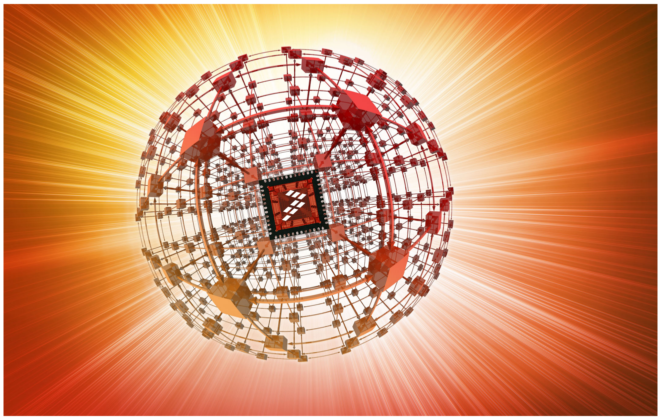
K20 Sub-Family Reference Manual

Supports: MK20FX512VLQ12, MK20FN1M0VLQ12,
MK20FX512VMD12, MK20FN1M0VMD12



Document Number: K20P144M120SF3RM
Rev. 2, Dec 2011

Energy Efficient Solutions optimized for low power

Preliminary

freescale

Contents

Section Number **Title** **Page**

Chapter 1
About This Document

Chapter 2
Introduction

Chapter 3
Chip Configuration

Section Number | Title | Page

Chapter 4
Memory Map

Section Number Title Page

Chapter 5
Clock Distribution

Chapter 6
Reset and Boot

Chapter 7
Power Management

Chapter 8
Security

Section Number | Title | Page

Chapter 9
Debug

Chapter 10 Signal Multiplexing and Signal Descriptions

Chapter 11 Port control and interrupts (PORT)

Chapter 12
System integration module (SIM)

Chapter 13
Reset Control Module (RCM)

Chapter 14
System Mode Controller

Chapter 15
Power Management Controller

Chapter 16
Low-Leakage Wake-up Unit (LLWU)

Chapter 17
Miscellaneous Control Module (MCM)

Chapter 18
Crossbar Switch (AXBS)

Chapter 19
Memory Protection Unit (MPU)

Chapter 20 Peripheral Bridge (AIPS-Lite)

Chapter 21 Direct memory access multiplexer (DMAMUX)

Chapter 22
Direct Memory Access Controller (eDMA)

Section Number Title Page

Chapter 23
External Watchdog Monitor (EWM)

Chapter 24
Watchdog Timer (WDOG)

Chapter 25
Multipurpose Clock Generator (MCG)

Chapter 26
Oscillator (OSC)

Chapter 27
RTC Oscillator

Chapter 28
Local Memory Controller

Section Number Title Page

Chapter 29
Flash Memory Controller (FMC)

Chapter 30
Flash Memory Module (FTFE)

Chapter 31
EzPort

Chapter 32
NAND Flash Controller (NFC)

Chapter 33
External Bus Interface (FlexBus)

Chapter 34
Cyclic redundancy check (CRC)

Chapter 35
Analog-to-Digital Converter (ADC)

Chapter 36 Comparator (CMP)

Chapter 37
12-bit Digital-to-Analog Converter (DAC)

Chapter 38
Voltage Reference (VREFV1)

Chapter 39
Programmable Delay Block (PDB)

Chapter 40
FlexTimer (FTM)

Chapter 41
Periodic Interrupt Timer (PIT)

Chapter 42
Low power timer (LPTMR)

Chapter 43
Carrier Modulator Transmitter (CMT)

Section Number Title Page

Chapter 44
Real Time Clock (RTC)

Chapter 45
Universal Serial Bus OTG Controller (USBOTG)

Section Number	Title	Page

Chapter 46
USB Device Charger Detection Module (USBDCD)

Chapter 47
USB Voltage Regulator

Chapter 48
USB High Speed OTG Controller (USBHS)

Chapter 49
CAN (FlexCAN)

Section Number | Title | Page

Chapter 50
SPI (DSPI)

Chapter 51
Inter-Integrated Circuit (I2C)

Chapter 52
Universal Asynchronous Receiver/Transmitter (UART)

Section Number Title Page

Section Number | Title | Page

Chapter 53
Secured digital host controller (SDHC)

Chapter 54
Synchronous Audio Interface (SAI)

Section Number | Title | Page

Chapter 55
General purpose input/output (GPIO)

Chapter 56
Touch sense input (TSI)

Chapter 57
JTAG Controller (JTAGC)

Chapter 1
About This Document

1.1 Overview

1.1.1 Purpose

This document describes the features, architecture, and programming model of the Freescale K20 microcontroller.

1.1.2 Audience

This document is primarily for system architects and software application developers who are using or considering using the K20 microcontroller in a system.

1.2 Conventions

1.2.1 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*.

1.2.2 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.

1.2.3 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, or field that is either reserved for future use or for which, when written to, the module or chip behavior is unpredictable.

Chapter 2
Introduction

2.1 Overview

This chapter provides an overview of the Kinetis portfolio and K20 family of products. It also presents high-level descriptions of the modules available on the devices covered by this document.

2.2 K20 Family Introduction

The K20 MCU family is pin, peripheral and software compatible with the K10 MCU family and adds full and high-speed USB 2.0 On-The-Go with device charger detect capability. Devices start from 32 KB of flash in 5 x 5 mm 32QFN packages extending up to 1 MB in a 144MAPBGA package with a rich suite of analog, communication, timing and control peripherals. High memory density K20 family devices include a single precision floating point unit and NAND flash controller.

2.3 Module Functional Categories

The modules on this device are grouped into functional categories. The following sections describe the modules assigned to each category in more detail.

Table 2-1. Module functional categories

Module category	Description
ARM Cortex-M4 core	• 32-bit MCU core from ARM's Cortex-M class adding DSP instructions and optional single-precision floating point unit, 1.25 DMIPS/MHz, based on ARMv7 architecture with 16 KB of cache in some devices

Table continues on the next page...

Table 2-1. Module functional categories (continued)

Module category	Description
System	• System integration module • Power management and mode controllers • Multiple power modes available based on run, wait, stop, and power-down modes • Low-leakage wakeup unit • Miscellaneous control module • Crossbar switch • Memory protection unit • Peripheral bridge • Direct memory access (DMA) controller with multiplexer to increase available DMA requests • External watchdog monitor • Watchdog
Memories	• Internal memories include: • Program flash memory • On devices with FlexMemory: FlexMemory • FlexNVM • FlexRAM • On devices with program flash only: Programming acceleration RAM • SRAM • Cache memory • External memory or peripheral bus interface: FlexBus • Serial programming interface: EzPort • NAND flash controller
Clocks	• Multiple clock generation options available from internally- and externally-generated clocks • System oscillator to provide clock source for the MCU • RTC oscillator to provide clock source for the RTC
Security	• Cyclic Redundancy Check module for error detection
Analog	• High speed analog-to-digital converter with integrated programmable gain amplifier • Comparator • Digital-to-analog converter • Internal voltage reference
Timers	• Programmable delay block • FlexTimers • Periodic interrupt timer • Low power timer • Carrier modulator transmitter • Independent real time clock
Communications	• USB OTG controller with built-in FS/LS transceiver • USB device charger detect • USB voltage regulator • HS USB OTG controller with ULPI interface • CAN • Serial peripheral interface • Inter-integrated circuit (I^2C) • UART • Secured Digital host controller • Integrated interchip sound (I^2S)

Table continues on the next page...

Table 2-1. Module functional categories (continued)

Module category	Description
Human-Machine Interfaces (HMI)	• General purpose input/output controller • Capacitive touch sense input interface enabled in hardware

2.3.1 ARM Cortex-M4 Core Modules

The following core modules are available on this device.

Table 2-2. Core modules

Module	Description
ARM Cortex-M4	The ARM Cortex-M4 is the newest member of the Cortex M Series of processors targeting microcontroller cores focused on very cost sensitive, deterministic, interrupt driven environments. The Cortex M4 processor is based on the ARMv7 Architecture and Thumb®-2 ISA and is upward compatible with the Cortex M3, Cortex M1, and Cortex M0 architectures. Cortex M4 improvements include an ARMv7 Thumb-2 DSP (ported from the ARMv7-A/R profile architectures) providing 32-bit instructions with SIMD (single instruction multiple data) DSP style multiply-accumulates and saturating arithmetic.
Floating point unit (FPU)	A single-precision floating point unit (FPU) that is compliant to the *IEEE Standard for Floating-Point Arithmetic* (IEEE 754).
NVIC	The ARMv7-M exception model and nested-vectored interrupt controller (NVIC) implement a relocatable vector table supporting many external interrupts, a single non-maskable interrupt (NMI), and priority levels. The NVIC replaces shadow registers with equivalent system and simplified programmability. The NVIC contains the address of the function to execute for a particular handler. The address is fetched via the instruction port allowing parallel register stacking and look-up. The first sixteen entries are allocated to ARM internal sources with the others mapping to MCU-defined interrupts.
AWIC	The primary function of the Asynchronous Wake-up Interrupt Controller (AWIC) is to detect asynchronous wake-up events in stop modes and signal to clock control logic to resume system clocking. After clock restart, the NVIC observes the pending interrupt and performs the normal interrupt or event processing.
Debug interfaces	Most of this device's debug is based on the ARM CoreSight™ architecture. Four debug interfaces are supported: • IEEE 1149.1 JTAG • IEEE 1149.7 JTAG (cJTAG) • Serial Wire Debug (SWD) • ARM Real-Time Trace Interface

2.3.2 System Modules

The following system modules are available on this device.

Table 2-3. System modules

Module	Description
System integration module (SIM)	The SIM includes integration logic and several module configuration settings.
System mode controller	The SMC provides control and protection on entry and exit to each power mode, control for the Power management controller (PMC), and reset entry and exit for the complete MCU.
Power management controller (PMC)	The PMC provides the user with multiple power options. Ten different modes are supported that allow the user to optimize power consumption for the level of functionality needed. Includes power-on-reset (POR) and integrated low voltage detect (LVD) with reset (brownout) capability and selectable LVD trip points.
Low-leakage wakeup unit (LLWU)	The LLWU module allows the device to wake from low leakage power modes (LLS and VLLS) through various internal peripheral and external pin sources.
Miscellaneous control module (MCM)	The MCM includes integration logic and embedded trace buffer details.
Crossbar switch (XBS)	The XBS connects bus masters and bus slaves, allowing all bus masters to access different bus slaves simultaneously and providing arbitration among the bus masters when they access the same slave.
Memory protection unit (MPU)	The MPU provides memory protection and task isolation. It concurrently monitors all bus master transactions for the slave connections.
Peripheral bridges	The peripheral bridge converts the crossbar switch interface to an interface to access a majority of peripherals on the device.
DMA multiplexer (DMAMUX)	The DMA multiplexer selects from many DMA requests down to 16 for the DMA controller.
Direct memory access (DMA) controller	The DMA controller provides programmable channels with transfer control descriptors for data movement via dual-address transfers for 8-, 16-, 32- and 128-bit data values.
External watchdog monitor (EWM)	The EWM is a redundant mechanism to the software watchdog module that monitors both internal and external system operation for fail conditions.
Software watchdog (WDOG)	The WDOG monitors internal system operation and forces a reset in case of failure. It can run from an independent 1 KHz low power oscillator with a programmable refresh window to detect deviations in program flow or system frequency.

2.3.3 Memories and Memory Interfaces

The following memories and memory interfaces are available on this device.

Table 2-4. Memories and memory interfaces

Module	Description
Flash memory	- Program flash memory — non-volatile flash memory that can execute program code - FlexMemory — encompasses the following memory types: - For devices with FlexNVM: FlexNVM — Non-volatile flash memory that can execute program code, store data, or backup EEPROM data - For devices with FlexNVM: FlexRAM — RAM memory that can be used as traditional RAM or as high-endurance EEPROM storage, and also accelerates flash programming - For devices with only program flash memory: Programming acceleration RAM — RAM memory that accelerates flash programming
Flash memory controller	Manages the interface between the device and the on-chip flash memory.
SRAM	Internal system RAM. Partial SRAM kept powered in VLLS2 low leakage mode.
Local memory controller	Manages simultaneous accesses to system RAM by multiple master peripherals and core. Controls cache which improves system performance by providing single-cycle access to the instruction and data pipelines.
System register file	32-byte register file that is accessible during all power modes and is powered by VDD.
VBAT register file	32-byte register file that is accessible during all power modes and is powered by VBAT.
Serial programming interface (EzPort)	Same serial interface as, and subset of, the command set used by industry-standard SPI flash memories. Provides the ability to read, erase, and program flash memory and reset command to boot the system after flash programming.
FlexBus	External bus interface with multiple independent, user-programmable chip-select signals that can interface with external SRAM, PROM, EPROM, EEPROM, flash, and other peripherals via 8-, 16- and 32-bit port sizes. Configurations include multiplexed or non-multiplexed address and data buses using 8-bit, 16-bit, 32-bit, and 16-byte line-sized transfers.
NAND flash controller	8-bit and 16-bit NAND flash interface with page sizes of 512 bytes, 2 KB, 4 KB, and 8 KB. Supports up to 32-bit ECC error correction.

2.3.4 Clocks

The following clock modules are available on this device.

Table 2-5. Clock modules

Module	Description
Multi-clock generator (MCG)	The MCG provides several clock sources for the MCU that include: - Phase-locked loop (PLL) — Voltage-controlled oscillator (VCO) - Frequency-locked loop (FLL) — Digitally-controlled oscillator (DCO) - Internal reference clocks — Can be used as a clock source for other on-chip peripherals
System oscillator	The system oscillator, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

Table continues on the next page...

Table 2-5. Clock modules (continued)

Module	Description
Real-time clock oscillator	The RTC oscillator has an independent power supply and supports a 32 kHz crystal oscillator to feed the RTC clock. Optionally, the RTC oscillator can replace the system oscillator as the main oscillator source.

2.3.5 Security and Integrity modules

The following security and integrity modules are available on this device:

Table 2-6. Security and integrity modules

Module	Description
Cyclic Redundancy Check (CRC)	Hardware CRC generator circuit using 16/32-bit shift register. Error detection for all single, double, odd, and most multi-bit errors, programmable initial seed value, and optional feature to transpose input data and CRC result via transpose register.

2.3.6 Analog modules

The following analog modules are available on this device:

Table 2-7. Analog modules

Module	Description
16-bit analog-to-digital converters (ADC) and programmable-gain amplifiers (PGA)	16-bit successive-approximation ADC designed with integrated programmable gain amplifiers (PGA)
Analog comparators	Compares two analog input voltages across the full range of the supply voltage.
6-bit digital-to-analog converters (DAC)	64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed.
12-bit digital-to-analog converters (DAC)	Low-power general-purpose DAC, whose output can be placed on an external pin or set as one of the inputs to the analog comparator or ADC.
Voltage reference (VREF)	Supplies an accurate voltage output that is trimmable in 0.5 mV steps. The VREF can be used in medical applications, such as glucose meters, to provide a reference voltage to biosensors or as a reference to analog peripherals, such as the ADC, DAC, or CMP.

