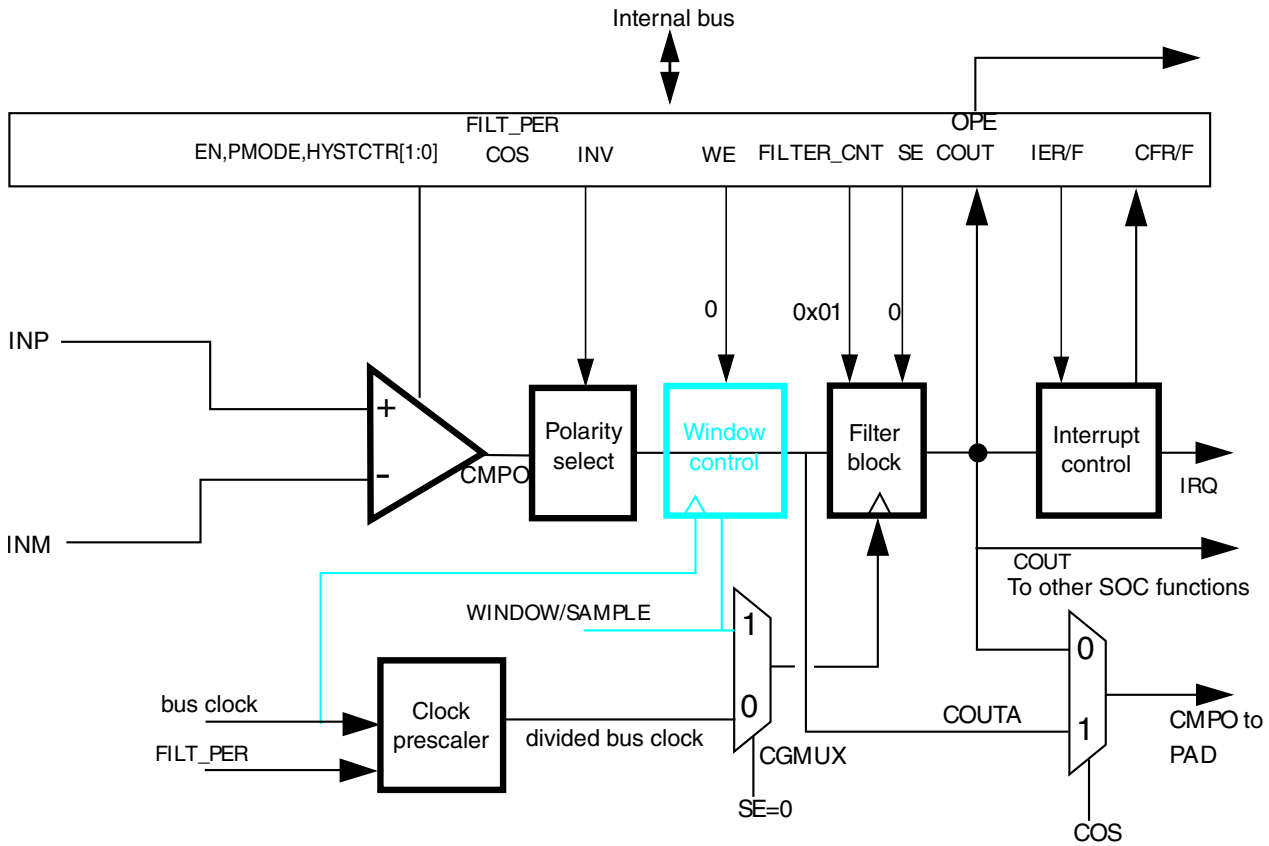


Figure 67-6. Sampled, Non-Filtered (# 3B): sampling interval internally derived

67.7.4 Sampled, Filtered mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that, now, CMP_C0[FILTER_CNT]>1, which activates filter operation.