2.3.7 Timer modules

The following timer modules are available on this device:

Table 2-8. Timer modules

Module	Description
Programmable delay block (PDB)	• 16-bit resolution • 3-bit prescaler • Positive transition of trigger event signal initiates the counter • Supports two triggered delay output signals, each with an independently-controlled delay from the trigger event • Outputs can be OR'd together to schedule two conversions from one input trigger event and can schedule precise edge placement for a pulsed output. This feature is used to generate the control signal for the CMP windowing feature and output to a package pin if needed for applications, such as critical conductive mode power factor correction. • Continuous-pulse output or single-shot mode supported, each output is independently enabled, with possible trigger events • Supports bypass mode • Supports DMA
Flexible timer modules (FTM)	• Selectable FTM source clock, programmable prescaler • 16-bit counter supporting free-running or initial/final value, and counting is up or up-down • Input capture, output compare, and edge-aligned and center-aligned PWM modes • Operation of FTM channels as pairs with equal outputs, pairs with complimentary outputs, or independent channels with independent outputs • Deadtime insertion is available for each complementary pair • Generation of hardware triggers • Software control of PWM outputs • Up to 4 fault inputs for global fault control • Configurable channel polarity • Programmable interrupt on input capture, reference compare, overflowed counter, or detected fault condition • Quadrature decoder with input filters, relative position counting, and interrupt on position count or capture of position count on external event • DMA support for FTM events
Periodic interrupt timers (PIT)	• Four general purpose interrupt timers • Interrupt timers for triggering ADC conversions • 32-bit counter resolution • Clocked by system clock frequency • DMA support
Low-power timer (LPTimer)	• Selectable clock for prescaler/glitch filter of 1 kHz (internal LPO), 32.768 kHz (external crystal), or internal reference clock • Configurable Glitch Filter or Prescaler with 16-bit counter • 16-bit time or pulse counter with compare • Interrupt generated on Timer Compare • Hardware trigger generated on Timer Compare
Carrier modulator timer (CMT)	• Four CMT modes of operation: • Time with independent control of high and low times • Baseband • Frequency shift key (FSK) • Direct software control of CMT_IRO pin • Extended space operation in time, baseband, and FSK modes • Selectable input clock divider • Interrupt on end of cycle with the ability to disable CMT_IRO pin and use as timer interrupt • DMA support

Table continues on the next page...

Table 2-8. Timer modules (continued)

Module	Description
Real-time clock (RTC)	• Independent power supply, POR, and 32 kHz Crystal Oscillator • 32-bit seconds counter with 32-bit Alarm • 16-bit Prescaler with compensation that can correct errors between 0.12 ppm and 3906 ppm • 64-bit monotonic counter • Time and monotonic counters are invalidated on tamper detection

2.3.8 Communication interfaces

The following communication interfaces are available on this device:

Table 2-9. Communication modules

Module	Description
USB OTG (low-/full-speed)	USB 2.0 compliant module with support for host, device, and On-The-Go modes. Includes an on-chip transceiver for full and low speeds.
USB OTG (low-/full-/high-speed)	USB 2.0 compliant module with support for host, device, and On-The-Go modes. Provides ULPI interface for low, full, and high speeds.
USB Device Charger Detect (USBDCD)	The USBDCD monitors the USB data lines to detect a smart charger meeting the USB Battery Charging Specification Rev1.1. This information allows the MCU to better manage the battery charging IC in a portable device.
USB voltage regulator	Up to 5 V regulator input typically provided by USB VBUS power with 3.3 V regulated output that powers on-chip USB subsystem, capable of sourcing 120 mA to external board components.
Controller Area Network (CAN)	Supports the full implementation of the CAN Specification Version 2.0, Part B
Serial peripheral interface (SPI)	Synchronous serial bus for communication to an external device
Inter-integrated circuit (I2C)	Allows communication between a number of devices. Also supports the System Management Bus (SMBus) Specification, version 2.
Universal asynchronous receiver/ transmitters (UART)	Asynchronous serial bus communication interface with programmable 8- or 9-bit data format and support of CEA709.1-B (LON), ISO 7816 smart card interface
Secure Digital host controller (SDHC)	Interface between the host system and the SD, SDIO, MMC, or CE-ATA cards. The SDHC acts as a bridge, passing host bus transactions to the cards by sending commands and performing data accesses to/from the cards. It handles the SD, SDIO, MMC, and CE-ATA protocols at the transmission level.
I2S	The I^{2}S is a full-duplex, serial port that allows the chip to communicate with a variety of serial devices, such as standard codecs, digital signal processors (DSPs), microprocessors, peripherals, and audio codecs that implement the inter-IC sound bus (I^{2}S) and the Intel® AC97 standards

2.3.9 Human-machine interfaces

The following human-machine interfaces (HMI) are available on this device:

Table 2-10. HMI modules

Module	Description
General purpose input/output (GPIO)	All general purpose input or output (GPIO) pins are capable of interrupt and DMA request generation. All GPIO pins have 5 V tolerance.
Capacitive touch sense input (TSI)	Contains up to 16 channel inputs for capacitive touch sensing applications. Operation is available in low-power modes via interrupts.

2.4 Orderable part numbers

The following table summarizes the part numbers of the devices covered by this document.

Table 2-11. Orderable part numbers summary

Freescale part number	CPU frequency	Pin count	Package	Total flash memory	Program flash	EEPROM	SRAM	GPIO
MK20FX512VLQ12	120 MHz	144	LQFP	1 MB	512 KB	16 KB	128 KB	100
MK20FN1M0VLQ12	120 MHz	144	LQFP	1 MB	1 MB	—	128 KB	100
MK20FX512VMD12	120 MHz	144	MAPBGA	1 MB	512 KB	16 KB	128 KB	100
MK20FN1M0VMD12	120 MHz	144	MAPBGA	1 MB	1 MB	—	128 KB	100

Chapter 3
Chip Configuration

3.1 Introduction

This chapter provides details on the individual modules of the microcontroller. It includes:

- module block diagrams showing immediate connections within the device,
- specific module-to-module interactions not necessarily discussed in the individual module chapters, and
- links for more information.

3.2 Core modules

3.2.1 ARM Cortex-M4 Core Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by ARM and can be found at http://www.arm.com.

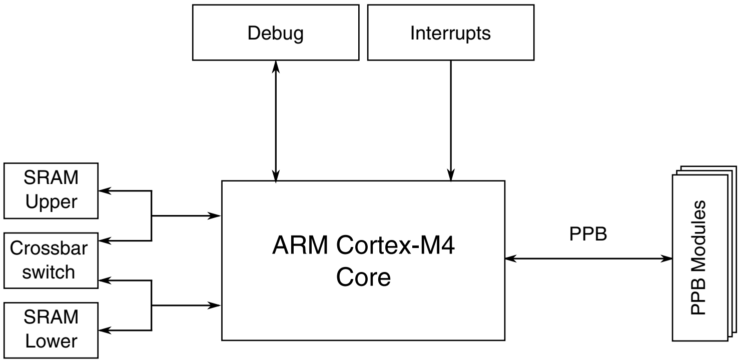


Figure 3-1. Core configuration

Table 3-1. Reference links to related information

Topic	Related module	Reference
Full description	ARM Cortex-M4 core, r0p1	http://www.arm.com
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
System/instruction/data bus module	Crossbar switch	Crossbar switch
System/instruction/data bus module	SRAM	SRAM
Debug	IEEE 1149.1 JTAG Serial Wire Debug (SWD) ARM Real-Time Trace Interface	Debug
Interrupts	Nested Vectored Interrupt Controller (NVIC)	NVIC
Private Peripheral Bus (PPB) module	Miscellaneous Control Module (MCM)	MCM
Private Peripheral Bus (PPB) module	Single-precision floating point unit (FPU)	FPU

3.2.1.1 Buses, interconnects, and interfaces

The ARM Cortex-M4 core has four buses as described in the following table.

Bus name	Description
Instruction code (ICODE) bus	The ICODE and DCODE buses are muxed. This muxed bus is called the CODE bus and is connected to the crossbar switch via a single master port. In addition, the CODE bus is also tightly coupled to the lower half of the system RAM (SRAM_L).
Data code (DCODE) bus	
System bus	The system bus is connected to a separate master port on the crossbar. In addition, the system bus is tightly coupled to the upper half system RAM (SRAM_U).
Private peripheral (PPB) bus	The PPB provides access to these modules: • ARM modules such as the NVIC, ETM, ITM, DWT, FBP, and ROM table • Freescale Miscellaneous Control Module (MCM)

3.2.1.2 System Tick Timer

The System Tick Timer's clock source is always the core clock, FCLK. This results in the following:

- The CLKSOURCE bit in SysTick Control and Status register is always set to select the core clock.
- Because the timing reference (FCLK) is a variable frequency, the TENMS bit in the SysTick Calibration Value Register is always zero.
- The NOREF bit in SysTick Calibration Value Register is always set, implying that FCLK is the only available source of reference timing.

3.2.1.3 Debug facilities

This device has extensive debug capabilities including run control and tracing capabilities. The standard ARM debug port that supports JTAG and SWD interfaces. Also the cJTAG interface is supported on this device.

3.2.1.4 Caches

This device includes two 8 KB of combined data/ instructions caches to minimize the perfromance impact of memory access latencies. One for the system bus and one for I/D bus.

Features of the cache are:

- 2-way set associative
- 4 word lines
- Lines can be individually flushed
- Entire cache can be flushed at once

3.2.1.4.1 Low Power

The caches are powered down in LLS and VLLSx modes. The caches need to be flushed before being powered down. This affects the time it takes for the system to go into the new mode but not the wakeup time.

3.2.1.4.2 Control

For control purposses the cache can be in one of these states:

1. Write Back \ Write Allocate (WBWA)
2. Write Through
3. No cache

For each defined region there will bit 2 bits allocated on the control register that determines the cache state for the memory region associated with this section. The user can only "lower" the cache attribute, given the fixed relationship of WBWA > WT > NC - so, you can demote a WBWA region to either WT or NC, you can demote a WT space to NC. In order to change the state upwards a system reset is required.

The 1Gb space allocated to the DRAM and Flexbus will be split in 16 MB regions and 2 bits assigned to the control state of each region. A total of 128 bits, distributed in 4 32-bit registers will be used for this control.

3.2.1.5 Core privilege levels

The ARM documentation uses different terms than this document to distinguish between privilege levels.

If you see this term...	it also means this term...
Privileged	Supervisor
Unprivileged or user	User

3.2.2 Nested Vectored Interrupt Controller (NVIC) Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by ARM and can be found at http://www.arm.com.

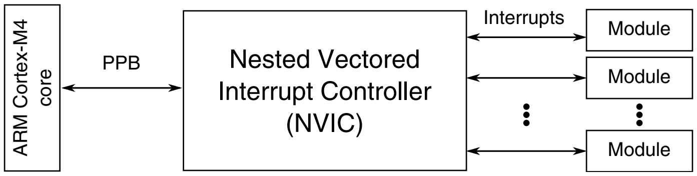


Figure 3-2. NVIC configuration

Table 3-2. Reference links to related information

Topic	Related module	Reference
Full description	Nested Vectored Interrupt Controller (NVIC)	http://www.arm.com
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Private Peripheral Bus (PPB)	ARM Cortex-M4 core	ARM Cortex-M4 core

3.2.2.1 Interrupt priority levels

This device supports 16 priority levels for interrupts. Therefore, in the NVIC each source in the IPR registers contains 4 bits. For example, IPR0 is shown below:

	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	IRQ3				0	0	0	0	IRQ2				0	0	0	0	IRQ1				0	0	0	0	IRQ0				0	0	0	0
W																																

3.2.2.2 Non-maskable interrupt

The non-maskable interrupt request to the NVIC is controlled by the external $\overline{\text{NMI}}$ signal. The pin the $\overline{\text{NMI}}$ signal is multiplexed on, must be configured for the $\overline{\text{NMI}}$ function to generate the non-maskable interrupt request.

3.2.2.3 Interrupt channel assignments

The interrupt source assignments are defined in the following table.

- Vector number — the value stored on the stack when an interrupt is serviced.
- IRQ number — non-core interrupt source count, which is the vector number minus 16.

The IRQ number is used within ARM's NVIC documentation.

Table 3-4. Interrupt vector assignments

Address	Vector	IRQ[1]	NVIC non-IPR register number[2]	NVIC IPR register number[3]	Source module	Source description
ARM Core System Handler Vectors						
0x0000_0000	0	–	–	–	ARM core	Initial Stack Pointer
0x0000_0004	1	–	–	–	ARM core	Initial Program Counter
0x0000_0008	2	–	–	–	ARM core	Non-maskable Interrupt (NMI)
0x0000_000C	3	–	–	–	ARM core	Hard Fault
0x0000_0010	4	–	–	–	ARM core	MemManage Fault
0x0000_0014	5	–	–	–	ARM core	Bus Fault
0x0000_0018	6	–	–	–	ARM core	Usage Fault
0x0000_001C	7	–	–	–	—	—
0x0000_0020	8	–	–	–	—	—
0x0000_0024	9	–	–	–	—	—
0x0000_0028	10	–	–	–	—	—
0x0000_002C	11	–	–	–	ARM core	Supervisor call (SVCall)
0x0000_0030	12	–	–	–	ARM core	Debug Monitor
0x0000_0034	13	–	–	–	—	—
0x0000_0038	14	–	–	–	ARM core	Pendable request for system service (PendableSrvReq)
0x0000_003C	15	–	–	–	ARM core	System tick timer (SysTick)
Non-Core Vectors						
0x0000_0040	16	0	0	0	DMA	DMA channel 0, 16 transfer complete
0x0000_0044	17	1	0	0	DMA	DMA channel 1, 17 transfer complete
0x0000_0048	18	2	0	0	DMA	DMA channel 2, 18 transfer complete
0x0000_004C	19	3	0	0	DMA	DMA channel 3, 19 transfer complete
0x0000_0050	20	4	0	1	DMA	DMA channel 4, 20 transfer complete
0x0000_0054	21	5	0	1	DMA	DMA channel 5, 21 transfer complete
0x0000_0058	22	6	0	1	DMA	DMA channel 6, 22 transfer complete
0x0000_005C	23	7	0	1	DMA	DMA channel 7, 23 transfer complete
0x0000_0060	24	8	0	2	DMA	DMA channel 8, 24 transfer complete

Table continues on the next page...