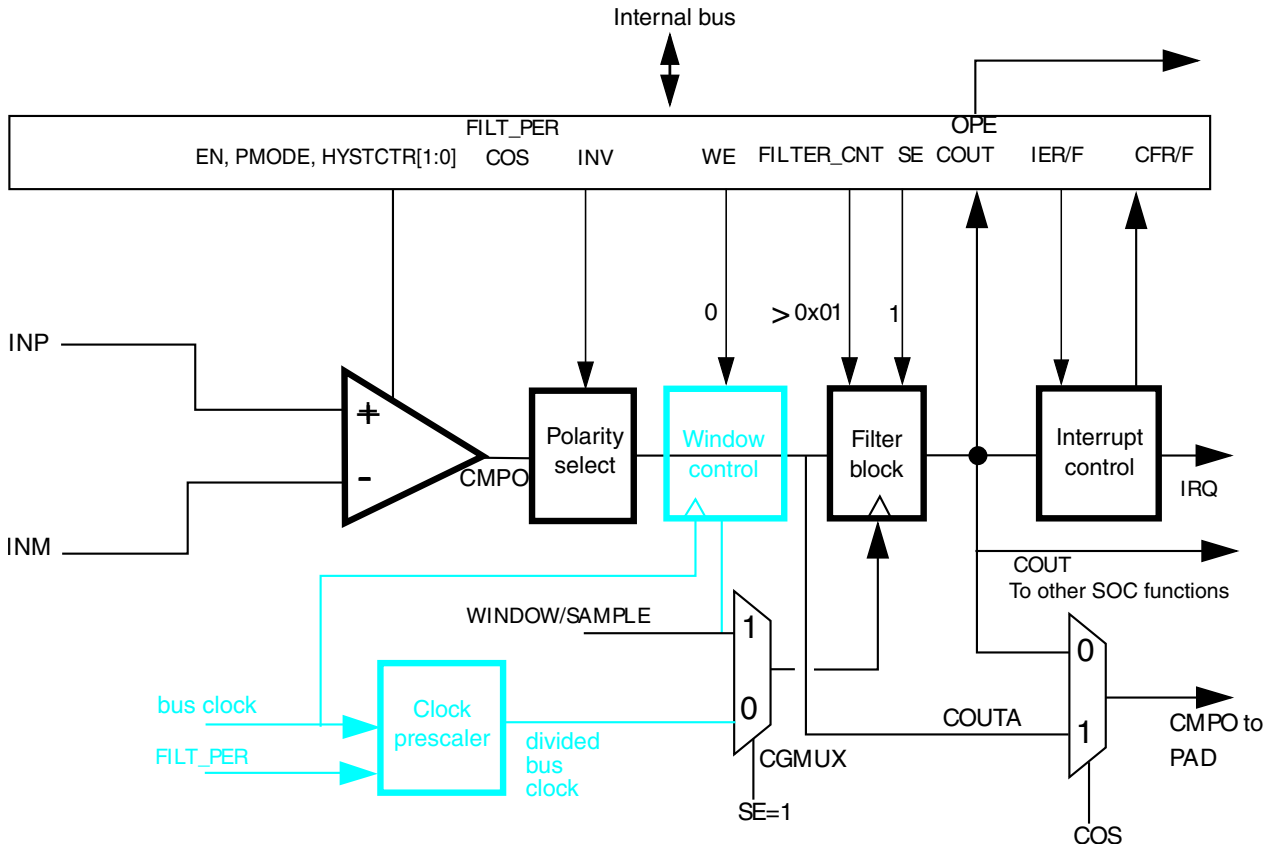


Figure 67-7. Sampled, Filtered (# 4A): sampling point externally driven

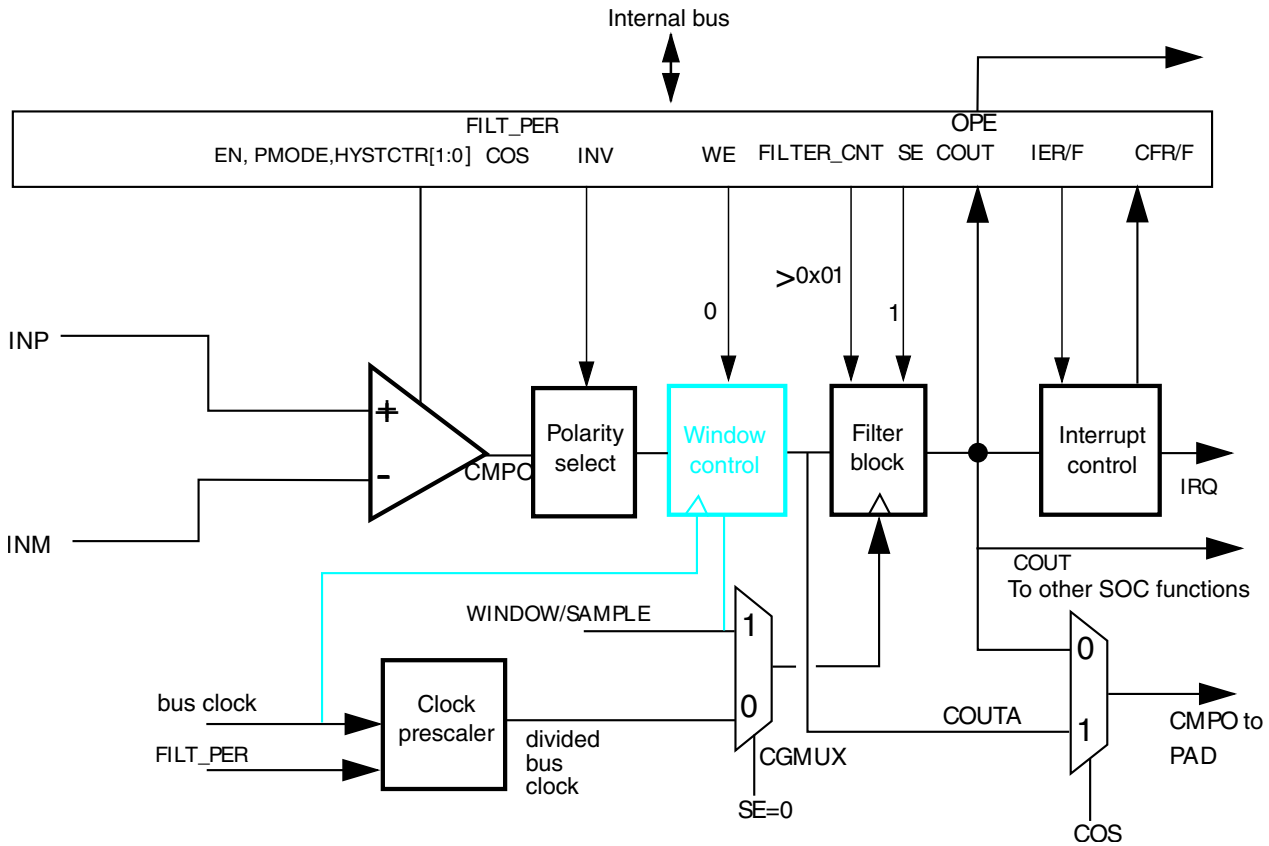


Figure 67-8. Sampled, Filtered (# 4B): sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that now, CMP_C0[FILTER_CNT]>1, which activates filter operation.

67.7.5 Windowed mode (#s 5A & 5B)

The following figure illustrates comparator operation in the Windowed mode, ignoring latency of the analog comparator, polarity select, and window control block. It also assumes that the polarity select is set to non-inverting state.

NOTE

The analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

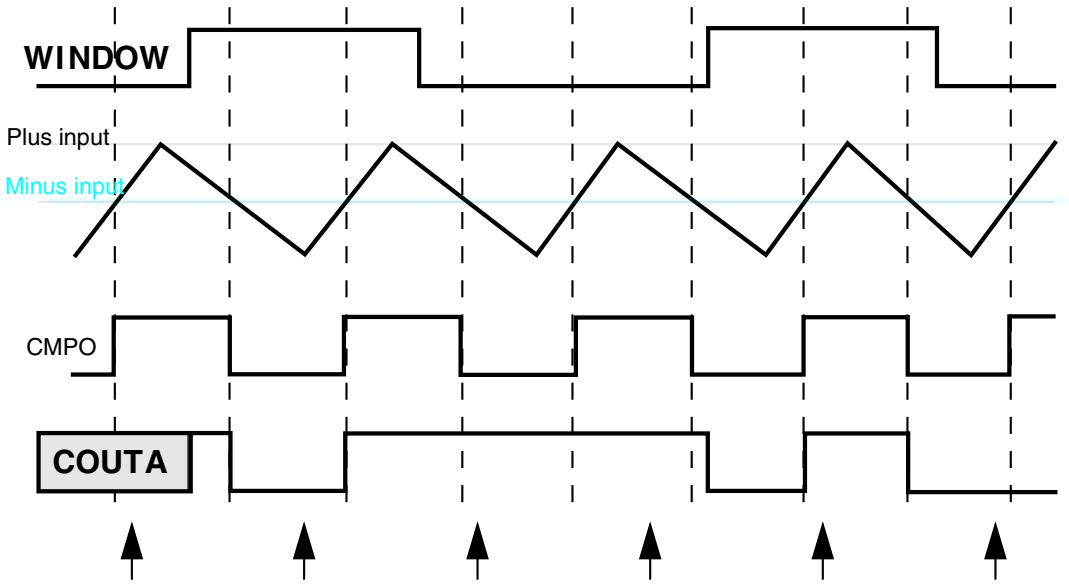


Figure 67-9. Windowed mode timing diagram

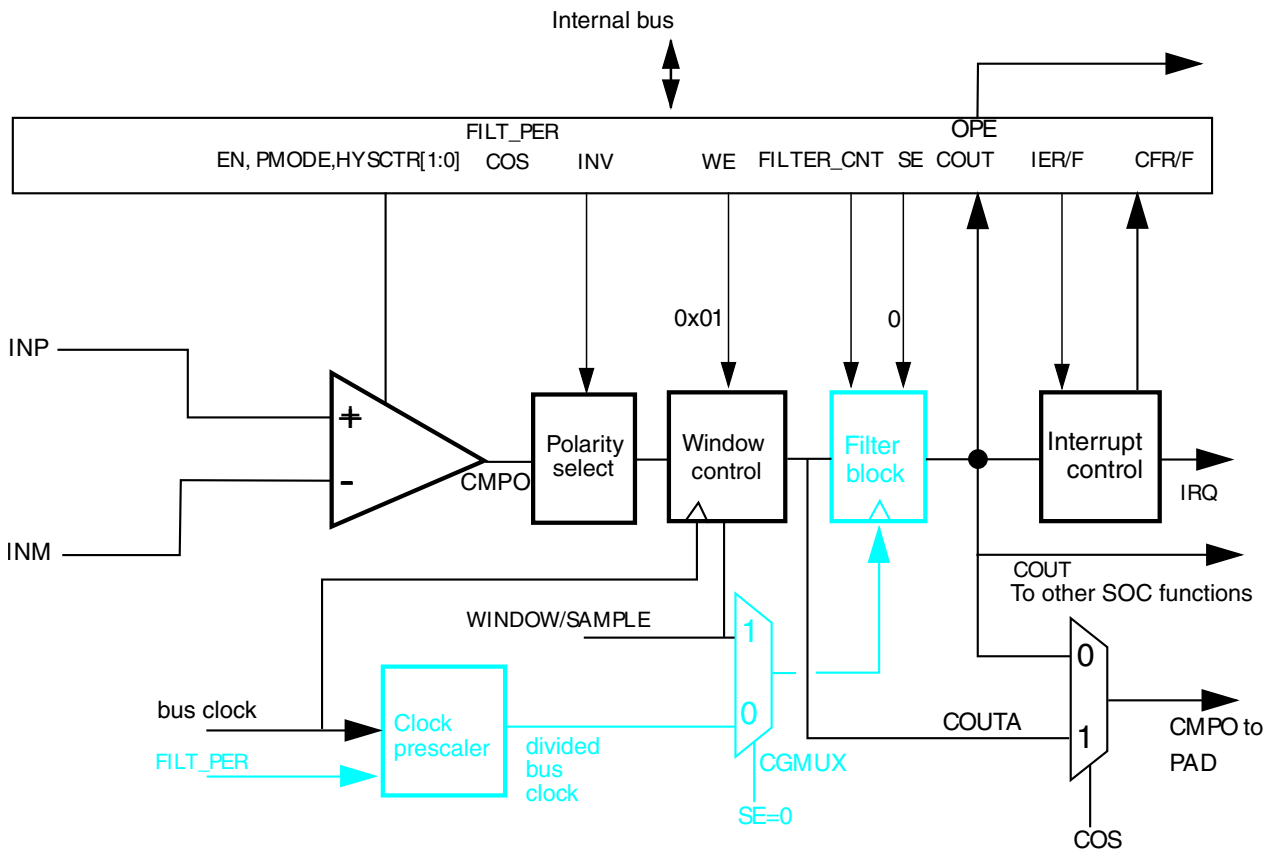


Figure 67-10. Windowed mode

For control configurations which result in disabling the filter block, see Figure 67-3.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

67.7.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in Figure 67-9, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows in the figure. Again, prop delays and latency are ignored for the sake of clarity.

This example was generated solely to demonstrate operation of the comparator in windowed/resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.

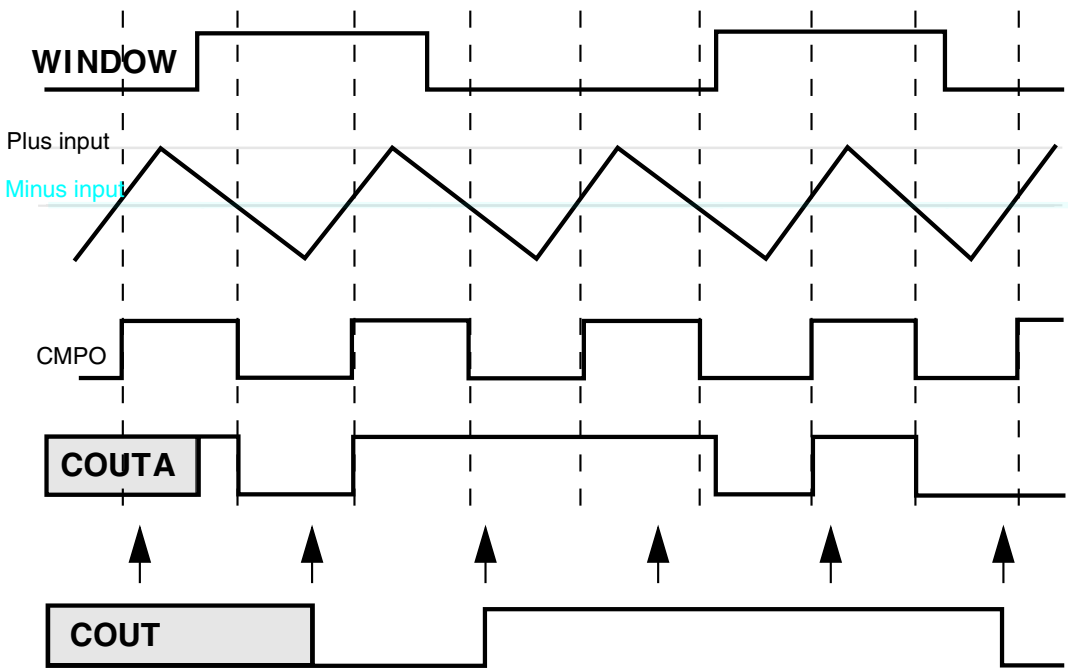


Figure 67-11. Windowed/resampled mode operation

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of C0[FILTER_CNT] must be 1.