Table 3-4. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0064	25	9	0	2	DMA	DMA channel 9, 25 transfer complete .
0x0000_0068	26	10	0	2	DMA	DMA channel 10, 26 transfer complete
0x0000_006C	27	11	0	2	DMA	DMA channel 11, 27 transfer complete
0x0000_0070	28	12	0	3	DMA	DMA channel 12, 28 transfer complete
0x0000_0074	29	13	0	3	DMA	DMA channel 13, 29 transfer complete
0x0000_0078	30	14	0	3	DMA	DMA channel 14, 30 transfer complete
0x0000_007C	31	15	0	3	DMA	DMA channel 15, 31 transfer complete
0x0000_0080	32	16	0	4	DMA	DMA error interrupt channels 0-31
0x0000_0084	33	17	0	4	MCM	Normal interrupt
0x0000_0088	34	18	0	4	Flash memory	Command complete
0x0000_008C	35	19	0	4	Flash memory	Read collision
0x0000_0090	36	20	0	5	Mode Controller	Low-voltage detect, low-voltage warning
0x0000_0094	37	21	0	5	LLWU	Low Leakage Wakeup **NOTE:** The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery.
0x0000_0098	38	22	0	5	WDOG	Watchdog interrupt
0x0000_009C	39	23	0	5	—	—
0x0000_00A0	40	24	0	6	I^{2}C0	—
0x0000_00A4	41	25	0	6	I^{2}C1	—
0x0000_00A8	42	26	0	6	SPI0	Single interrupt vector for all sources
0x0000_00AC	43	27	0	6	SPI1	Single interrupt vector for all sources
0x0000_00B0	44	28	0	7	SPI2	Single interrupt vector for all sources
0x0000_00B4	45	29	0	7	CAN0	OR'ed Message buffer (0-15)
0x0000_00B8	46	30	0	7	CAN0	Bus Off
0x0000_00BC	47	31	0	7	CAN0	Error
0x0000_00C0	48	32	1	8	CAN0	Transmit Warning
0x0000_00C4	49	33	1	8	CAN0	Receive Warning
0x0000_00C8	50	34	1	8	CAN0	Wake Up
0x0000_00CC	51	35	1	8	I^{2}S0	Transmit
0x0000_00D0	52	36	1	9	I^{2}S0	Receive

Table continues on the next page...

Table 3-4. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_00D4	53	37	1	9	CAN1	OR'ed Message buffer (0-15)
0x0000_00D8	54	38	1	9	CAN1	Bus off
0x0000_00DC	55	39	1	9	CAN1	Error
0x0000_00E0	56	40	1	10	CAN1	Transmit Warning
0x0000_00E4	57	41	1	10	CAN1	Receive Warning
0x0000_00E8	58	42	1	10	CAN1	Wake Up
0x0000_00EC	59	43	1	10	—	—
0x0000_00F0	60	44	1	11	UART0	Single interrupt vector for UART LON sources
0x0000_00F4	61	45	1	11	UART0	Single interrupt vector for UART status sources
0x0000_00F8	62	46	1	11	UART0	Single interrupt vector for UART error sources
0x0000_00FC	63	47	1	11	UART1	Single interrupt vector for UART status sources
0x0000_0100	64	48	1	12	UART1	Single interrupt vector for UART error sources
0x0000_0104	65	49	1	12	UART2	Single interrupt vector for UART status sources
0x0000_0108	66	50	1	12	UART2	Single interrupt vector for UART error sources
0x0000_010C	67	51	1	12	UART3	Single interrupt vector for UART status sources
0x0000_0110	68	52	1	13	UART3	Single interrupt vector for UART error sources
0x0000_0114	69	53	1	13	UART4	Single interrupt vector for UART status sources
0x0000_0118	70	54	1	13	UART4	Single interrupt vector for UART error sources
0x0000_011C	71	55	1	13	UART5	Single interrupt vector for UART status sources
0x0000_0120	72	56	1	14	UART5	Single interrupt vector for UART error sources
0x0000_0124	73	57	1	14	ADC0	—
0x0000_0128	74	58	1	14	ADC1	—
0x0000_012C	75	59	1	14	CMP0	—
0x0000_0130	76	60	1	15	CMP1	—

Table continues on the next page...

Table 3-4. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0134	77	61	1	15	CMP2	—
0x0000_0138	78	62	1	15	FTM0	Single interrupt vector for all sources
0x0000_013C	79	63	1	15	FTM1	Single interrupt vector for all sources
0x0000_0140	80	64	2	16	FTM2	Single interrupt vector for all sources
0x0000_0144	81	65	2	16	CMT	—
0x0000_0148	82	66	2	16	RTC	Alarm interrupt
0x0000_014C	83	67	2	16	RTC	Seconds interrupt
0x0000_0150	84	68	2	17	PIT	Channel 0
0x0000_0154	85	69	2	17	PIT	Channel 1
0x0000_0158	86	70	2	17	PIT	Channel 2
0x0000_015C	87	71	2	17	PIT	Channel 3
0x0000_0160	88	72	2	18	PDB	—
0x0000_0164	89	73	2	18	USB OTG	—
0x0000_0168	90	74	2	18	USB Charger Detect	—
0x0000_016C	91	75	2	18	—	—
0x0000_0170	92	76	2	19	—	—
0x0000_0174	93	77	2	19	—	—
0x0000_0178	94	78	2	19	—	—
0x0000_017C	95	79	2	19	—	—
0x0000_0180	96	80	2	20	SDHC	—
0x0000_0184	97	81	2	20	DAC0	—
0x0000_0188	98	82	2	20	DAC1	—
0x0000_018C	99	83	2	20	TSI	Single interrupt vector for all sources
0x0000_0190	100	84	2	21	MCG	—
0x0000_0194	101	85	2	21	Low Power Timer	—
0x0000_0198	102	86	2	21	—	—
0x0000_019C	103	87	2	21	Port control module	Pin detect (Port A)
0x0000_01A0	104	88	2	22	Port control module	Pin detect (Port B)
0x0000_01A4	105	89	2	22	Port control module	Pin detect (Port C)

Table continues on the next page...

Table 3-4. Interrupt vector assignments (continued)

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_01A8	106	90	2	22	Port control module	Pin detect (Port D)
0x0000_01AC	107	91	2	22	Port control module	Pin detect (Port E)
0x0000_01B0	108	92	2	23	Port control module	Pin detect (Port F)
0x0000_01B4	109	93	2	23	—	—
0x0000_01B8	110	94	2	23	Software	Software interrupt[4]
0x0000_01BC	111	95	2	23	NAND flash controller (NFC)	—
0x0000_01C0	112	96	3	24	USB HS	—
0x0000_01C4	113	97	3	24		—
0x0000_01C8	114	98	3	24	CMP3	—
0x0000_01CC	115	99	3	24	—	—
0x0000_01D0	116	100	3	25	—	—
0x0000_01D4	117	101	3	25	FTM3	Single interrupt vector for all sources
0x0000_01D8	118	102	3	25	ADC2	—
0x0000_01DC	119	103	3	25	ADC3	—
0x0000_01E0	120	104	3	26	I^{2}S1	Transmit
0x0000_01E4	121	105	3	26	I^{2}S1	Receive

1. Indicates the NVIC's interrupt source number.
2. Indicates the NVIC's ISER, ICER, ISPR, ICPR, and IABR register number used for this IRQ. The equation to calculate this value is: IRQ div 32
3. Indicates the NVIC's IPR register number used for this IRQ. The equation to calculate this value is: IRQ div 4
4. This interrupt can only be pended or cleared via the NVIC registers.

3.2.2.3.1 Determining the bitfield and register location for configuring a particular interrupt

Suppose you need to configure the low-power timer (LPTMR) interrupt. The following table is an excerpt of the LPTMR row from Interrupt channel assignments.

Table 3-5. LPTMR interrupt vector assignment

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0194	101	85	2	21	Low Power Timer	—

1. Indicates the NVIC's interrupt source number.
2. Indicates the NVIC's ISER, ICER, ISPR, ICPR, and IABR register number used for this IRQ. The equation to calculate this value is: IRQ div 32
3. Indicates the NVIC's IPR register number used for this IRQ. The equation to calculate this value is: IRQ div 4

- The NVIC registers you would use to configure the interrupt are:
  - NVICISER2
  - NVICICER2
  - NVICISPR2
  - NVICICPR2
  - NVICIABR2
  - NVICIPR21
- To determine the particular IRQ's bitfield location within these particular registers:
  - NVICISER2, NVICICER2, NVICISPR2, NVICICPR2, NVICIABR2 bit location = IRQ mod 32 = 21
  - NVICIPR21 bitfield starting location = 8 * (IRQ mod 4) + 4 = 12

    Since the NVICIPR bitfields are 4-bit wide (16 priority levels), the NVICIPR21 bitfield range is 12-15

Therefore, the following bitfield locations are used to configure the LPTMR interrupts:

- NVICISER2[21]
- NVICICER2[21]
- NVICISPR2[21]
- NVICICPR2[21]
- NVICIABR2[21]
- NVICIPR21[15:12]

3.2.3 Asynchronous Wake-up Interrupt Controller (AWIC) Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by ARM and can be found at http://www.arm.com.

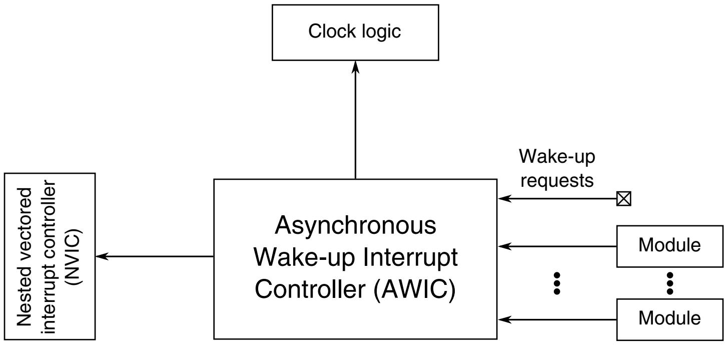

Clock logic
Wake-up requests
Nested vectored interrupt controller (NVIC)
Asynchronous Wake-up Interrupt Controller (AWIC)
Module
Module

Figure 3-3. Asynchronous Wake-up Interrupt Controller configuration

Table 3-6. Reference links to related information

Topic	Related module	Reference
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
	Nested Vectored Interrupt Controller (NVIC)	NVIC
Wake-up requests		AWIC wake-up sources

3.2.3.1 Wake-up sources

The device uses the following internal and external inputs to the AWIC module.

Table 3-7. AWIC Stop and VLPS Wake-up Sources

Wake-up source	Description
Available system resets	$\overline{\text{RESET}}$ pin and WDOG when LPO is its clock source, and JTAG
Low-voltage detect	Mode Controller
Low-voltage warning	Mode Controller
Pin interrupts	Port Control Module - Any enabled pin interrupt is capable of waking the system
ADCx	The ADC is functional when using internal clock source
CMPx	Since no system clocks are available, functionality is limited
I^2C	Address match wakeup
UART	Active edge on RXD
USB	Wakeup
LPTMR	Functional in Stop/VLPS modes
RTC	Functional in Stop/VLPS modes
SDHC	Wakeup

Table continues on the next page...

Table 3-7. AWIC Stop and VLPS Wake-up Sources (continued)

Wake-up source	Description
I2S	Functional when using an external bit clock or external master clock
TSI	
CAN	

3.2.4 FPU Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

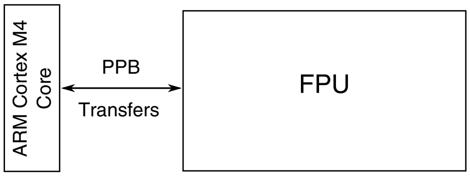


Figure 3-4. FPU configuration

Table 3-8. Reference links to related information

Topic	Related module	Reference
Full description	FPU	http://www.arm.com
System memory map		System memory map
Clocking		Clock Distribution
Power Management		Power Management
Transfers Private Peripheral Bus (PPB)	ARM Cortex M4 core	/projects/Microcontrollers/devices/Pioneer/topics/core/core-diagram-table.xml

3.2.5 JTAG Controller Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

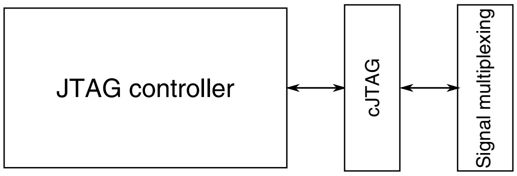


Figure 3-5. JTAGC Controller configuration

Table 3-9. Reference links to related information

Topic	Related module	Reference
Full description	JTAGC	JTAGC
Signal multiplexing	Port control	Signal multiplexing

3.3 System modules

3.3.1 SIM Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

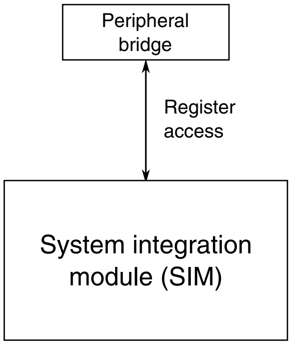


Figure 3-6. SIM configuration

Table 3-10. 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

3.3.2 System Mode Controller (SMC) Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

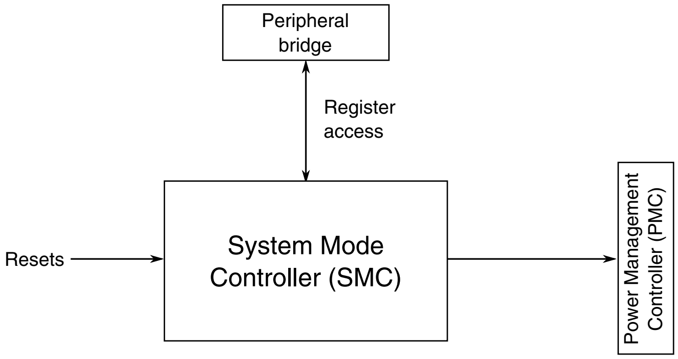

Peripheral bridge
Register access
Resets
System Mode Controller (SMC)
Power Management Controller (PMC)

Figure 3-7. System Mode Controller configuration

Table 3-11. Reference links to related information

Topic	Related module	Reference
Full description	System Mode Controller (SMC)	SMC
System memory map		System memory map
Power management		Power management
	Power management controller (PMC)	PMC
	Low-Leakage Wakeup Unit (LLWU)	LLWU
	Reset Control Module (RCM)	Reset

3.3.3 PMC Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

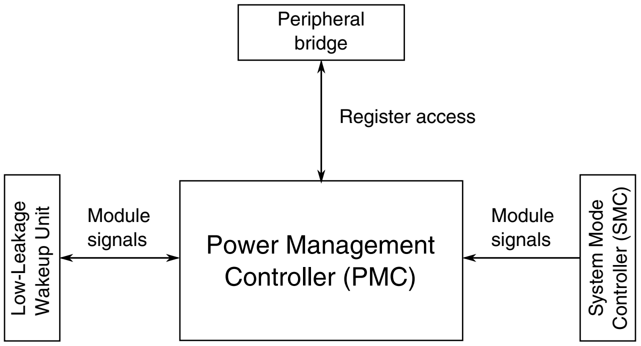


Figure 3-8. PMC configuration

Table 3-12. Reference links to related information

Topic	Related module	Reference
Full description	PMC	PMC
System memory map		System memory map
Power management		Power management
Full description	System Mode Controller (SMC)	System Mode Controller
	Low-Leakage Wakeup Unit (LLWU)	LLWU
	Reset Control Module (RCM)	Reset

3.3.4 Low-Leakage Wake-up Unit (LLWU) Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

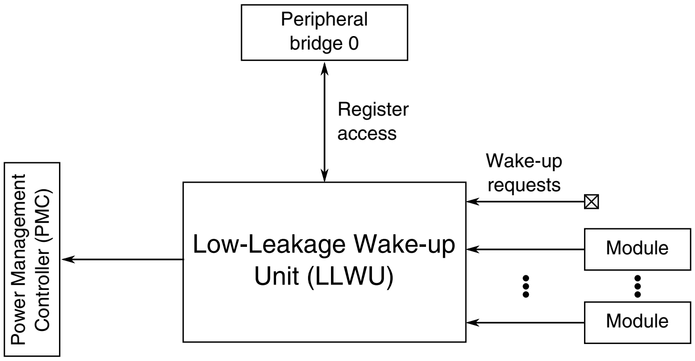

Peripheral bridge 0
Register access
Wake-up requests
Power Management Controller (PMC)
Low-Leakage Wake-up Unit (LLWU)
Module
Module

Figure 3-9. Low-Leakage Wake-up Unit configuration

Table 3-13. 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 chapter
	Power Management Controller (PMC)	Power Management Controller (PMC)
	Mode Controller	Mode Controller
Wake-up requests		LLWU wake-up sources

3.3.4.1 Wake-up Sources

This chip uses the following internal peripheral and external pin inputs as wakeup sources to the LLWU module:

- LLWU_P0-15 are external pin inputs. See the chip's signal multiplexing table for the individual input signal options.
- LLWU_M0IF-M7IF are connections to the internal peripheral interrupt flags.