67.7.7 Windowed/Filtered mode (#7)

This is the most complex mode of operation for the comparator block, as it uses both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function + ((C0[FILTER_CNT] * C0[FPR]) + 1) * bus clock for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

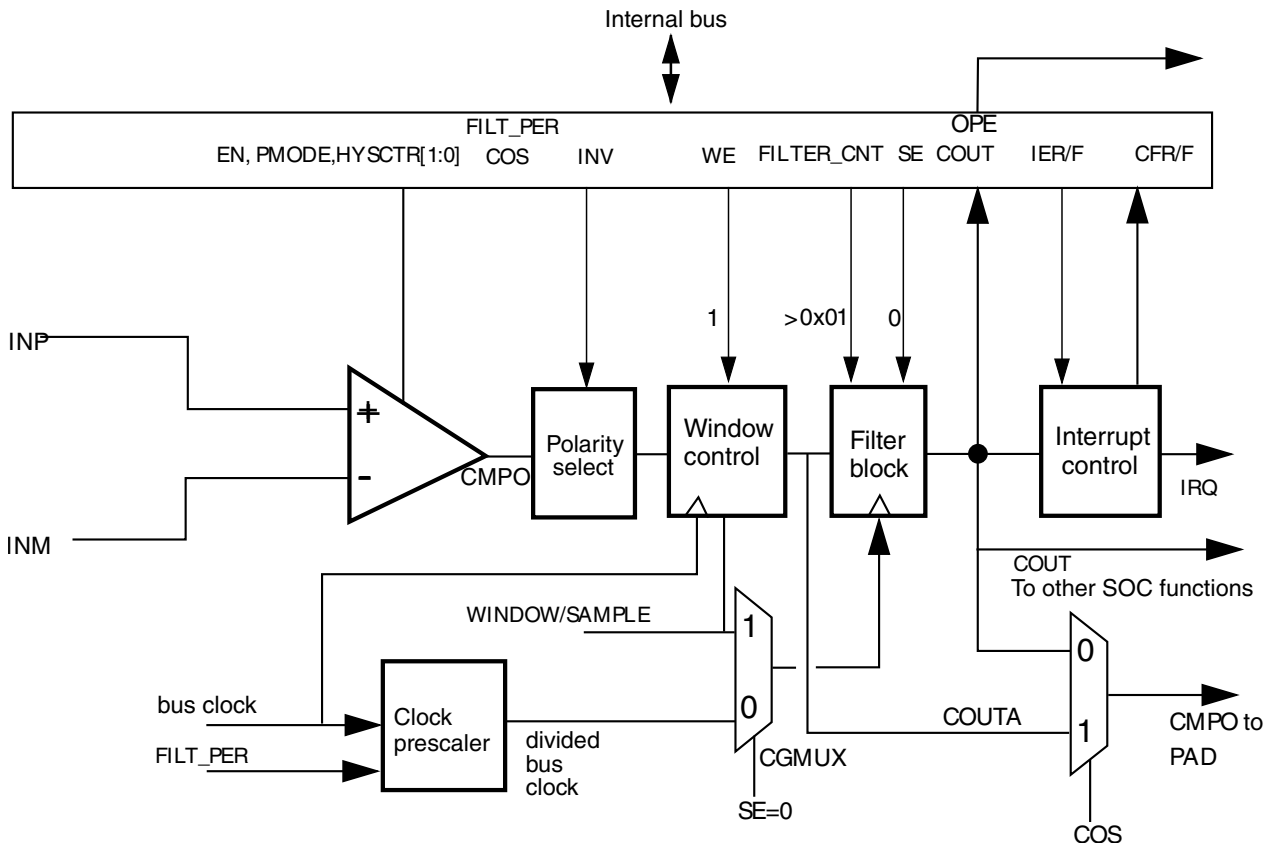


Figure 67-12. Windowed/Filtered mode

67.8 Memory map/register definitions

67.8.1 CMP register descriptions

67.8.1.1 CMP Memory map

CMP0 base address: 4104_2000h

CMP1 base address: 4104_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VERID)	32	RO	0100_0000h
4h	Parameter Register (PARAM)	32	RO	0000_0000h
8h	CMP Control Register 0 (C0)	32	RW	0000_0000h
Ch	CMP Control Register 1 (C1)	32	RW	0000_0000h
10h	CMP Control Register 2 (C2)	32	RW	0000_0000h
14h	CMP Control Register 3 (C3)	32	RW	1100_0000h

67.8.1.2 Version ID Register (VERID)

67.8.1.2.1 Offset

Register	Offset
VERID	0h

67.8.1.2.2 Function

Access:

- Supervisor read only
- User read only

67.8.1.2.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MAJOR								MINOR							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FEATURE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

67.8.1.2.4 Fields

Field	Function
31-24 MAJOR	Major Version Number. This read only field returns the major version number for the module specification.
23-16 MINOR	Minor Version Number. This read only field returns the minor version number for the module specification.
15-0 FEATURE	Feature Specification Number. This read only filed returns the feature set number.

67.8.1.3 Parameter Register (PARAM)

67.8.1.3.1 Offset

Register	Offset
PARAM	4h

67.8.1.3.2 Function

Access:

- Supervisor read only
- User read only

67.8.1.3.3 Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PARAM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PARAM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

67.8.1.3.4 Fields

Field	Function
31-0 PARAM	Parameter Registers. This read only filed returns the feature parameters implemented along with the Version ID register.

67.8.1.4 CMP Control Register 0 (C0)

67.8.1.4.1 Offset

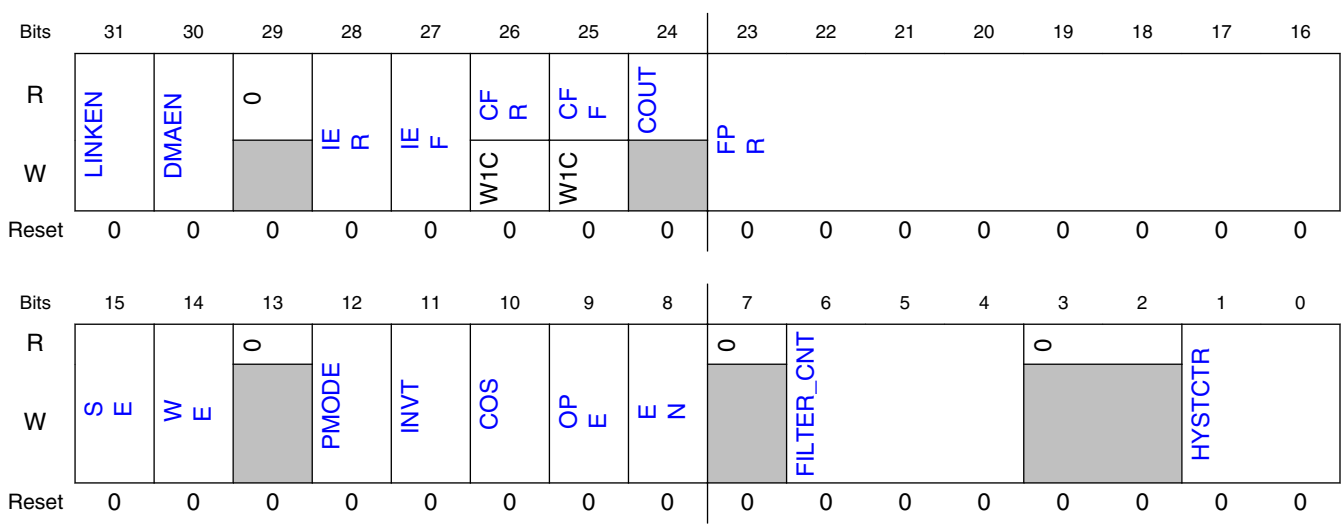
Register	Offset
C0	8h

67.8.1.4.2 Function

Access:

- Supervisor read/write
- User read/write

67.8.1.4.3 Diagram



67.8.1.4.4 Fields

Field	Function
31 LINKEN	CMP to DAC link enable. This bit is used to enable the link from CMP to DAC enables. When this bit is set, the DAC enable/disable is controlled by CMP_C0[EN] bit instead of CMP_C1[DACEN]. 0b - CMP to DAC link is disabled 1b - CMP to DAC link is enabled.
30 DMAEN	DMA Enable Enables the DMA transfer triggered from the CMP module. When this field and the corresponding interrupt enable bit are set, a DMA request is asserted when CFR or CFF is set. 0b - DMA is disabled. 1b - DMA is enabled.
29 —	Reserved
28 IER	Comparator Interrupt Enable Rising Enables the CFR interrupt from the CMP. When this field is set, an interrupt will be asserted when CFR is set. 0b - Interrupt is disabled. 1b - Interrupt is enabled.
27 IEF	Comparator Interrupt Enable Falling Enables the CFF interrupt from the CMP. When this field is set, an interrupt will be asserted when CFF is set. 0b - Interrupt is disabled. 1b - Interrupt is enabled.
26 CFR	Analog Comparator Flag Rising Detects a rising-edge on COUT, when set, during normal operation. CFR is cleared by writing 1 to it. During Stop modes, CFR is level sensitive 0b - A rising edge has not been detected on COUT. 1b - A rising edge on COUT has occurred.
25 CFF	Analog Comparator Flag Falling Detects a falling-edge on COUT, when set, during normal operation. CFF is cleared by writing 1 to it. During Stop modes, CFF is level senstive . 0b - A falling edge has not been detected on COUT. 1b - A falling edge on COUT has occurred.
24 COUT	Analog Comparator Output Returns the current value of the Analog Comparator output, when read. The field is reset to 0 and will read as C0[INVT] when the Analog Comparator module is disabled, that is, when C0[EN] = 0. Writes to this field are ignored.
23-16 FPR	Filter Sample Period Specifies the sampling period, in bus clock cycles, of the comparator output filter, when C1[SE] = 0. Setting FPR to 0x0 disables the filter. Filter programming and latency details are provided in the CMP functional description. This field has no effect when C0[SE]= 1. In that case, the external SAMPLE signal is used to determine the sampling period.
15 SE	Sample Enable If a write to this register attempts to set both SE and WE, then only Sampling mode is taking effect in MCU Run mode and the Round Robin Cycling(Trigger Mode) is enabled in MCU STOP mode. 0b - Sampling mode is not selected. 1b - Sampling mode is selected.

Table continues on the next page...

Field	Function
14 WE	Windowing Enable If a write to this register attempts to set both SE and WE, then only Sampling mode is taking effect in MCU Run mode and the Round Robin Cycling(Trigger Mode) is enabled in MCU STOP mode. 0b - Windowing mode is not selected. 1b - Windowing mode is selected.
13 —	Reserved
12 PMODE	Power Mode Select 0b - Low Speed (LS) comparison mode is selected. 1b - High Speed (HS) comparison mode is selected.
11 INVT	Comparator invert This bit allows selecting the polarity of the analog comparator function. It is also driven to the COUT output (on both the device pin and as C0[COUT]) when CR1[OPE]=0. 0b - Does not invert the comparator output. 1b - Inverts the comparator output.
10 COS	Comparator Output Select 0b - Set CMPO to equal COUT (filtered comparator output). 1b - Set CMPO to equal COUTA (unfiltered comparator output).
9 OPE	Comparator Output Pin Enable The OPE bit enables the path from the comparator output to a selected pin. 0b - When OPE is 0, the comparator output (after window/filter settings dependent on software configuration) is not available to a packaged pin. 1b - When OPE is 1, and if the software has configured the comparator to own a packaged pin, the comparator is available in a packaged pin.
8 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, the analog part remains in the off state, and consumes no power. When the same input is selected from analog mux to the positive and negative port, the comparator is disabled automatically. 0b - Analog Comparator is disabled. 1b - Analog Comparator is enabled.
7 —	Reserved
6-4 FILTER_CNT	Filter Sample Count This field specifies the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency, please see the Functional Description. 000b - Filter is disabled. If SE = 1, then COUT is a logic zero (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001b - 1 consecutive sample must agree (comparator output is simply sampled). 010b - 2 consecutive samples must agree. 011b - 3 consecutive samples must agree. 100b - 4 consecutive samples must agree. 101b - 5 consecutive samples must agree. 110b - 6 consecutive samples must agree. 111b - 7 consecutive samples must agree.
3-2 —	Reserved
1-0 HYSTCTR	Comparator hard block hysteresis control. See chip data sheet to get the actual hystersis value with each level

Field	Function
	00b - The hard block output has level 0 hysteresis internally. 01b - The hard block output has level 1 hysteresis internally. 10b - The hard block output has level 2 hysteresis internally. 11b - The hard block output has level 3 hysteresis internally.

67.8.1.5 CMP Control Register 1 (C1)

67.8.1.5.1 Offset

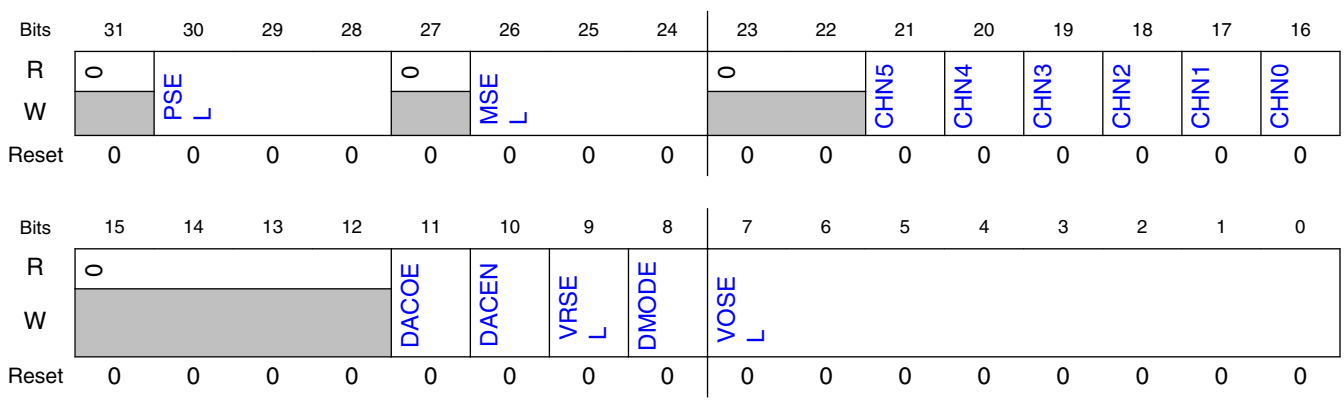
Register	Offset
C1	Ch

67.8.1.5.2 Function

Access:

- Supervisor read/write
- User read/write

67.8.1.5.3 Diagram



67.8.1.5.4 Fields

Field	Function
31 —	Reserved

Table continues on the next page...

Field	Function
30-28 PSEL	Plus Input MUX Control These bits determines which input is selected for the positive mux. Refer to the chip configuation chapters to get the detailed connections about Reference Input{0,1,2,3,4,5} and Internal Positive Input 0. The 8b DAC output has been connected internally. Note: For the round robin mode of operation, Internal Posivitive input 0 should be excluded from the activie channels and the fixed channel ports, meanwhile, the MSEL and PSEL bitfields in CMP_C1 register must have different values. 000b - Internal Positive Input 0 for Plus Channel -- Internal Plus Input 001b - External Input 1 for Plus Channel -- Reference Input 0 010b - External Input 2 for Plus Channel -- Reference Input 1 011b - External Input 3 for Plus Channel -- Reference Input 2 100b - External Input 4 for Plus Channel -- Reference Input 3 101b - External Input 5 for Plus Channel -- Reference Input 4 110b - External Input 6 for Plus Channel -- Reference Input 5 111b - Internal 8b DAC output
27 —	Reserved
26-24 MSEL	Minus Input MUX Control These bits determine which input is selected for the negative mux. Refer to the chip configuation chapters to get the detailed connections about Reference Input{0,1,2,3,4,5} and Internal Negative Input 0. The 8b DAC output has been connected internally. Note: For the round robin mode of operation, Internal negative input 0 should be excluded from the activie channels and the fixed channel ports, meanwhile, the MSEL and PSEL bitfields in CMP_C1 register must have different values. 000b - Internal Negative Input 0 for Minus Channel -- Internal Minus Input 001b - External Input 1 for Minus Channel -- Reference Input 0 010b - External Input 2 for Minus Channel -- Reference Input 1 011b - External Input 3 for Minus Channel -- Reference Input 2 100b - External Input 4 for Minus Channel -- Reference Input 3 101b - External Input 5 for Minus Channel -- Reference Input 4 110b - External Input 6 for Minus Channel -- Reference Input 5 111b - Internal 8b DAC output
23-22 —	Reserved
21 CHN5	Channel 5 input enable Channel 5 of the input enable for the round-robin checker. If CHN5 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
20 CHN4	Channel 4 input enable Channel 4 of the input enable for the round-robin checker. If CHN4 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
19 CHN3	Channel 3 input enable Channel 3 of the input enable for the round-robin checker. If CHN3 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
18 CHN2	Channel 2 input enable

Table continues on the next page...