NOTE

$\overline{\text{RESET}}$ is also a wakeup source, depending on the bit setting in the LLWU_RST register. On devices where $\overline{\text{RESET}}$ is not a dedicated pin, it must also be enabled in the explicit port mux control.

Table 3-14. Wakeup sources for LLWU inputs

Input	Wakeup source		Input	Wakeup source
LLWU_P0	PTE1/LLWU_P0 pin		LLWU_P12	PTD0/LLWU_P12 pin

Table continues on the next page...

Table 3-14. Wakeup sources for LLWU inputs (continued)

Input	Wakeup source		Input	Wakeup source
LLWU_P1	PTE2/LLWU_P1 pin		LLWU_P13	PTD2/LLWU_P13 pin
LLWU_P2	PTE4/LLWU_P2 pin		LLWU_P14	PTD4/LLWU_P14 pin
LLWU_P3	PTA4/LLWU_P3 pin[1]		LLWU_P15	PTD6/LLWU_P15 pin
LLWU_P4	PTA13/LLWU_P4 pin		LLWU_M0IF	LPTMR[2]
LLWU_P5	PTB0/LLWU_P5 pin		LLWU_M1IF	CMP0[2]
LLWU_P6	PTC1/LLWU_P6 pin		LLWU_M2IF	CMP1[2]
LLWU_P7	PTC3/LLWU_P7 pin		LLWU_M3IF	CMP2/CMP3[2]
LLWU_P8	PTC4/LLWU_P8 pin		LLWU_M4IF	TSI[2]
LLWU_P9	PTC5/LLWU_P9 pin		LLWU_M5IF	RTC Alarm[2]
LLWU_P10	PTC6/LLWU_P10 pin		LLWU_M6IF	Reserved
LLWU_P11	PTC11/LLWU_P11 pin		LLWU_M7IF	RTC Seconds[2]

1. The $\overline{\text{EZP\_CS}}$ signal is checked only on *Chip Reset not VLLS*, so a VLLS wakeup via a non-reset source does not cause EzPort mode entry. If NMI was enabled on entry to LLS/VLLS, asserting the NMI pin generates an NMI interrupt on exit from the low power mode.
2. Requires the peripheral and the peripheral interrupt to be enabled. The LLWU's WUME bit enables the internal module flag as a wakeup input. After wakeup, the flags are cleared based on the peripheral clearing mechanism.

3.3.5 MCM Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

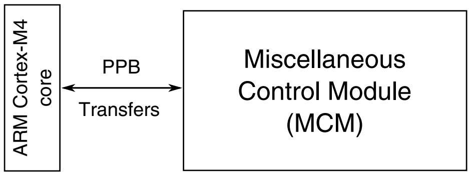


Figure 3-10. MCM configuration

Table 3-15. Reference links to related information

Topic	Related module	Reference
Full description	Miscellaneous control module (MCM)	MCM
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Transfers Private Peripheral Bus (PPB)	ARM Cortex-M4 core	ARM Cortex-M4 core

3.3.6 Crossbar Switch Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

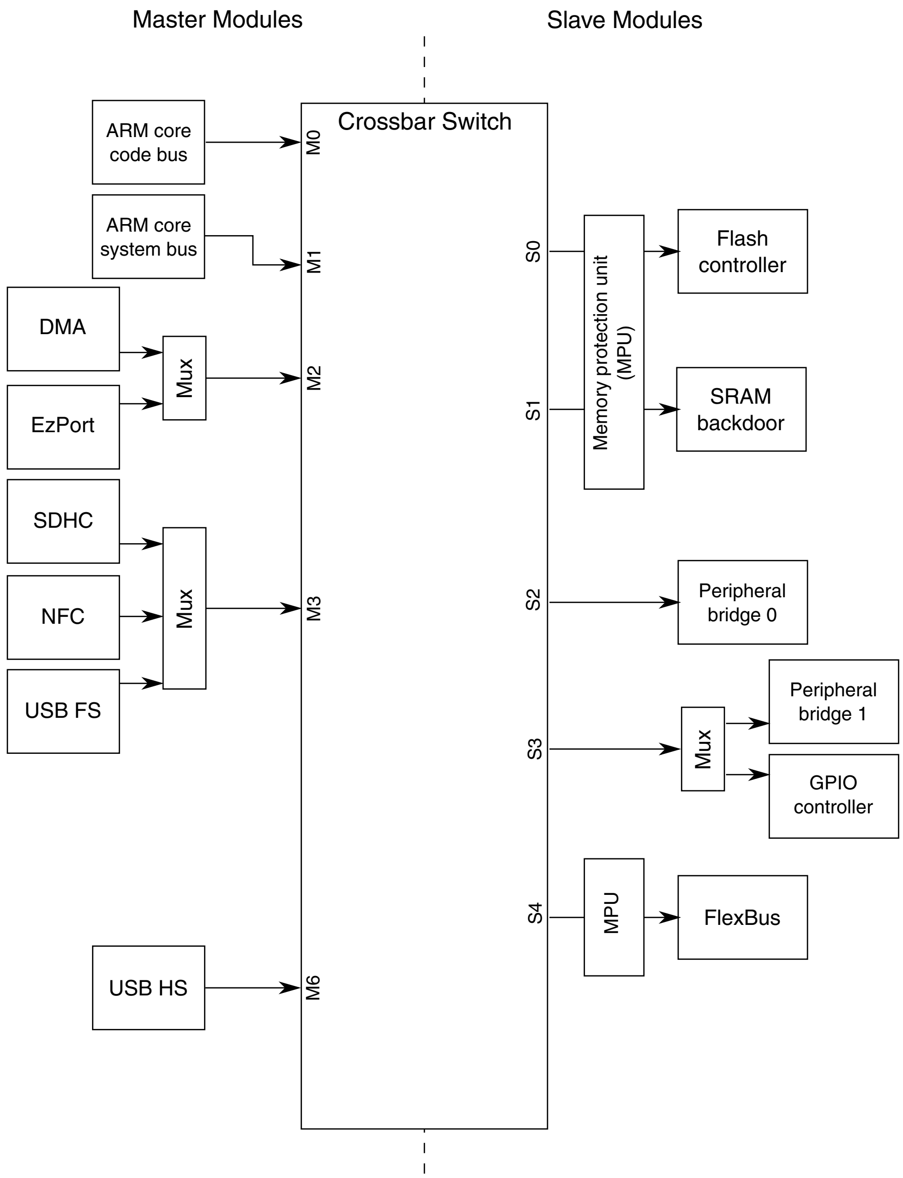


Figure 3-11. Crossbar switch integration

Table 3-16. Reference links to related information

Topic	Related module	Reference
Full description	Crossbar switch	Crossbar Switch
System memory map		System memory map

Table continues on the next page...

Table 3-16. Reference links to related information (continued)

Topic	Related module	Reference
Clocking		Clock Distribution
Memory protection	MPU	MPU
Crossbar switch master	ARM Cortex-M4 core	ARM Cortex-M4 core
Crossbar switch master	DMA controller	DMA controller
Crossbar switch master	EzPort	EzPort
Crossbar switch master	USB FS/LS	USB FS/LS
Crossbar switch master	USB HS/FS/LS	USB HS/FS/LS
Crossbar switch master	SDHC	SDHC
Crossbar switch master	NFC	NFC
Crossbar switch slave	Flash	Flash
Crossbar switch slave	SRAM backdoor	SRAM backdoor
Crossbar switch slave	Peripheral bridges	Peripheral bridge
Crossbar switch slave	GPIO controller	GPIO controller
Crossbar switch slave	FlexBus	FlexBus

3.3.6.1 Crossbar Switch Master Assignments

The masters connected to the crossbar switch are assigned as follows:

Master module	Master port number
ARM core code bus	0
ARM core system bus	1
DMA/EzPort	2
SDHC/NFC/USB FS	3
USB HS	6

NOTE

The DMA and EzPort share a master port. Since these modules never operate at the same time, no configuration or arbitration explanations are necessary.

3.3.6.2 Crossbar Switch Slave Assignments

The slaves connected to the crossbar switch are assigned as follows:

Slave module	Slave port number	Protected by MPU?
Flash memory controller	0	Yes
SRAM backdoor	1	Yes
Peripheral bridge 0[1]	2	No. Protection built into bridge.
Peripheral bridge 1/GPIO[1]	3	No. Protection built into bridge.
FlexBus	4	Yes

1. See System memory map for access restrictions.

3.3.6.3 PRS register reset values

The AXBS_PRS*n* registers reset to 7654_3210h.

3.3.7 Memory Protection Unit (MPU) Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

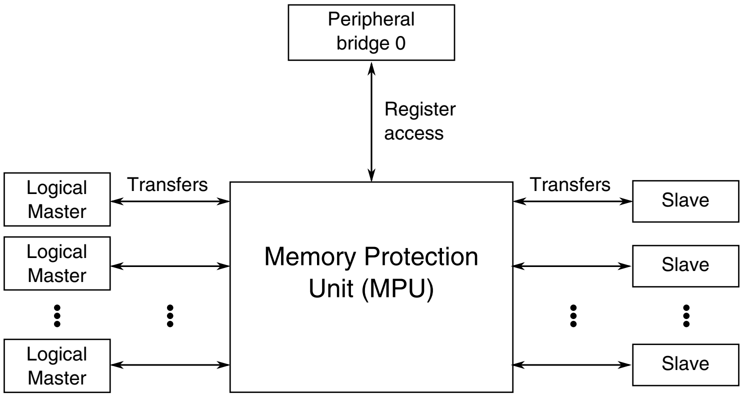


Figure 3-12. Memory Protection Unit configuration

Table 3-17. Reference links to related information

Topic	Related module	Reference
Full description	Memory Protection Unit (MPU)	MPU
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Logical masters		Logical master assignments
Slave modules		Slave module assignments

3.3.7.1 MPU Slave Port Assignments

The memory-mapped resources protected by the MPU are:

Table 3-18. MPU Slave Port Assignments

Source	MPU Slave Port Assignment	Destination
Crossbar slave port 0	MPU slave port 0	Flash Controller
Crossbar slave port 1	MPU slave port 1	SRAM backdoor
Code Bus	MPU slave port 2	SRAM_L frontdoor
System Bus	MPU slave port 3	SRAM_U frontdoor
Crossbar slave port 4	MPU slave port 4	FlexBus

3.3.7.2 MPU Logical Bus Master Assignments

The logical bus master assignments for the MPU are:

Table 3-19. MPU Logical Bus Master Assignments

MPU Logical Bus Master Number	Bus Master
0	Core
1	Debugger
2	DMA
3	none
4	USB
5	SDHC
6	none
7	none

3.3.7.3 MPU Access Violation Indications

Access violations detected by the MPU are signaled to the appropriate bus master as shown below:

Table 3-20. Access Violation Indications

Bus Master	Core Indication
Core	Bus fault (interrupt vector #5) Note: To enable bus faults set the core's System Handler Control and State Register's BUSFAULTENA bit. If this bit is not set, MPU violations result in a hard fault (interrupt vector #3).

Table continues on the next page...

Table 3-20. Access Violation Indications (continued)

Bus Master	Core Indication
Debugger	The STICKYERROR flag is set in the Debug Port Control/Status Register.
DMA	Interrupt vector #32
USB_OTG	Interrupt vector #89
SDHC	Interrupt vector #96

3.3.7.4 Reset Values for RGD0 Registers

At reset, the MPU is enabled with a single region descriptor (RGD0) that maps the entire 4 GB address space with read, write and execute permissions given to the core, debugger and the DMA bus masters.

The following table shows the chip-specific reset values for RGD0 and RGDAAC0.

Table 3-21. Reset Values for RGD0 Registers

Register	Reset value
RGD0_WORD0	0000_0000h
RGD0_WORD1	FFFF_FFFFh
RGD0_WORD2	0061_F7DFh
RGD0_WORD3	0000_0001h
RGDAAC0	0061_F7DFh

3.3.7.5 Write Access Restrictions for RGD0 Registers

In addition to configuring the initial state of RGD0, the MPU implements further access control on writes to the RGD0 registers. Specifically, the MPU assigns a priority scheme where the debugger is treated as the highest priority master followed by the core and then all the remaining masters.

The MPU does not allow writes from the core to affect the RGD0 start or end addresses nor the permissions associated with the debugger; it can only write the permission fields associated with the other masters.

These protections (summarized below) guarantee that the debugger always has access to the entire address space and those rights cannot be changed by the core or any other bus master.

Table 3-22. Write Access to RGD0 Registers

Bus Master	Write Access?
Core	Partial. The Core cannot write to the following registers or register fields: • RGD0_WORD0, RGD0_WORD1, RGD0_WORD3 • RGD0_WORD2[M1SM, M1UM] • RGDAAC0[M1SM, M1UM] **NOTE:** Changes to the RGD0_WORD2 alterable fields should be done via a write to RGDAAC0.
Debugger	Yes
All other masters	No

3.3.8 Peripheral Bridge Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

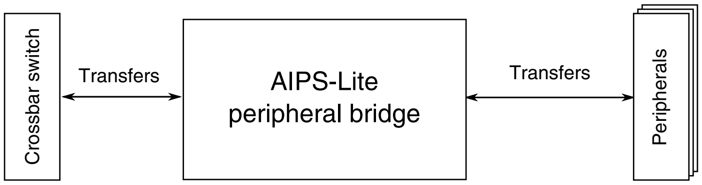


Figure 3-13. Peripheral bridge configuration

Table 3-23. Reference links to related information

Topic	Related module	Reference
Full description	Peripheral bridge (AIPS-Lite)	Peripheral bridge (AIPS-Lite)
System memory map		System memory map
Clocking		Clock Distribution
Crossbar switch	Crossbar switch	Crossbar switch

3.3.8.1 Number of peripheral bridges

This device contains two identical peripheral bridges.

3.3.8.2 Memory maps

The peripheral bridges are used to access the registers of most of the modules on this device. See AIPS0 Memory Map and AIPS1 Memory Map for the memory slot assignment for each module.

3.3.8.3 MPRA register

Each of the two peripheral bridges supports up to 8 crossbar switch masters, each assigned to a MPROTx field in the MPRA register. However, fewer are supported on this device. See Crossbar switch for details of the master port assignments for this device.

3.3.8.4 AIPS_Lite MPRA register reset value

- AIPS*x*_MPRA reset value is 0x7770_0000

Therefore, masters 0, 1, and 2 are trusted bus masters after reset.

3.3.8.5 PACR registers

Each of the two peripheral bridges support up to 128 peripherals each assigned to an PACRx field within the PACRA-PACRP registers. However, fewer peripherals are supported on this device. See AIPS0 Memory MapandAIPS1 Memory Map for details of the peripheral slot assignments for this device. Unused PACRx fields are reserved.

3.3.8.6 AIPS_Lite PACRE-P register reset values

The AIPS*x*_PACRE-P reset values depend on if the module is available on your particular device. For each populated slot in slots 32-127 in Peripheral Bridge 0 (AIPS-Lite 0) Memory Map and Peripheral Bridge 1 (AIPS-Lite 1) Memory Map, the corresponding module's PACR[32:127] field resets to 0x4.

3.3.9 DMA request multiplexer configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

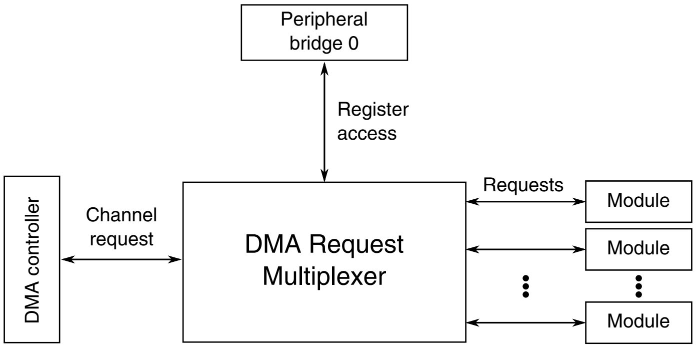


Figure 3-14. DMA request multiplexer configuration

Table 3-24. Reference links to related information

Topic	Related module	Reference
Full description	DMA request multiplexer	DMA Mux
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Channel request	DMA controller	DMA Controller
Requests		DMA request sources

3.3.9.1 DMA MUX request sources

This device includes two DMA request MUXes that allows up to 126 DMA request signals to be mapped to any of the 32 DMA channels. The first MUX is connected to channels 0-15 and the second MUX is connected to channels 16-31. To allow for flexibility and optimal usage of the available DMA channels some of the DMA request sources are available on both muxes.

Because of the mux there is not a hard correlation between any of the DMA request sources and a specific DMA channel.

Table 3-25. DMA request sources - MUX 0

Source number	Source module	Source description
0	—	Channel disabled[1]
1	Reserved	Not used
2	UART0	Receive
3	UART0	Transmit
4	UART1	Receive

Table continues on the next page...

Table 3-25. DMA request sources - MUX 0 (continued)

Source number	Source module	Source description
5	UART1	Transmit
6	UART2	Receive
7	UART2	Transmit
8	UART3	Receive
9	UART3	Transmit
10	UART4	Receive
11	UART4	Transmit
12	UART5	Receive
13	UART5	Transmit
14	I^{2}S0	Receive
15	I^{2}S0	Transmit
16	SPI0	Receive
17	SPI0	Transmit
18	SPI1	Receive
19	SPI1	Transmit
20	SPI2	Receive
21	SPI2	Transmit
22	I^{2}C0	—
23	I^{2}C1 or I^{2}C2	—
24	FTM0	Channel 0
25	FTM0	Channel 1
26	FTM0	Channel 2
27	FTM0	Channel 3
28	FTM0	Channel 4
29	FTM0	Channel 5
30	FTM0	Channel 6
31	FTM0	Channel 7
32	FTM1	Channel 0
33	FTM1	Channel 1
34	FTM2	Channel 0
35	FTM2	Channel 1
36	IEEE 1588 Timers	Timer 0
37	IEEE 1588 Timers	Timer 1
38	IEEE 1588 Timers	Timer 2

Table continues on the next page...

Table 3-25. DMA request sources - MUX 0 (continued)

Source number	Source module	Source description
39	IEEE 1588 Timers	Timer 3
40	ADC0	—
41	ADC1	—
42	CMP0	—
43	CMP1	—
44	CMP2	—
45	DAC0	—
46	DAC1	—
47	CMT	—
48	PDB	—
49	Port control module	Port A
50	Port control module	Port B
51	Port control module	Port C
52	Port control module	Port D
53	Port control module	Port E
54	DMA MUX	Always enabled
55	DMA MUX	Always enabled
56	DMA MUX	Always enabled
57	DMA MUX	Always enabled
58	DMA MUX	Always enabled
59	DMA MUX	Always enabled
60	DMA MUX	Always enabled
61	DMA MUX	Always enabled
62	DMA MUX	Always enabled
63	DMA MUX	Always enabled

1. Configuring a DMA channel to select source 0 or any of the reserved sources disables that DMA channel.

Table 3-26. DMA Request Sources - Mux 1

Source number	Source module	Source description
0	—	Channel disabled[1]
1	Reserved	Not used
2	UART0	Receive
3	UART0	Transmit
4	UART1	Receive

Table continues on the next page...

Table 3-26. DMA Request Sources - Mux 1 (continued)

Source number	Source module	Source description
5	UART1	Transmit
6	UART2	Receive
7	UART2	Transmit
8	UART3	Receive
9	UART3	Transmit
10	UART4	Receive
11	UART4	Transmit
12	UART5	Receive
13	UART5	Transmit
14	I^{2}S1	Receive
15	I^{2}S1	Transmit
16	SPI0	Receive
17	SPI0	Transmit
18	SPI1	Receive
19	SPI1	Transmit
20	SPI2	Receive
21	SPI2	Transmit
22	Reserved	—
23	Reserved	—
24	FTM3	Channel 0
25	FTM3	Channel 1
26	FTM3	Channel 2
27	FTM3	Channel 3
28	FTM3	Channel 4
29	FTM3	Channel 5
30	FTM3	Channel 6
31	FTM3	Channel 7
32	Reserved	—
33	Reserved	—
34	Reserved	—
35	Reserved	—
36	Reserved	—
37	Reserved	—
38	Reserved	—

Table continues on the next page...

Table 3-26. DMA Request Sources - Mux 1 (continued)

Source number	Source module	Source description
39	Reserved	—
40	ADC0	—
41	ADC1	—
42	ADC2	—
43	ADC3	—
44	Reserved	—
45	DAC0	—
46	DAC1	—
47	CMP0	—
48	CMP1	—
49	CMP2	—
50	CMP3	—
51	Reserved	—
52	Reserved	—
53	Port control module	Port F
54	DMA MUX	Always enabled
55	DMA MUX	Always enabled
56	DMA MUX	Always enabled
57	DMA MUX	Always enabled
58	DMA MUX	Always enabled
59	DMA MUX	Always enabled
60	DMA MUX	Always enabled
61	DMA MUX	Always enabled
62	DMA MUX	Always enabled
63	DMA MUX	Always enabled

1. Configuring a DMA channel to select source 0 or any of the reserved sources disables that DMA channel.

3.3.9.2 DMA transfers via PIT trigger

The PIT module can trigger a DMA transfer on the first four DMA channels. The assignments are detailed at PIT/DMA Periodic Trigger Assignments .

3.3.10 DMA Controller Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

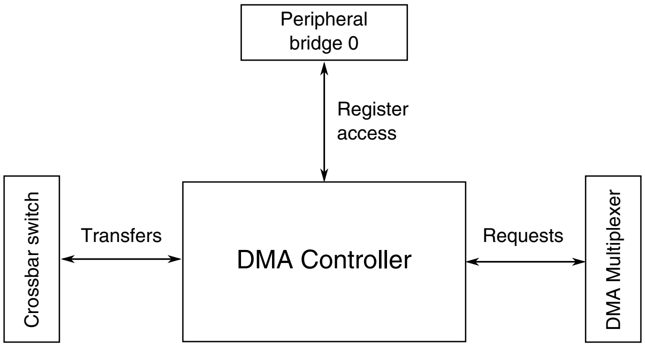


Figure 3-15. DMA Controller configuration

Table 3-27. Reference links to related information

Topic	Related module	Reference
Full description	DMA Controller	DMA Controller
System memory map		System memory map
Register access	Peripheral bridge (AIPS-Lite 0)	AIPS-Lite 0
Clocking		Clock distribution
Power management		Power management
Transfers	Crossbar switch	Crossbar switch

3.3.11 External Watchdog Monitor (EWM) Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

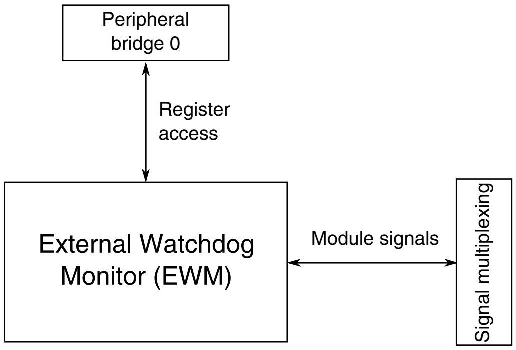


Figure 3-16. External Watchdog Monitor configuration

Table 3-28. Reference links to related information

Topic	Related module	Reference
Full description	External Watchdog Monitor (EWM)	EWM
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port Control Module	Signal multiplexing

3.3.11.1 EWM clocks

This table shows the EWM clocks and the corresponding chip clocks.

Table 3-29. EWM clock connections

Module clock	Chip clock
Low Power Clock	1 kHz LPO Clock

3.3.11.2 EWM low-power modes

This table shows the EWM low-power modes and the corresponding chip low-power modes.

Table 3-30. EWM low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS, LLS
Power Down	VLLS3, VLLS2, VLLS1

3.3.11.3 $\overline{\text{EWM\_OUT}}$ pin state in low power modes

During Wait, Stop and Power Down modes the $\overline{\text{EWM\_OUT}}$ 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.

3.3.12 Watchdog Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

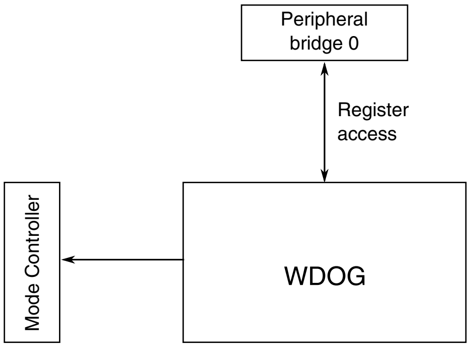


Figure 3-17. Watchdog configuration

Table 3-31. Reference links to related information

Topic	Related module	Reference
Full description	Watchdog	Watchdog
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
	Mode Controller (MC)	System Mode Controller

3.3.12.1 WDOG clocks

This table shows the WDOG module clocks and the corresponding chip clocks.

Table 3-32. WDOG clock connections

Module clock	Chip clock
LPO Oscillator	1 kHz LPO Clock
Alt Clock	Bus Clock
Fast Test Clock	Bus Clock
System Bus Clock	Bus Clock

3.3.12.2 WDOG low-power modes

This table shows the WDOG low-power modes and the corresponding chip low-power modes.

Table 3-33. WDOG low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Stop	Stop, VLPS
Power Down	LLS, VLLSx

3.4 Clock Modules

3.4.1 MCG Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

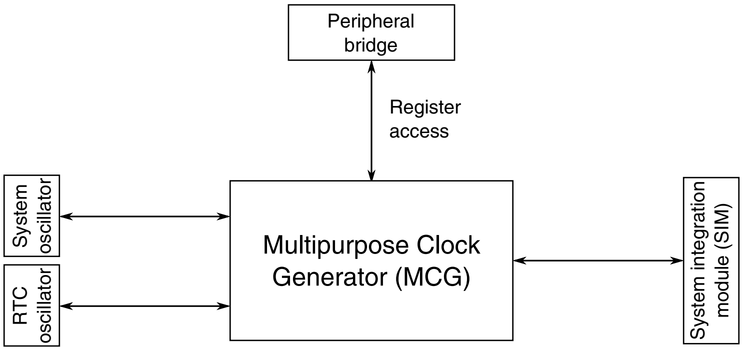

Peripheral bridge
Register access
System oscillator
RTC oscillator
Multipurpose Clock Generator (MCG)
System integration module (SIM)

Figure 3-18. MCG configuration

Table 3-34. Reference links to related information

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

3.4.2 OSCx Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

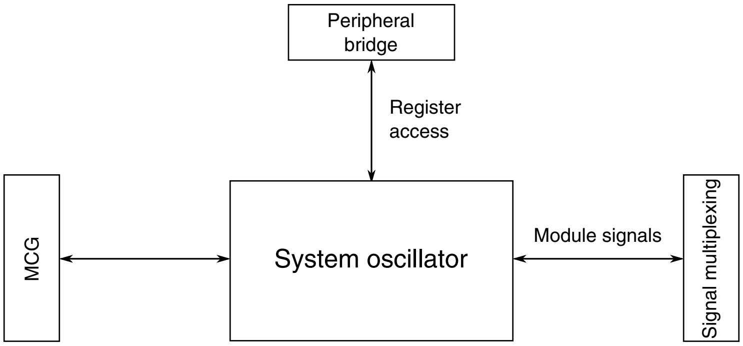


Figure 3-19. OSCx configuration

Table 3-35. Reference links to related information

Topic	Related module	Reference
Full description	OSCx	OSCx
System memory map		System memory map
Clocking		Clock distribution

Table continues on the next page...

Table 3-35. Reference links to related information (continued)

Topic	Related module	Reference
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing
Full description	MCG	MCG

3.4.2.1 OSC modes of operation with MCG

The MCG's C2 register bits configure the oscillator frequency range. See the OSC and MCG chapters for more details.