Field	Function
	Channel 2 of the input enable for the round-robin checker. If CHN2 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
17 CHN1	Channel 1 input enable Channel 1 of the input enable for the round-robin checker. If CHN1 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
16 CHN0	Channel 0 input enable Channel 0 of the input enable for the round-robin checker. If CHN0 is set, then the corresponding channel to the non-fixed mux port is enabled to check its voltage value in the round-robin mode. If the same channel is selected as the reference voltage, this bit has no effect.
15-12 —	Reserved
11 DACOE	DAC Output Enable This bit is used to enable the DAC output. When the DAC output is selected as either the positive input by CMP_C1[PSEL] or negative input by CMP_C1[MSEL], this bit must be set otherwise the behavior will be unpredictable, same applies to round robin mode. 0b - DAC output is enabled 1b - DAC output is disabled
10 DACEN	DAC Enable This bit is used to enable the DAC. When the DAC is disabled, it is powered down to conserve power. 0b - DAC is disabled. 1b - DAC is enabled.
9 VRSEL	Supply Voltage Reference Source Select 0b - Vin1 is selected as resistor ladder network supply reference Vin. Vin1 is from internal PMC. 1b - Vin2 is selected as resistor ladder network supply reference Vin. Vin2 is from PAD.
8 DMODE	DAC Mode Selection 0b - DAC is selected to work in low speed and low power mode. 1b - DAC is selected to work in high speed high power mode.
7-0 VOSEL	DAC Output Voltage Select This bit selects an output voltage from one of 256 distinct levels. DACO = (Vin/256) * (VOSEL[7:0] + 1), so the DACO range is from Vin/256 to Vin.

67.8.1.6 CMP Control Register 2 (C2)

67.8.1.6.1 Offset

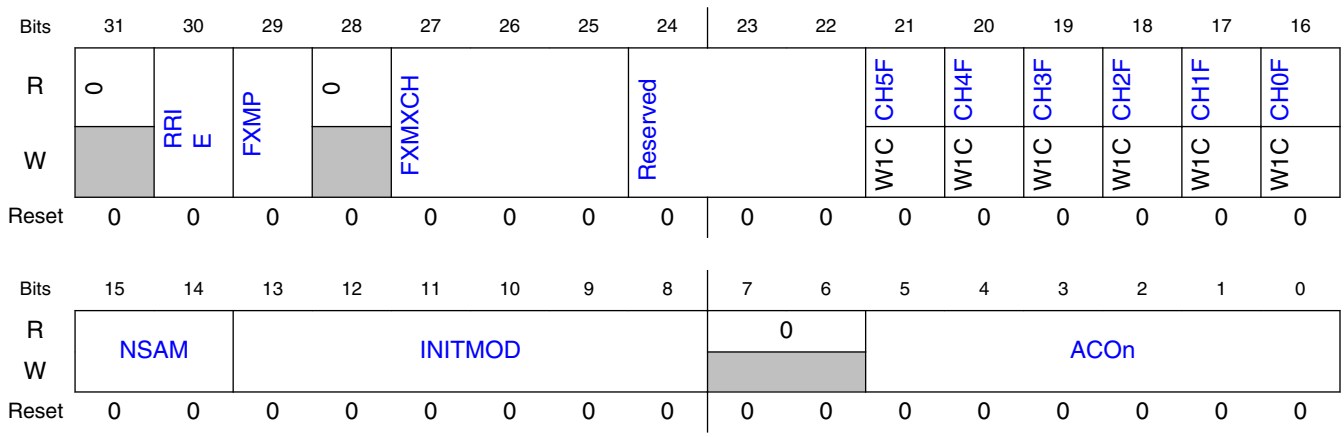
Register	Offset
C2	10h

67.8.1.6.2 Function

Access:

- Supervisor read/write
- User read/write

67.8.1.6.3 Diagram



67.8.1.6.4 Fields

Field	Function
31 —	Reserved
30 RRIE	Round-Robin interrupt enable This bit enables the interrupt/wake-up when the comparison result changes for a given channel. 0b - The round-robin interrupt is disabled. 1b - The round-robin interrupt is enabled when a comparison result changes from the last sample.
29 FXMP	Fixed MUX Port This bit is used to fix the analog mux port for the round-robin mode. 0b - The Plus port is fixed. Only the inputs to the Minus port are swept in each round. 1b - The Minus port is fixed. Only the inputs to the Plus port are swept in each round.
28 —	Reserved
27-25 FXMXCH	Fixed channel selection This field indicates which channel in the mux port is fixed in a given round-robin mode. If FXDACI is set, FXMXCH has no effect. 000b - External Reference Input 0 is selected as the fixed reference input for the fixed mux port. 001b - External Reference Input 1 is selected as the fixed reference input for the fixed mux port. 010b - External Reference Input 2 is selected as the fixed reference input for the fixed mux port. 011b - External Reference Input 3 is selected as the fixed reference input for the fixed mux port. 100b - External Reference Input 4 is selected as the fixed reference input for the fixed mux port.

Table continues on the next page...

Field	Function
	101b - External Reference Input 5 is selected as the fixed reference input for the fixed mux port. 110b - Reserved. 111b - The 8bit DAC is selected as the fixed reference input for the fixed mux port.
24-22 —	Reserved
21 CH5F	CH5F External Channel 5 input changed flag. This bit is set If the channel 5 input changed from the last comparison with the fixed mux port.
20 CH4F	CH4F External Channel 4 input changed flag. This bit is set If the channel 4 input changed from the last comparison with the fixed mux port.
19 CH3F	CH3F External Channel 3 input changed flag. This bit is set If the channel 3 input changed from the last comparison with the fixed mux port.
18 CH2F	CH2F External Channel 2 input changed flag. This bit is set If the channel 2 input changed from the last comparison with the fixed mux port.
17 CH1F	CH1F External Channel 1 input changed flag. This bit is set If the channel 1 input changed from the last comparison with the fixed mux port.
16 CH0F	CH0F External Channel 0 input changed flag. This bit is set If the channel 0 input changed from the last comparison with the fixed mux port.
15-14 NSAM	Number of sample clocks For a given channel, this field specifies how many round-robin clock cycles later the sample takes place. 00b - The comparison result is sampled as soon as the active channel is scanned in one round-robin clock. 01b - The sampling takes place 1 round-robin clock cycle after the next cycle of the round-robin clock. 10b - The sampling takes place 2 round-robin clock cycles after the next cycle of the round-robin clock. 11b - The sampling takes place 3 round-robin clock cycles after the next cycle of the round-robin clock.
13-8 INITMOD	Comparator and DAC initialization delay modulus. These values specify the round robin clock cycles used to determine the comparator and DAC initialization delays specified by the datasheet. For example the initialization delay is 80us and the round robin clock is 100kHz, then INITMOD should be set to 80us/10us = 8. 000000 - The modulus is set to 64(same with 111111). other values - Inilization delay is set to INITMOD * round robin clock period.
7-6 —	Reserved
5-0 ACOn	ACOn The result of the input comparison for channel *n* . This field stores the latest comparison result of the input channel *n* with the fixed mux port. Reading this bit returns the latest comparison result. Writing this field defines the pre-set state of channel *n*.

67.8.1.7 CMP Control Register 3 (C3)

67.8.1.7.1 Offset

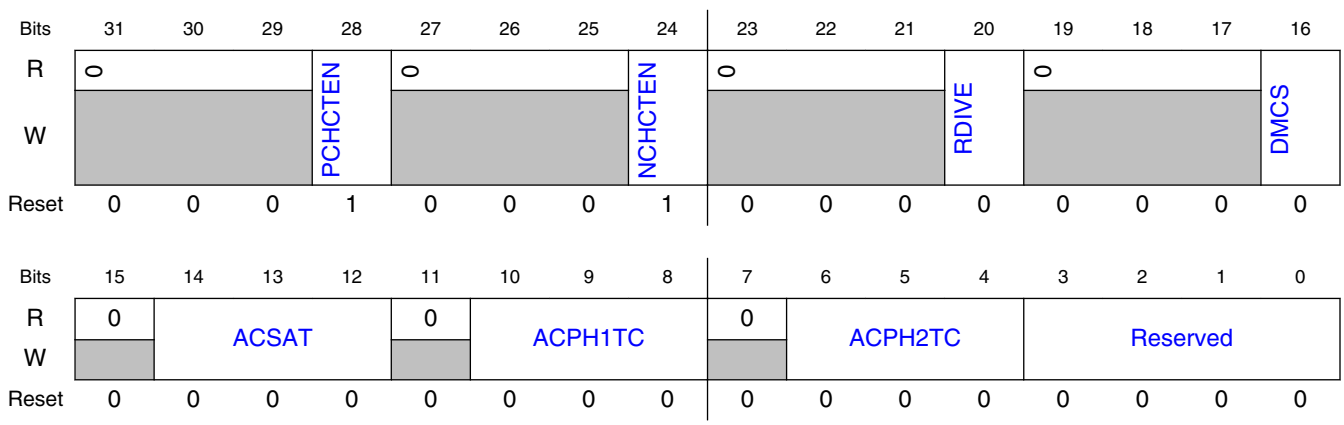
Register	Offset
C3	14h

67.8.1.7.2 Function

Access:

- Supervisor read/write
- User read/write

67.8.1.7.3 Diagram



67.8.1.7.4 Fields

Field	Function
31-29 —	Reserved
28 PCHCTEN	Positive Channel Continuous Mode Enable. This bit is used to enable the positive channel working in continuous mode. When this bit is set to 1, it means all the inputs to positive channels are coming from 1.8v domain and no discrete timing is required for the positive channel mux. Otherwise, when it is cleared to 0, positive channel will work in discrete mode, means the positive inputs are coming from 3v domain. 0b - Positive channel is in Discrete Mode and special timing needs to be configured.