3.4.3 RTC OSC configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

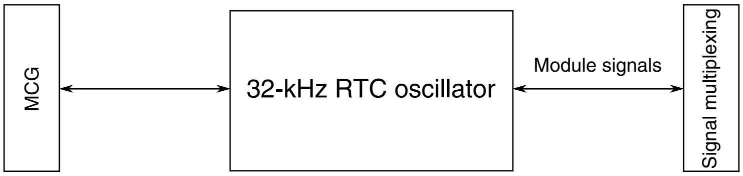


Figure 3-20. RTC OSC configuration

Table 3-36. Reference links to related information

Topic	Related module	Reference
Full description	RTC OSC	RTC OSC
Signal multiplexing	Port control	Signal multiplexing
Full description	MCG	MCG

3.5 Memories and Memory Interfaces

3.5.1 Flash Memory Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

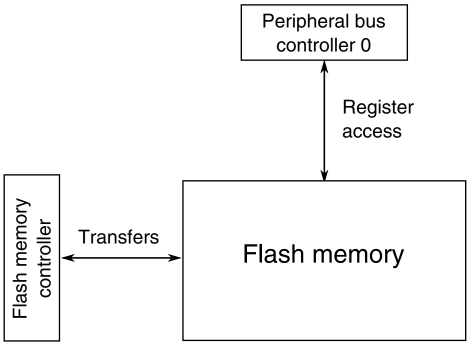


Figure 3-21. Flash memory configuration

Table 3-37. Reference links to related information

Topic	Related module	Reference
Full description	Flash memory	Flash memory
System memory map		System memory map
Clocking		Clock Distribution
Transfers	Flash memory controller	Flash memory controller
Register access	Peripheral bridge	Peripheral bridge

3.5.1.1 Flash memory types

This device contains the following types of flash memory:

- Program flash memory — non-volatile flash memory that can execute program code
- FlexMemory — encompasses the following memory types:
  - For devices with FlexNVM: FlexNVM — Non-volatile flash memory that can execute program code, store data, or backup EEPROM data
  - For devices with FlexNVM: FlexRAM — RAM memory that can be used as traditional RAM or as high-endurance EEPROM storage, and also accelerates flash programming
  - For devices with only program flash memory: Programming acceleration RAM — RAM memory that accelerates flash programming

3.5.1.2 Flash Memory Sizes

The devices covered in this document contain:

- For devices with program flash only: 4 blocks of program flash consisting of 4 KB sectors

- For devices that contain FlexNVM: 2 blocks of program flash consisting of 4 KB sectors
- For devices that contain FlexNVM: 2 blocks of FlexNVM consisting of 4 KB sectors
- For devices that contain FlexNVM: 1 block of FlexRAM

The amounts of flash memory for the devices covered in this document are:

Device	Program flash (KB)	Block 0 (P-Flash) address range[1]	FlexNVM (KB)	Block 1 (FlexNVM/ P-Flash) address range[1]	FlexRAM (KB)	FlexRAM address range
MK20FX512VLQ12	512	0x0000_0000 – 0x0007_FFFF	512	0x1000_0000 – 0x1007_FFFF	16	0x1400_0000 – 0x1400_3FFF
MK20FN1M0VLQ12	1024	0x0000_0000 – 0x0007_FFFF	—	0x0008_0000 – 0x000F_FFFF	—	N/A
MK20FX512VMD12	512	0x0000_0000 – 0x0007_FFFF	512	0x1000_0000 – 0x1007_FFFF	16	0x1400_0000 – 0x1400_3FFF
MK20FN1M0VMD12	1024	0x0000_0000 – 0x0007_FFFF	—	0x0008_0000 – 0x000F_FFFF	—	N/A

[1] For program flash only devices: The addresses shown assume program flash swap is disabled (default configuration).

3.5.1.3 Flash Memory Size Considerations

Since this document covers devices that contain program flash only and devices that contain program flash and FlexNVM, there are some items to consider when reading the flash memory chapter.

- The flash memory chapter shows a mixture of information depending on the device you are using.
- For the program flash only devices:
  - Two program flash blocks are supported: program flash 1 and program flash 2. The two blocks are contiguous in the system memory map.
  - The program flash blocks support a swap feature in which the starting address of the program flash blocks can be swapped.
  - The FlexRAM is not available as EEPROM or traditional RAM. Its space is only used for programming acceleration through the Program Section command.
- For the devices containing program flash and FlexNVM:
  - Since there is only one program flash block, the program flash swap feature is not available.

3.5.1.4 Flash Memory Map

The various flash memories and the flash registers are located at different base addresses as shown in the following figure. The base address for each is specified in System memory map.

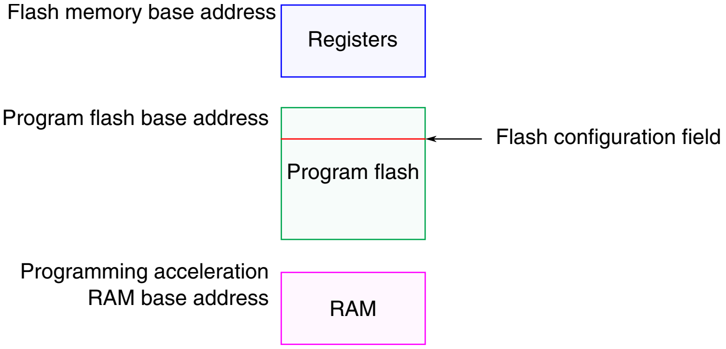

Flash memory base address
Registers
Program flash base address
Flash configuration field
Program flash
Programming acceleration RAM base address
RAM

Figure 3-22. Flash memory map for devices containing only program flash

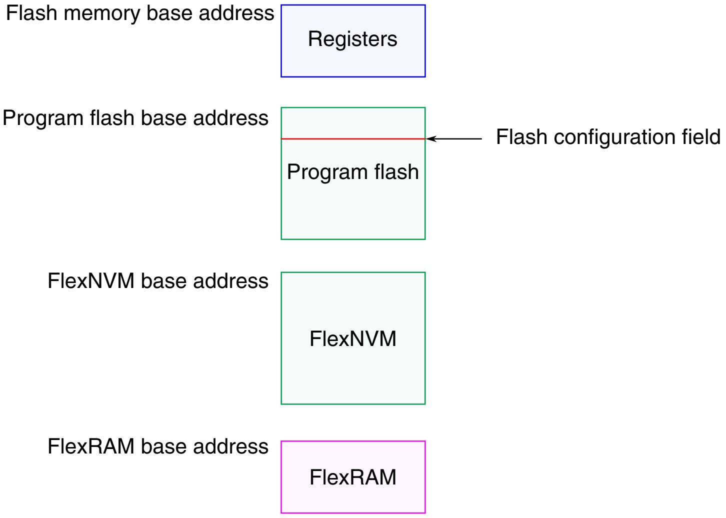

Flash memory base address
Registers
Program flash base address
Flash configuration field
Program flash
FlexNVM base address
FlexNVM
FlexRAM base address
FlexRAM

Figure 3-23. Flash memory map for devices containing FlexNVM

3.5.1.5 Flash Security

How flash security is implemented on this device is described in Chip Security.

3.5.1.6 Flash Modes

The flash memory operates in NVM normal and NVM special modes. The flash memory enters NVM special mode when the EzPort is enabled ($\overline{\text{EZP\_CS}}$ asserted during reset), or the system is under debug mode. Otherwise, flash memory operates in NVM normal mode.

3.5.1.7 Erase All Flash Contents

In addition to software, the entire flash memory may be erased external to the flash memory in two ways:

1. Via the EzPort by issuing a bulk erase (BE) command. See the EzPort chapter for more details.
2. Via the SWJ-DP debug port by setting DAP_CONTROL[0]. DAP_STATUS[0] is set to indicate the mass erase command has been accepted. DAP_STATUS[0] is cleared when the mass erase completes.

3.5.1.8 FTFE_FOPT Register

The flash memory's FTFE_FOPT register allows the user to customize the operation of the MCU at boot time. See FOPT boot options for details of its definition.

3.5.2 Flash Memory Controller Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

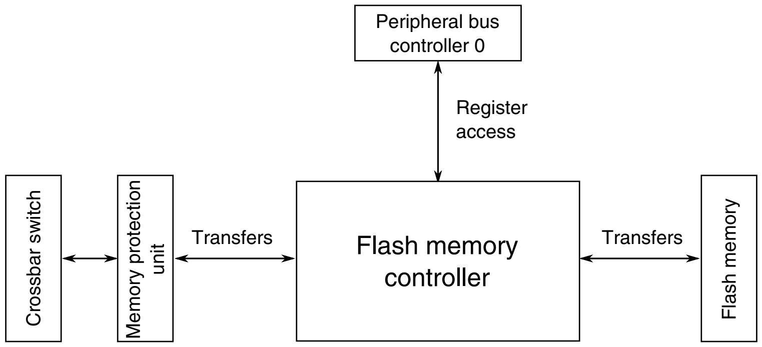


Figure 3-24. Flash memory controller configuration

Table 3-38. Reference links to related information

Topic	Related module	Reference
Full description	Flash memory controller	Flash memory controller
System memory map		System memory map
Clocking		Clock Distribution
Transfers	Flash memory	Flash memory

Table continues on the next page...

Table 3-38. Reference links to related information (continued)

Topic	Related module	Reference
Transfers	MPU	MPU
Transfers	Crossbar switch	Crossbar Switch
Register access	Peripheral bridge	Peripheral bridge

3.5.2.1 Number of masters

The Flash Memory Controller supports up to eight crossbar switch masters. However, this device has a different number of crossbar switch masters. See Crossbar Switch Configuration for details on the master port assignments.

3.5.2.2 Program Flash Swap

On devices that contain program flash memory only, the program flash memory blocks may swap their base addresses.

While not using swap:

- FMC_PFB01CR controls the lower code addresses (blocks 0-1)
- FMC_PFB23CR controls the upper code addresses (blocks 2-3)

If swap is used, the opposite is true:

- FMC_PFB01CR controls the upper code addresses (now in blocks 0-1)
- FMC_PFB23CR controls the lower code addresses (now in blocks 2-3)

3.5.3 SRAM Configuration

This section summarizes how the module has been configured in the chip.

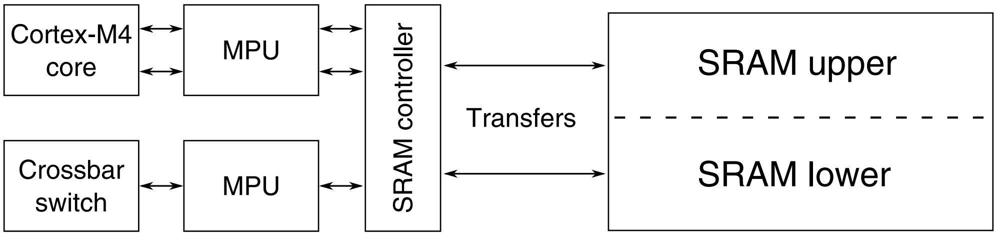


Figure 3-25. SRAM configuration

Table 3-39. Reference links to related information

Topic	Related module	Reference
Full description	SRAM	SRAM
System memory map		System memory map
Clocking		Clock Distribution
Transfers	SRAM controller	SRAM controller
	ARM Cortex-M4 core	ARM Cortex-M4 core
	Memory protection unit	Memory protection unit

3.5.3.1 SRAM sizes

This device contains SRAM tightly coupled to the ARM Cortex-M4 core. The amount of SRAM for the devices covered in this document is shown in the following table.

Device	SRAM (KB)
MK20FX512VLQ12	128
MK20FN1M0VLQ12	128
MK20FX512VMD12	128
MK20FN1M0VMD12	128

3.5.3.2 SRAM Arrays

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

The on-chip RAM is implemented such that the SRAM_L and SRAM_U ranges form a contiguous block in the memory map. As such:

- SRAM_L is anchored to 0x1FFF_FFFF and occupies the space before this ending address.
- SRAM_U is anchored to 0x2000_0000 and occupies the space after this beginning address.

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

- SRAM_L = [0x2000_0000–(SRAM_size/2)] to 0x1FFF_FFFF
- SRAM_U = 0x2000_0000 to [0x2000_0000+(SRAM_size/2)-1]

This is illustrated in the following figure.

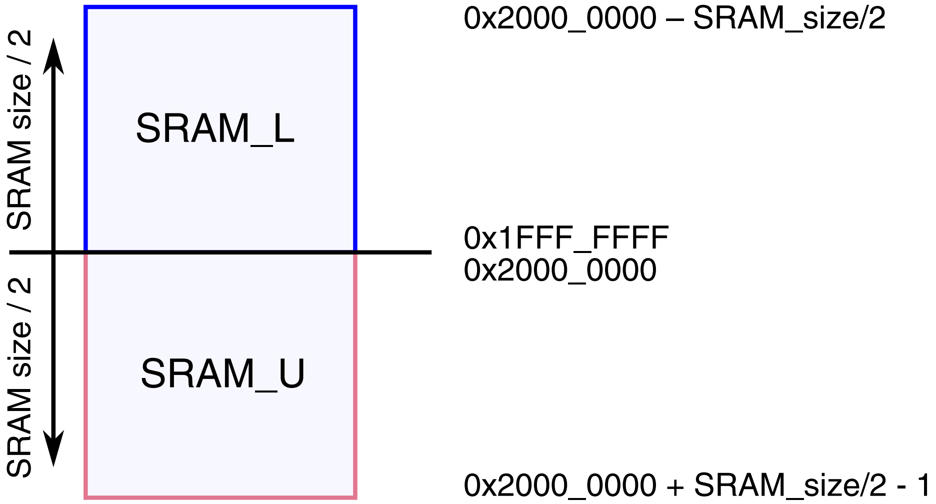


Figure 3-26. SRAM blocks memory map

For example, for a device containing 64 KB of SRAM the ranges are:

- SRAM_L: 0x1FFF_8000 – 0x1FFF_FFFF
- SRAM_U: 0x2000_0000 – 0x2000_7FFF

3.5.3.3 SRAM retention in low power modes

The SRAM is retained down to VLLS3 mode.

In VLLS2 the 16 KB region of SRAM_U from 0x2000_0000 is powered.

In VLLS1 no SRAM is retained. However, the 32-byte register file is available in VLLS1.

3.5.3.4 SRAM accesses

The SRAM is split into two logical arrays that are 32-bits wide.

- SRAM_L — Accessible by the code bus of the Cortex-M4 core and by the backdoor port.
- SRAM_U — Accessible by the system bus of the Cortex-M4 core and by the backdoor port.

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

The following figure illustrates the SRAM accesses within the device.

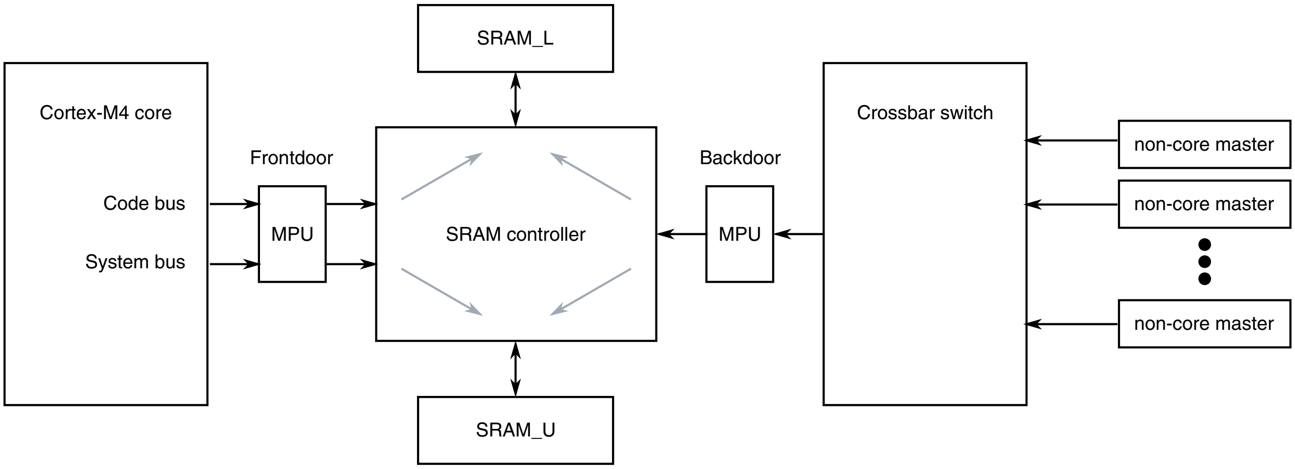


Figure 3-27. SRAM access diagram

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,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.

3.5.3.5 SRAM arbitration and priority control

The MCM's SRAMAP register controls the arbitration and priority schemes for the two SRAM arrays.

3.5.4 Local Memory Controller Configuration

This section summarizes how the module has been configured in the chip.

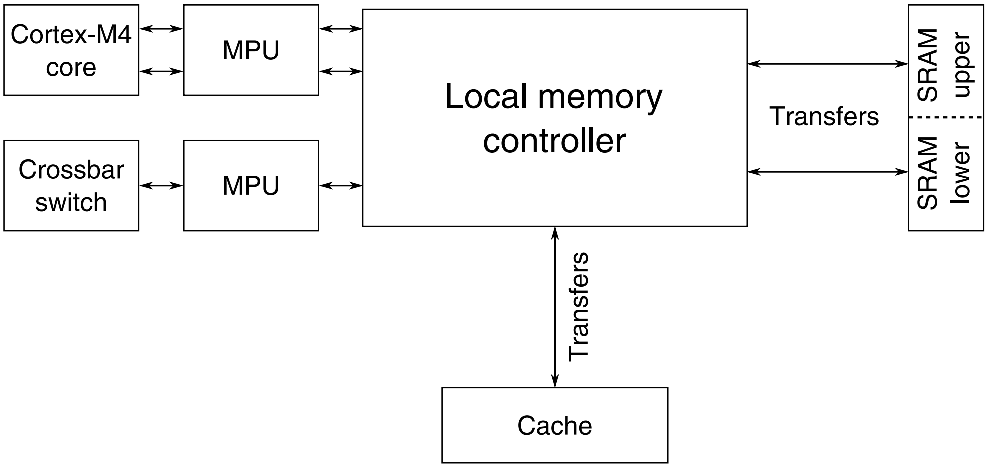


Figure 3-28. Local memory controller configuration

Table 3-40. Reference links to related information

Topic	Related module	Reference
Full description	Local memory controller	Local memory controller
System memory map		System memory map
Power management		Power management
Transfers	SRAM	SRAM
	ARM Cortex-M4 core	ARM Cortex-M4 core
	Memory protection unit	Memory protection unit

3.5.4.1 Local memory controller region assignment

The following table shows the LMEM's region mode register assignment for each region field. It also shows the available cache modes for each region.

Table 3-41. Cache regions

Address range	Destination slave	Region number	Available cache modes
0x0000_0000–0x07FF_FFFF	Program flash and read-only data	R0	Write-through and non-cacheable [1]
0x0800_0000–0x0FFF_FFFF	DRAM Controller (Aliased Area)	R1	Write-through and non-cacheable

Table continues on the next page...

Table 3-41. Cache regions (continued)

Address range	Destination slave	Region number	Available cache modes
0x1000_0000–0x17FF_FFFF	FlexNVM	R2	Write-through and non-cacheable [1]
0x1800_0000–0x1BFF_FFFF	FlexBus (Aliased Area)	R3	Write-through and non-cacheable
0x1C00_0000–0x1FFF_FFFF	SRAM_L: Lower SRAM (ICODE/ DCODE)	R4	Non-cacheable
0x2000_0000–0x200F_FFFF	SRAM_U: Upper SRAM	R5	Non-cacheable
0x6000_0000–0x6FFF_FFFF	Flexbus (External memory - Write-back)	R6	Write-back, write-through, and non-cacheable
0x9000_0000–0x9FFF_FFFF	FlexBus (External memory - Write-through)	R9	Write-through and non-cacheable

[1] Cache write hits do not write-through to program flash or FlexNVM regions because flash writes require flash programming.

3.5.5 System Register File Configuration

This section summarizes how the module has been configured in the chip.

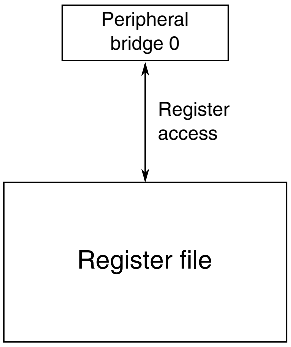


Figure 3-29. System Register file configuration

Table 3-42. Reference links to related information

Topic	Related module	Reference
Full description	Register file	Register file
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management

3.5.5.1 System Register file

This device includes a 32-byte register file that is powered in all power modes.

Also, it retains contents during low-voltage detect (LVD) events and is only reset during a power-on reset.

3.5.6 VBAT Register File Configuration

This section summarizes how the module has been configured in the chip.

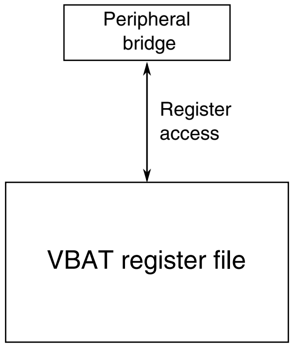


Figure 3-30. VBAT Register file configuration

Table 3-43. Reference links to related information

Topic	Related module	Reference
Full description	VBAT register file	VBAT register file
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management

3.5.6.1 VBAT register file

This device includes a 32-byte register file that is powered in all power modes and is powered by VBAT.

It is only reset during VBAT power-on reset.

3.5.7 EzPort Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

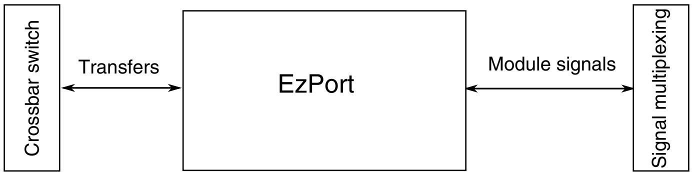


Figure 3-31. EzPort configuration

Table 3-44. Reference links to related information

Topic	Related module	Reference
Full description	EzPort	EzPort
System memory map		System memory map
Clocking		Clock Distribution
Transfers	Crossbar switch	Crossbar switch
Signal Multiplexing	Port control	Signal Multiplexing

3.5.7.1 JTAG instruction

The system JTAG controller implements an EZPORT instruction. When executing this instruction, the JTAG controller resets the core logic and asserts the EzPort chip select signal to force the processor into EzPort mode.

3.5.7.2 Flash Option Register (FOPT)

The FOPT[EZPORT_DIS] bit can be used to prevent entry into EzPort mode during reset. If the FOPT[EZPORT_DIS] bit is cleared, then the state of the chip select signal ($\overline{\text{EZP\_CS}}$) is ignored and the MCU always boots in normal mode.

This option is useful for systems that use the $\overline{\text{EZP\_CS}}$/NMI signal configured for its NMI function. Disabling EzPort mode prevents possible unwanted entry into EzPort mode if the external circuit that drives the NMI signal asserts it during reset.

The FOPT register is loaded from the flash option byte. If the flash option byte is modified the new value takes effect for any subsequent resets, until the value is changed again.

3.5.8 FlexBus Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

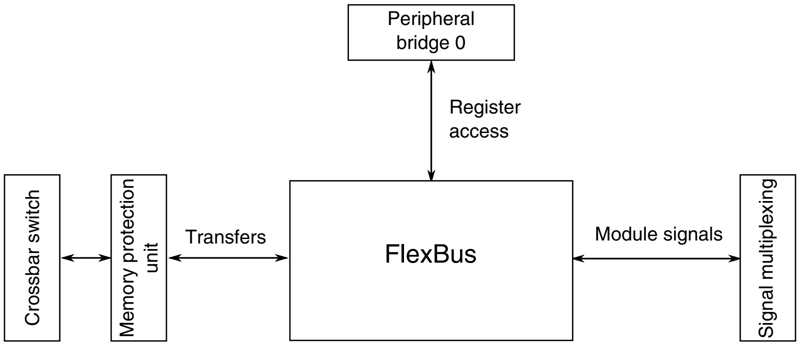


Figure 3-32. FlexBus configuration

Table 3-45. 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
Transfers	Memory protection unit (MPU)	Memory protection unit (MPU)
Signal multiplexing	Port control	Signal multiplexing

3.5.8.1 FlexBus clocking

The system provides a dedicated clock source to the FlexBus module's external FB_CLKOUT. Its clock frequency is derived from a divider of the MCGOUTCLK. See Clock Distribution for more details.

3.5.8.2 FlexBus signal multiplexing

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 the port control and FlexBus modules. The port control module registers control whether the FlexBus or another module signals are available on the external pin, while the FlexBus's CSPMCR register configures which FlexBus signals are available from the module. The control signals are grouped as illustrated:

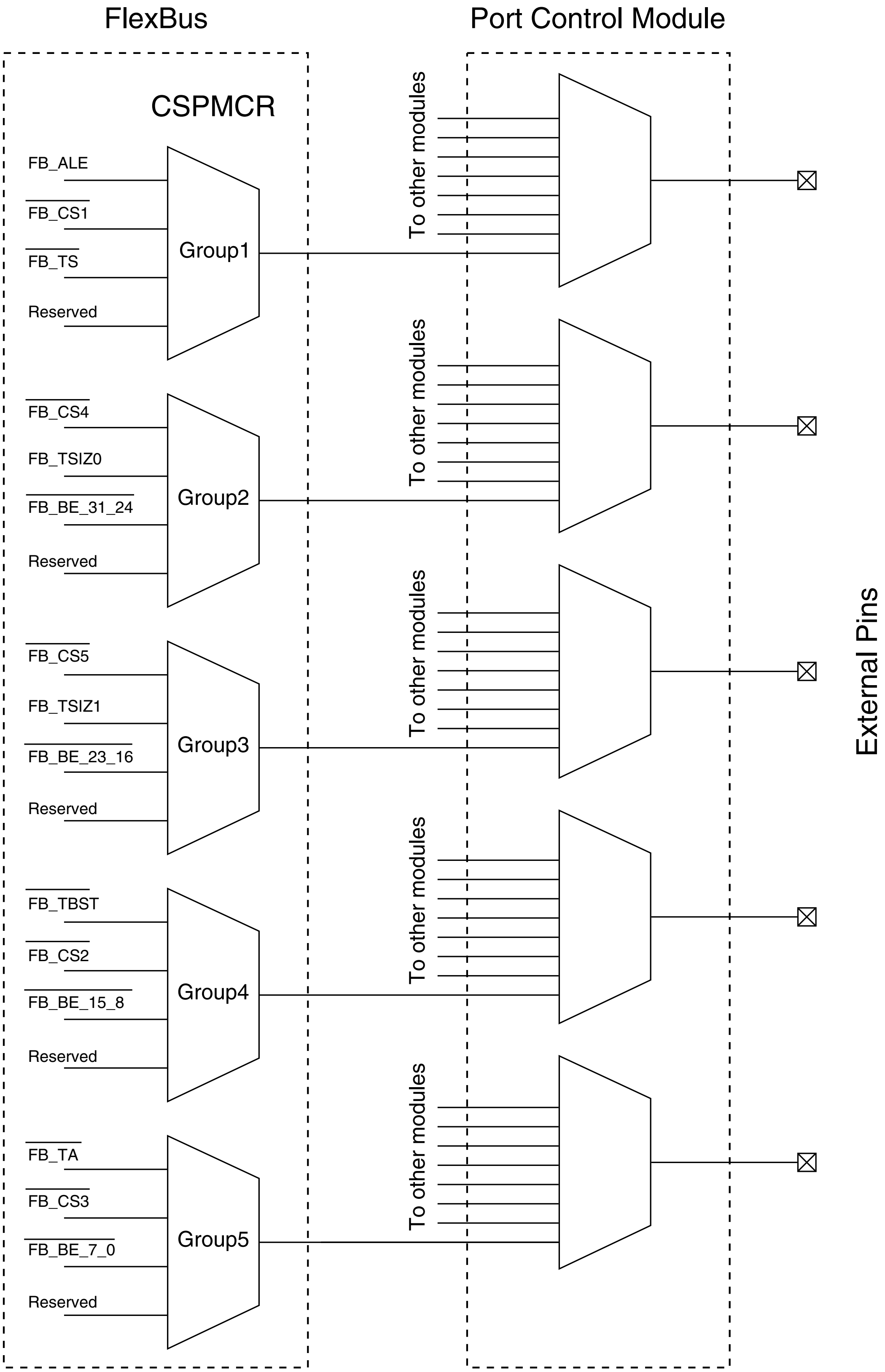


Figure 3-33. FlexBus control signal multiplexing

Therefore, use the CSPMCR and port control registers to configure which control signal is available on the external pin. All control signals, except for $\overline{\text{FB\_TA}}$, are assigned to the ALT5 function in the port control module. Since, unlike the other control signals, $\overline{\text{FB\_TA}}$ is an input signal, it is assigned to the ALT6 function.

3.5.8.3 FlexBus CSCR0 reset value

On this device the CSCR0 resets to 0x003F_FC00. Configure this register as needed before performing any FlexBus access.

3.5.8.4 FlexBus Security

When security is enabled on the device, FlexBus accesses may be restricted by configuring the FBSL field in the SIM's SOPT2 register. See System Integration Module (SIM) for details.

3.5.8.5 FlexBus line transfers

Line transfers are not possible from the ARM Cortex-M4 core. Ignore any references to line transfers in the FlexBus chapter.

3.5.9 NAND Flash Controller (NFC) Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

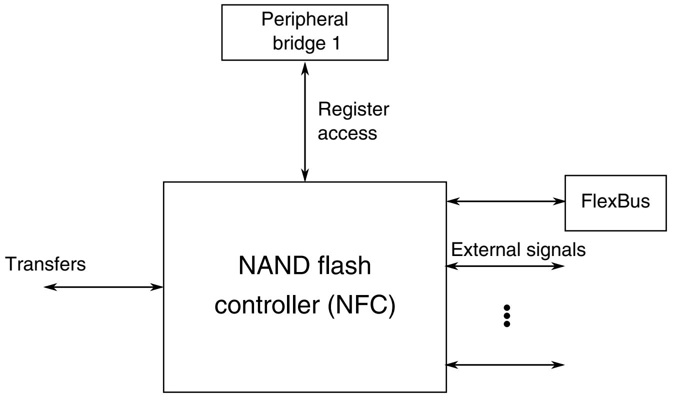


Figure 3-34. NAND Flash Controller configuration

Table 3-46. Reference links to related information

Topic	Related module	Reference
Full description	NAND Flash Controller (NFC)	NFC
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.5.9.1 NFC clock

The NFC clock frequency is determined by the selected NFC clock source and the SIM_CLKDIV4[NFCDIV, NFCFRAC] bitfields. See NAND Flash Controller (NFC) clocking for more details.