Table continues on the next page...

Field	Function
	1b - Positive channel is in Continuous Mode and no special timing is requried.
27-25 —	Reserved
24 NCHCTEN	Negative Channel Continuous Mode Enable. This bit is used to enable the negative channel working in continuous mode. When this bit is set to 1, it means all the inputs to negative channels are coming from 1.8v domain and no discrete timing is required for the positive channel mux. Otherwise, when it is cleared to 0, negative channel will work in discrete mode, means the negative inputs are coming from 3v domain. 0b - Negative channel is in Discrete Mode and special timing needs to be configured. 1b - Negative channel is in Continuous Mode and no special timing is requried.
23-21 —	Reserved
20 RDIVE	Resistor Divider Enable This bit is used to enable the resistor divider for the inputs when they come from 3v domain and their values are above 1.8v. 0b - The resistor is not enabled even when either NCHEN or PCHEN is set to1 but the actual input is in the range of 0 - 1.8v. 1b - The resistor is enabled because the inputs are above 1.8v.
19-17 —	Reserved
16 DMCS	Discrete Mode Clock Selection This bit is used to select the clock source in order to generate the requiried timing for comparator to work in discrete mode. There are generally two kinds of clocks coming from SoC, one is a fast clock(16 - 20MHz) to generate fast prop delay and another one is noramlly 32kHz to work in low power mode. 0b - Slow clock is selected for the timing generation. 1b - Fast clock is selected for the timing generation.
15 —	Reserved
14-12 ACSAT	Analog Comparator Sampling Time control. These values configures the analog comparator sampling timing(speicified by the discrete mode clock period T which is selected by DMCS) in discrete mode. 000b - The sampling time equals to T 001b - The sampling time equasl to 2*T 010b - The sampling time equasl to 4*T 011b - The sampling time equasl to 8*T 100b - The sampling time equasl to 16*T 101b - The sampling time equasl to 32*T 110b - The sampling time equasl to 64*T 111b - The sampling time equasl to 256*T
11 —	Reserved
10-8 ACPH1TC	Analog Comparator Phase1 Timing Control. These values configures the analog comparator phase1 timing when RDIV is set to 1 which means the input volage level is above 1.8v. T means the discrete mode clock period in the table. 000b - Phase1 active time in one sampling period equals to T 001b - Phase1 active time in one sampling period equals to 2*T 010b - Phase1 active time in one sampling period equals to 4*T 011b - Phase1 active time in one sampling period equals to 8*T

Table continues on the next page...

Field	Function
	100b - Phase1 active time in one sampling period equals to T 101b - Phase1 active time in one sampling period equals to T 110b - Phase1 active time in one sampling period equals to T 111b - Phase1 active time in one sampling period equals to 0
7 —	Reserved
6-4 ACPH2TC	Analog Comparator Phase2 Timing Control. These values configures the analog comparator phase2 timing when RDIV is set to 1 which means the input volage level is above 1.8v. T means the discrete mode clock period in the table. 000b - Phase2 active time in one sampling period equals to T 001b - Phase2 active time in one sampling period equals to 2*T 010b - Phase2 active time in one sampling period equals to 4*T 011b - Phase2 active time in one sampling period equals to 8*T 100b - Phase2 active time in one sampling period equals to 16*T 101b - Phase2 active time in one sampling period equals to 32*T 110b - Phase2 active time in one sampling period equals to 64*T 111b - Phase2 active time in one sampling period equals to 16*T
3-0 —	Reserved

67.9 CMP functional description

The CMP module can be used to compare two analog input voltages applied to INP and INM. CMPO is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting CMP_C0[INVT] = 1.

CMP_C0[IER] and CMP_C0[IEF] are used to select the condition that causes the CMP module to assert an interrupt to the processor. CMP_C0[CFF] is set on a falling edge, and CMP_C0[CFR] is set on a rising edge of the comparator output. The optionally filtered CMPO can be read directly through CMP_C0[COUT].

67.9.1 Initialization

A typical startup sequence is as follows.

The time required to stabilize COUT is the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function, and filter. See the datasheet for power-on delays of the comparators. The windowing function has a maximum of one bus clock period delay. The filter delay is specified in the Low-pass filter section.

During operation, the propagation delay of the selected data paths must always be considered. It may take many bus clock cycles for COUT and CMP_C0[CFR]/ CMP_C0[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.

When programmed for filtering modes, COUT initially equals 0 until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

67.9.2 Low-pass filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT. Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by CMP_C0[FPR], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high-frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

67.9.2.1 Enabling filter modes

Filter modes can be enabled by:

- Setting CMP_C0[FILTER_CNT] > 0x01 and
- Setting CMP_C0[FPR] to a nonzero value or setting C0[SE]=1

If using the divided bus clock to drive the filter, it samples COUTA every CMP_C0[FPR] bus clock cycles.

The filter output is at logic 0 when first initialized, and subsequently changes when all the consecutive CMP_C0[FILTER_CNT] samples agree that the output value has changed. In other words, CMP_C0[COUT] is 0 for some initial period, even when COUTA is at logic 1.

Setting all of CMP_C0[SE], CMP_C0[FPR] and CMP_C0[FILTER_CNT] to 0 disables the filter and eliminates switching current associated with the filtering process.

Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching CMP_C0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CMP_C0[SE]=1, the filter samples COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive CMP_C0[FILTER_CNT] samples agree that the output value has changed.

67.9.2.2 Latency issues

The value of CMP_C0[FPR] or SAMPLE period must be set such that the sampling period is just longer than the period of the expected noise. This way a noise spike will corrupt only one sample. The value of CMP_C0[FILTER_CNT] must be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CMP_C0[FILTER_CNT].

The values of CMP_C0[FPR] or SAMPLE period and CMP_C0[FILTER_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CMP_C0[FILTER_CNT].

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

Table 67-5. Comparator sample/filter maximum latencies

Mode #	CMP_C0[EN]	CMP_C0[WE]	CMP_C0[SE]	CMP_C0[FILTER_CNT]	Co[FPR]	Operation	Maximum latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	Sampled, Non-Filtered mode	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	0x01	> 0x00		$T_{PD} + (CMP\_C0[FPR] * T_{per}) + T_{per}$

Table continues on the next page...

Table 67-5. Comparator sample/filter maximum latencies (continued)

Mode #	CMP_C0[EN]	CMP_C0[WE]	CMP_C0[SE]	CMP_C0[FILTER_CNT]	Co[FPR]	Operation	Maximum latency[1]
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (CMP_C0[FILTER_CNT] * T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (CMP_C0[FILTER_CNT] * CMP_C0[FPR] x T_{per}) + T_{per}
5A	1	1	0	0x00	X	Windowed mode	T_{PD} + T_{per}
5B	1	1	0	X	0x00		T_{PD} + T_{per}
6	1	1	0	0x01	0x01 - 0xFF	Windowed / Resampled mode	T_{PD} + (CMP_C0[FPR] * T_{per}) + 2T_{per}
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (CMP_C0[FILTER_CNT] * CMP_C0[FPR] x T_{per}) + 2T_{per}

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

NOTE

The delay in this table does not include the delay caused by the Discrete Mode.

67.9.3 CMP Discrete mode timing sequence

In order to support the 3 V domain input the analog comparator should be configured to work in Discrete Mode since the internal transistors are not 3 V tolerance. This section summarizes the possible configurations depending on the input range.

Note that the sample signal below means that the analog comparator samples the input analog input. It is different from the sample mode that occurs on the digital process to the final comparison result. Also, note that the cross domain round robin cycling (see Trigger mode) is not allowed.

1. When CMP_C3[PCHCTEN] and CMP_C3[NCHCTEN] are set to 1, the CMP is in continuous time operation. No special timing is required.
2. When either CMP_C3[PCHCTEN] or CMP_C3[NCHCTEN] is 0, it means at least one input comes from the 3V PAD.
   a. If CMP_C3[RDIVE] is zero, it means the input signal comes from 3V PAD, but its range still is in the range of 0 to 1.8V, no timing requirement for the generation of Phase1 and Phase2.

In this condition, there are two modes for this comparator, high speed mode and low speed mode. Clock source may come from fast clock(16-20M)

In Figure 67-13, the Analog ready indicates the analog settling time is reached and is ready for comparison. Tc is set by CMP_C3[ACSAT], in high speed mode, it is normally used as 1*T, 2*T, 4*T and 8*T where T is CLOCK period, and in low speed mode (CMP_C0[PMODE] = 0), CMP_C3[ACSAT] is normally used as 16*T, 32*T, 64*T and 256*T.

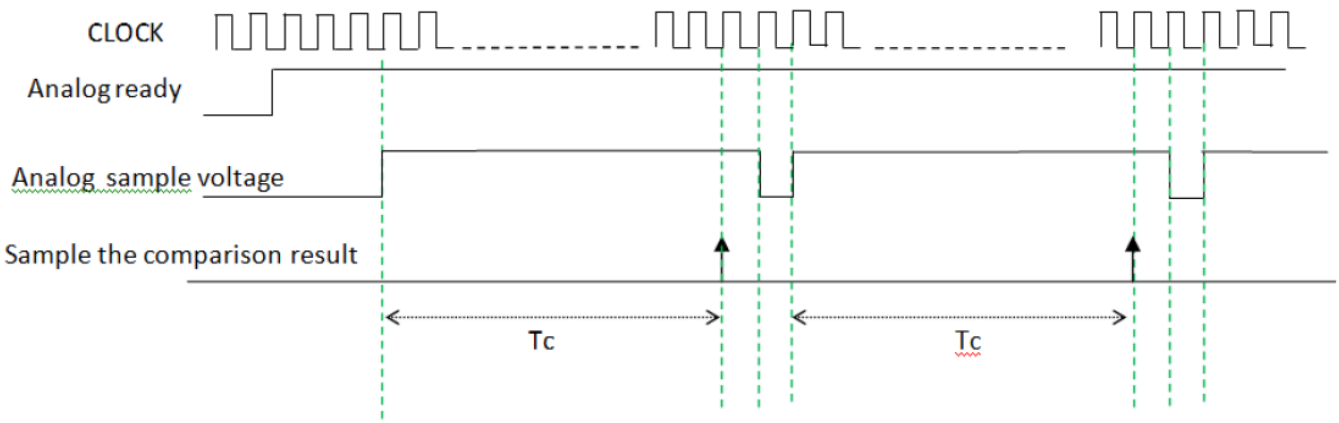


Figure 67-13. CMP_C3[RDIVE] = 0

b. If CMP_C3[RDIVE] == 1, it means the signals come from the 3V PAD and it could over 1.8 V, Phase1, Phase2 will be generated.

If CMP is in high speed mode (CMP_C0[PMODE] = 1) , Tc, Phase1, and Phase2 can be set based on the possible combinations in Table 67-6.

If CMP works in low speed mode, Phase1 is suggested to set to 1*T, Phase2 is suggested to set to 8*T, and Tc is normally set to 16*T, 32*T, 64*T and 256*T.

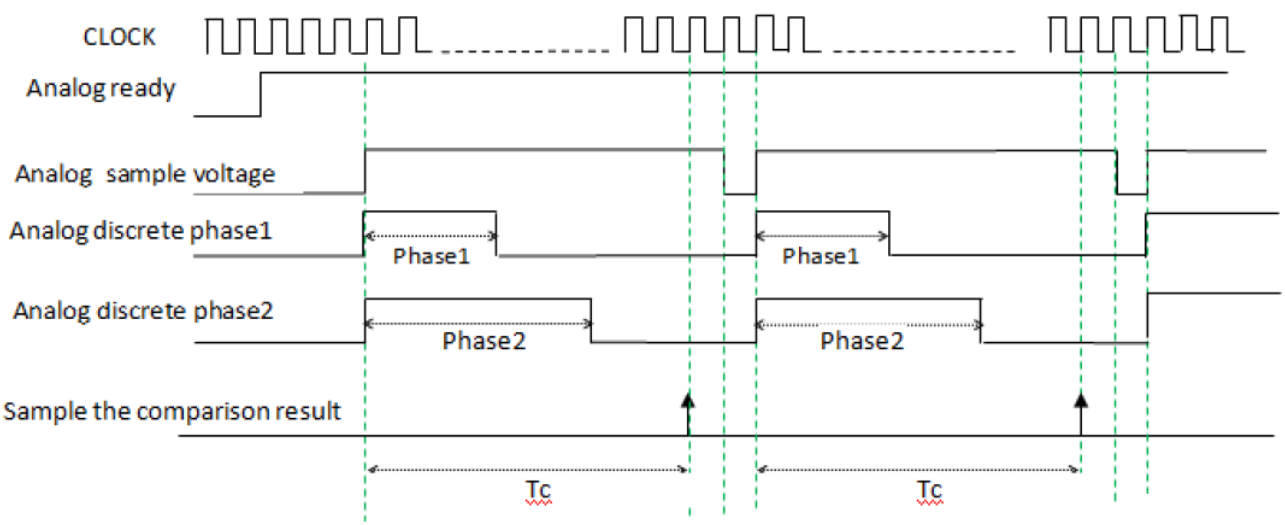


Figure 67-14. CMP_C3[RDIVE] = 1

Table 67-6. Possible Combinations of CMP_C3[ACSAT], CMP_C3[ACPH1TC] and CMP_C3[ACPH2TC]

Tc	Phase1	Phase2
1*T	1*T	1*T
2*T	2*T	2*T
4*T	4*T	4*T
8*T	8*T	8*T
16*T	1*T	16*T
32*T	1*T	32*T
64*T	1*T	64*T
256*T	0	16*T

If the clock comes from 32KHz slow clock(CMP_C3[DMCLKSEL] = 0), both analog phase 1 and phase 2 will be driven to 0 and Tc is limited to T as shown in the figure below.

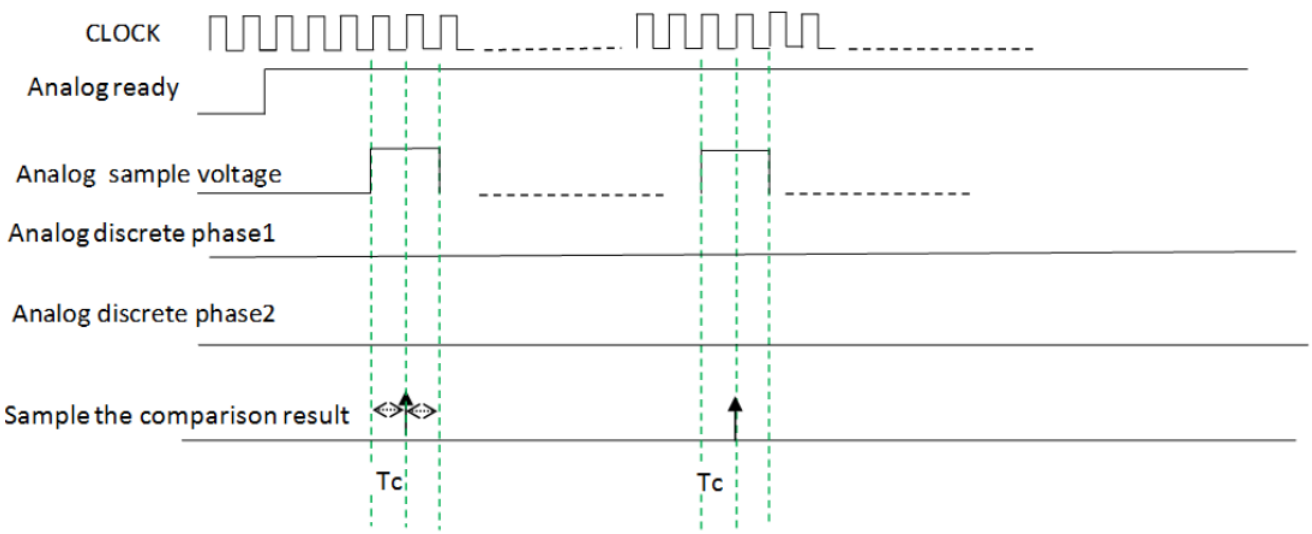


Figure 67-15. Slow (32 kHz) clock timing

67.10 Interrupts

The CMP module is capable of generating an interrupt on either the rising- or falling-edge of the comparator output, or both. Assuming the CMP DMA enable bit is not set, the following table gives the conditions in which the interrupt request is asserted and deasserted.

Table 67-7. CMP interrupt generations

When	Then
CMP_C0[IER] and CMP_C0[CFR] are set	The interrupt request is asserted
CMP_C0[IEF] and CMP_C0[CFF] are set	The interrupt request is asserted
CMP_C0[IER] and CMP_C0[CFR] are cleared for a rising-edge interrupt	The interrupt request is deasserted
CMP_C0[IEF] and CMP_C0[CFF] are cleared for a falling-edge interrupt	The interrupt request is deasserted

67.11 DMA support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support is enabled by setting CMP_C0[DMAEN] and the interrupt is enabled by setting CMP_C0[IER], CMP_C0[IEF], or both, the corresponding change on COUT forces a DMA transfer request rather than a CPU interrupt instead. When the DMA has completed the transfer, it sends a transfer completing indicator signal that deasserts the DMA transfer request and clears the flags (CMP_C0[CFR] and CMP_C0[CFF]) to allow a subsequent change on comparator output to occur and force another DMA request.

The comparator can remain functional in STOP modes. When DMA support is enabled by setting CMP_C0[DMAEN] and the interrupt is enabled by setting CMP_C0[IER], CMP_C0[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, the system will go back to STOP modes. Refer to the DMA chapters in the device reference manual for the asynchronous DMA function for details.

67.12 DAC functional description

This section provides DAC functional description.

67.12.1 Digital-to-analog converter block diagram

The following figure shows the block diagram of the DAC module. It contains a 256-tap resistor ladder network and a 256-to-1 multiplexer, which selects an output voltage from one of 256 distinct levels that outputs from DACO. It is controlled through the control register2(CMP_C1). Its supply reference source can be selected from two sources V_{in1} and V_{in2}. The module can be powered down or disabled when not in use. When in Disabled mode, DACO is connected to the analog ground.

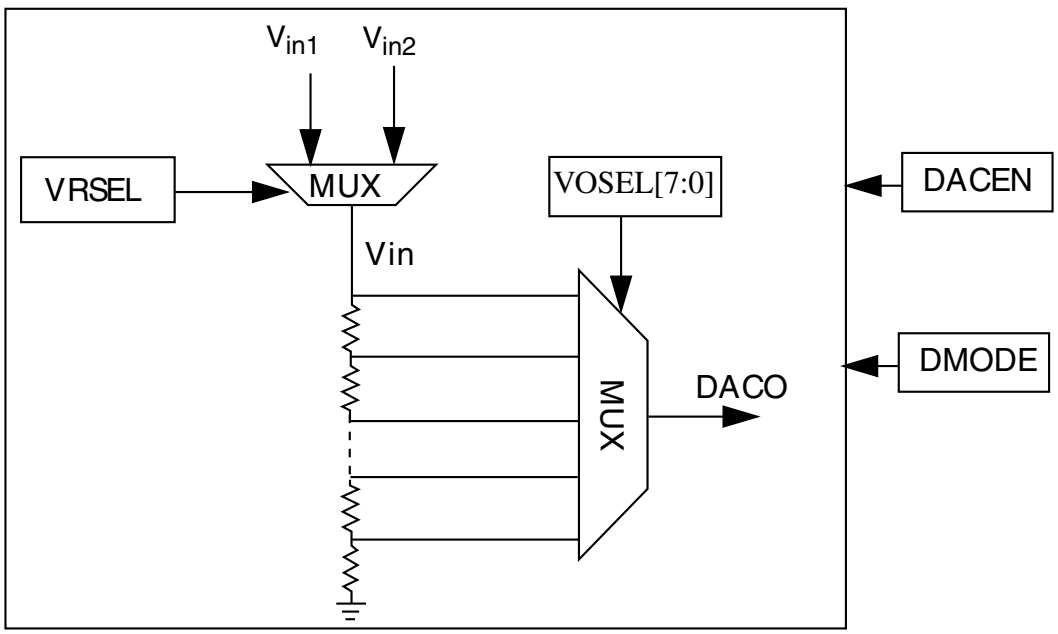


Figure 67-16. 8-bit DAC block diagram

67.12.2 Voltage reference source select

- V_{in1} must be used to connect to the primary voltage source as supply reference of 64 tap resistor ladder
- V_{in2} must be used to connect to an alternate voltage source, or primary source, if an alternate voltage source is not available

67.13 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

67.14 DAC clocks

This module has a single clock input, the bus clock.

67.15 DAC interrupts

This module has no interrupts.

67.16 Trigger mode

The CMP and the 8-bit DAC are designed to support the trigger mode operation, which is enabled when the MCU enters STOP modes with CMP_C0[WE], CMP_C0[SE] and CMP_C0[EN] are set.

With this mode enabled, the trigger events that include the operation clock and a trigger start signal will initiate a compare sequence that must first enable the CMP and DAC prior to performing a CMP operation and capturing the output. A fixed channel for either the plus side mux or the minus side mux is selected by software with CMP_C2[FXMP] and CMP_C2[FXMXCH]. It is a mandatory request that the round robin cycling period must be set longer than the time that all the active channels complete the specified comparison cycles specified by CMP_C2[NSAM].

The active channels selected by CMP_C1[CHN*n*] are then routed to the non-fixed channel mux and compared with the reference input in a round-robin manner. In order to meet the comparator stabilization time, after the configurable number of operation clocks defined by CMP_C2[NSAM], the comparison result is sampled for the selected channel. A software pre-programmed state for each channel is configured by writing to CMP_C2[ACO*n*] field. After all the active channels are sampled, if the comparison result changes from its pre-programmed state, the corresponding flag in CMP_C2[CH*n*F] is set. If CMP_C2[RRIE] is set, an asynchronous reset is asserted to bring the MCU out of STOP mode.

Note that these flags do not support generating a DMA transfer event.

This mode is active when the MCU is in STOP mode, so none of the window/filter functions are available. A basic assumption of this mode is that the selected inputs are changing at a much slower rate than the operation clock. It is suggested to configure the comparator in low power comparison mode as well. In programming the

CMP_C2[INITMOD] registers it is need to make sure the INITMOD*round robin clock period must be longer than the initialization delay which can be referred from the chip datasheet.

The following diagram shows the basic flow of this mode. In the diagram, CMP_C1[CHN1], CMP_C1[CHN3], and CMP_C1[CHN4] are set, so channels #1, #3, and #4 are selected for round robin depended on their polarity setting. CMP_C2[NSAM] is set to 2'b01, so one clock later the comparison result of the selected channel is sampled. When channel #3 is compared, the result is sampled, and round robin ends. If any of the comparison results from channel #1, #3, or #4 changed from their programmed value (written to CMP_C2[ACO1], CMP_C2[ACO3], and CMP_C2[ACO4]), an interrupt is generated to wake up the MCU from the STOP mode. Software can then poll the CMP_C2[CH*n*F] to see which channel input(s) changed value during the STOP mode.

NOTE

In round robin mode, it should be ensured that the RTC_CLK period is greater than the comparison time corresponding to the value of CMPx_C0[PMODE].

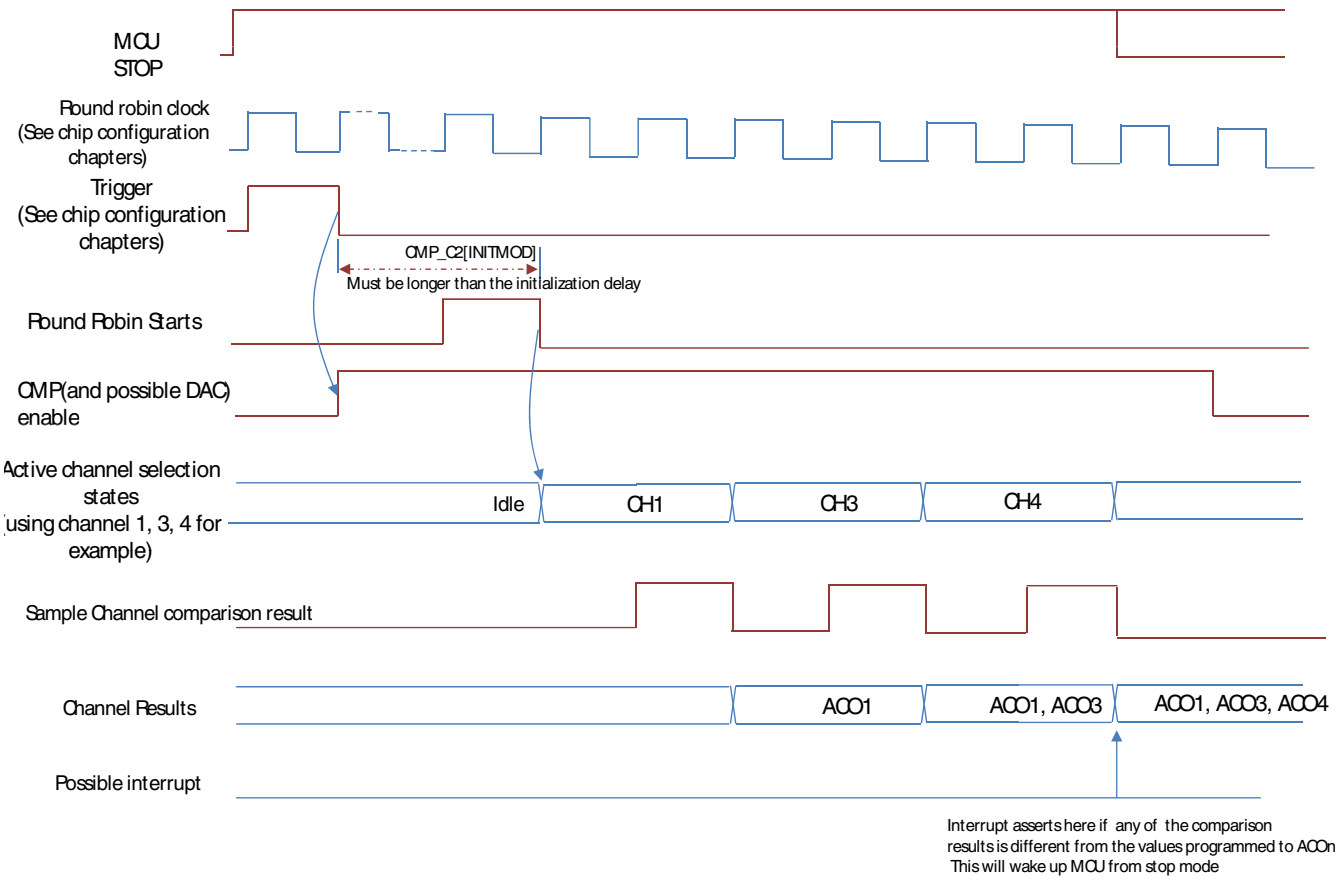


Figure 67-17. Trigger mode

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number IMX7ULPRM
Revision 0, 06/2019