3.6 Security

3.6.1 CRC Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

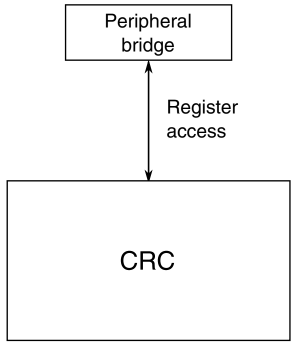


Figure 3-35. CRC configuration

Table 3-47. Reference links to related information

Topic	Related module	Reference
Full description	CRC	CRC
System memory map		System memory map
Power management		Power management

3.7 Analog

3.7.1 16-bit SAR ADC with PGA Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

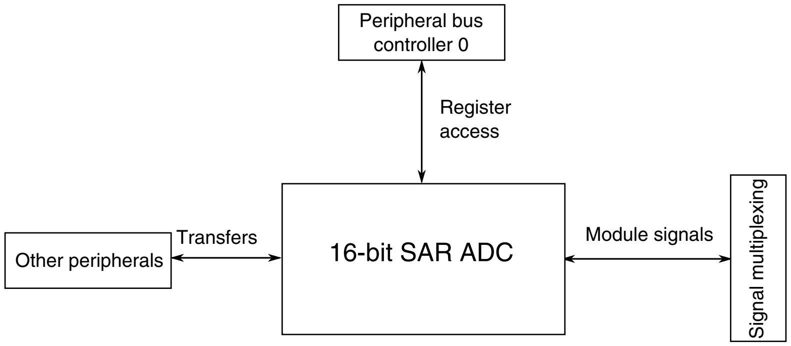


Figure 3-36. 16-bit SAR ADC with PGA configuration

Table 3-48. Reference links to related information

Topic	Related module	Reference
Full description	16-bit SAR ADC with PGA	16-bit SAR ADC with PGA

Table continues on the next page...

Table 3-48. Reference links to related information (continued)

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

3.7.1.1 ADC instantiation information

This device contains four ADCs. Each ADC contains a PGA channel for a total of four separate PGAs.

3.7.1.1.1 Number of ADC channels

The number of ADC channels present on the device is determined by the pinout of the specific device package. For details regarding the number of ADC channel available on a particular package, refer to the signal multiplexing chapter of this MCU.

3.7.1.2 DMA Support on ADC

Applications may require continuous sampling of the ADC (4K samples/sec) that may have considerable load on the CPU. Though using PDB to trigger ADC may reduce some CPU load, The ADC supports DMA request functionality for higher performance when the ADC is sampled at a very high rate or cases were PDB is bypassed. The ADC can trigger the DMA (via DMA req) on conversion completion.

3.7.1.3 ADC0 Connections/Channel Assignment

NOTE

As indicated by the following sections, each ADCx_DPx input and certain ADCx_DMx inputs may operate as single-ended ADC channels in single-ended mode.

3.7.1.3.1 ADC0 Channel Assignment for 144-Pin Package

ADC Channel (SC1n[ADCH])	Channel	Input signal (SC1n[DIFF]= 1)	Input signal (SC1n[DIFF]= 0)
00000	DAD0	ADC0_DP0 and ADC0_DM0[1]	ADC0_DP0[2]
00001	DAD1	ADC0_DP1 and ADC0_DM1	ADC0_DP1
00010	DAD2	PGA0_DP and PGA0_DM	PGA0_DP
00011	DAD3	ADC0_DP3 and ADC0_DM3[3]	ADC0_DP3[4]
00100[5]	AD4a	Reserved	Reserved
00101[5]	AD5a	Reserved	Reserved
00110[5]	AD6a	Reserved	Reserved
00111[5]	AD7a	Reserved	Reserved
00100[5]	AD4b	Reserved	ADC0_SE4b
00101[5]	AD5b	Reserved	ADC0_SE5b
00110[5]	AD6b	Reserved	ADC0_SE6b
00111[5]	AD7b	Reserved	ADC0_SE7b
01000	AD8	Reserved	ADC0_SE8[6]
01001	AD9	Reserved	ADC0_SE9[7]
01010	AD10	Reserved	ADC0_SE10
01011	AD11	Reserved	ADC0_SE11
01100	AD12	Reserved	ADC0_SE12
01101	AD13	Reserved	ADC0_SE13
01110	AD14	Reserved	ADC0_SE14
01111	AD15	Reserved	ADC0_SE15
10000	AD16	Reserved	ADC0_SE16
10001	AD17	Reserved	ADC0_SE17
10010	AD18	Reserved	ADC0_SE18
10011	AD19	Reserved	ADC0_DM0[8]
10100	AD20	Reserved	ADC0_DM1
10101	AD21	Reserved	
10110	AD22	Reserved	
10111	AD23	Reserved	12-bit DAC0 Output
11000	AD24	Reserved	Reserved
11001	AD25	Reserved	Reserved
11010	AD26	Temperature Sensor (Diff)	Temperature Sensor (S.E)
11011	AD27	Bandgap (Diff)[9]	Bandgap (S.E)[9]
11100	AD28	Reserved	Reserved
11101	AD29	-VREFH (Diff)	VREFH (S.E)
11110	AD30	Reserved	VREFL
11111	AD31	Module Disabled	Module Disabled

[1] Interleaved with ADC1_DP3 and ADC1_DM3
[2] Interleaved with ADC1_DP3
[3] Interleaved with ADC1_DP0 and ADC1_DM0
[4] Interleaved with ADC1_DP0

5. ADCx_CFG2[MUXSEL] bit selects between ADCx_SEn channels a and b. Refer to MUXSEL description in ADC chapter for details.
6. Interleaved with ADC1_SE8
7. Interleaved with ADC1_SE9
8. Interleaved with ADC1_DM3
9. This is the PMC bandgap 1V reference voltage not the VREF module 1.2 V reference voltage. Prior to reading from this ADC channel, ensure that you enable the bandgap buffer by setting the PMC_REGSC[BGBE] bit. Refer to the device data sheet for the bandgap voltage (V_{BG}) specification.

3.7.1.4 ADC1 Connections/Channel Assignment

NOTE

As indicated in the following tables, each ADCx_DPx input and certain ADCx_DMx inputs may operate as single-ended ADC channels in single-ended mode.

3.7.1.4.1 ADC1 Channel Assignment for 144-Pin Package

ADC Channel (SC1n[ADCH])	Channel	Input signal (SC1n[DIFF]= 1)	Input signal (SC1n[DIFF]= 0)
00000	DAD0	ADC1_DP0 and ADC1_DM0[1]	ADC1_DP0[2]
00001	DAD1	ADC1_DP1 and ADC1_DM1	ADC1_DP1
00010	DAD2	PGA1_DP and PGA1_DM	PGA1_DP
00011	DAD3	ADC1_DP3 and ADC1_DM3[3]	ADC1_DP3[4]
00100[5]	AD4a	Reserved	ADC1_SE4a
00101[5]	AD5a	Reserved	ADC1_SE5a
00110[5]	AD6a	Reserved	ADC1_SE6a
00111[5]	AD7a	Reserved	ADC1_SE7a
00100[5]	AD4b	Reserved	ADC1_SE4b
00101[5]	AD5b	Reserved	ADC1_SE5b
00110[5]	AD6b	Reserved	ADC1_SE6b
00111[5]	AD7b	Reserved	ADC1_SE7b
01000	AD8	Reserved	ADC1_SE8[6]
01001	AD9	Reserved	ADC1_SE9[7]
01010	AD10	Reserved	ADC1_SE10
01011	AD11	Reserved	ADC1_SE11
01100	AD12	Reserved	ADC1_SE12
01101	AD13	Reserved	ADC1_SE13
01110	AD14	Reserved	ADC1_SE14
01111	AD15	Reserved	ADC1_SE15
10000	AD16	Reserved	ADC1_SE16
10001	AD17	Reserved	ADC1_SE17
10010	AD18	Reserved	VREF Output

Table continues on the next page...

ADC Channel (SC1n[ADCH])	Channel	Input signal (SC1n[DIFF]= 1)	Input signal (SC1n[DIFF]= 0)
10011	AD19	Reserved	ADC1_DM0[8]
10100	AD20	Reserved	ADC1_DM1
10101	AD21	Reserved	Reserved
10110	AD22	Reserved	
10111	AD23	Reserved	12-bit DAC1 Output
11000	AD24	Reserved	Reserved
11001	AD25	Reserved	Reserved
11010	AD26	Temperature Sensor (Diff)	Temperature Sensor (S.E)
11011	AD27	Bandgap (Diff)[9]	Bandgap (S.E)[9]
11100	AD28	Reserved	Reserved
11101	AD29	-VREFH (Diff)	VREFH (S.E)
11110	AD30	Reserved	VREFL
11111	AD31	Module Disabled	Module Disabled

1. Interleaved with ADC0_DP3 and ADC0_DM3
2. Interleaved with ADC0_DP3
3. Interleaved with ADC0_DP0 and ADC0_DM0
4. Interleaved with ADC0_DP0
5. ADCx_CFG2[MUXSEL] bit selects between ADCx_SEn channels a and b. Refer to MUXSEL description in ADC chapter for details.
6. Interleaved with ADC0_SE8
7. Interleaved with ADC0_SE9
8. Interleaved with ADC0_DM3
9. This is the PMC bandgap 1V reference voltage not the VREF module 1.2 V reference voltage. Prior to reading from this ADC channel, ensure that you enable the bandgap buffer by setting the PMC_REGSC[BGBE] bit. Refer to the device data sheet for the bandgap voltage (V_{BG}) specification.

3.7.1.5 ADC2 Connections/Channel Assignment

NOTE

As indicated by the above table, each ADCx_DPx input and certain ADCx_DMx inputs may operate as single-ended ADC channels in single-ended mode.

3.7.1.6 ADC3 Connections/Channel Assignment

NOTE

As indicated by the above table, each ADCx_DPx input and certain ADCx_DMx inputs may operate as single-ended ADC channels in single-ended mode.

3.7.1.7 ADC Channels MUX Selection

The following figure shows the assignment of ADCx_SEn channels a and b through a MUX selection to ADC. To select between alternate set of channels, refer to ADCx_CFG2[MUXSEL] bit settings for more details.

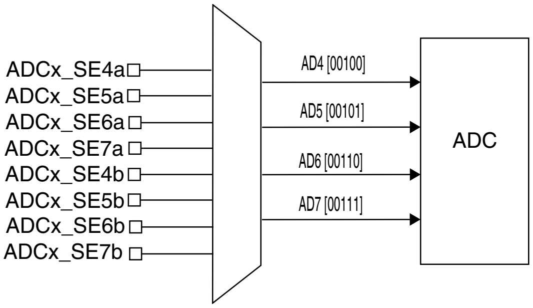


Figure 3-37. ADCx_SEn channels a and b selection

3.7.1.8 ADC Hardware Interleaved Channels

The AD8 and AD9 channels on ADCx are interleaved in hardware using the following configuration.

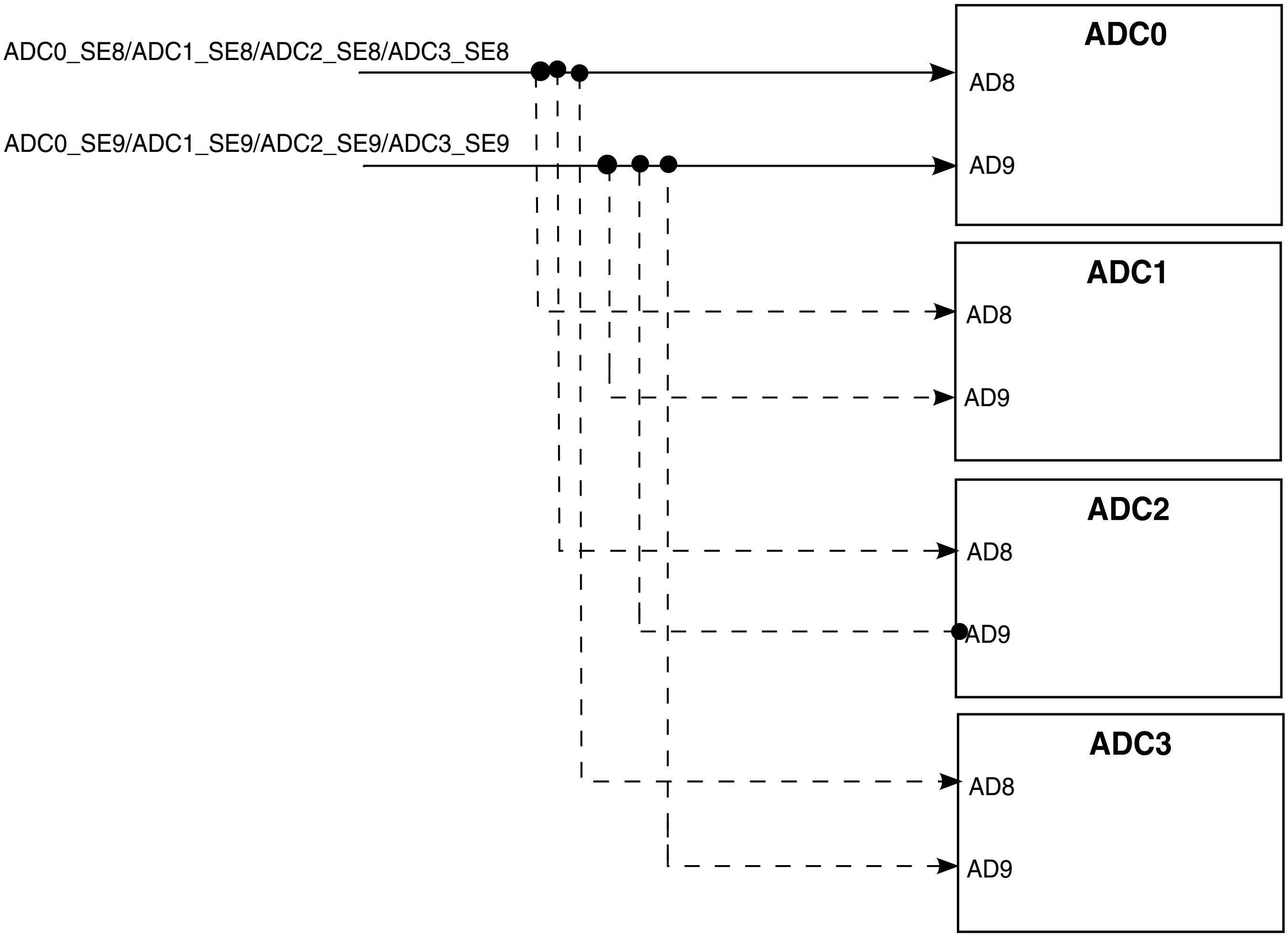

ADC0_SE8/ADC1_SE8/ADC2_SE8/ADC3_SE8
ADC0_SE9/ADC1_SE9/ADC2_SE9/ADC3_SE9
ADC0
AD8
AD9
ADC1
AD8
AD9
ADC2
AD8
AD9
ADC3
AD8
AD9

Figure 3-38. ADC hardware interleaved channels integration

3.7.1.9 ADC and PGA Reference Options

The ADC supports the following references:

- VREFH/VREFL - connected as the primary reference option
- 1.2 V VREF_OUT - connected as the V_{ALT} reference option

ADCx_SC2[REFSEL] bit selects the voltage reference sources for ADC. Refer to REFSEL description in ADC chapter for more details.

The only reference option for the PGA is the 1.2 V VREF_OUT source. The VREF_OUT signal can either be driven by an external voltage source via the VREF_OUT pin or from the output of the VREF module. Ensure that the VREF module is disabled when an external voltage source is used instead. For PGA maximum differential input signal swing range, refer to the device data sheet for 16-bit ADC with PGA characteristics.

3.7.1.10 ADC triggers

The ADC supports both software and hardware triggers. The primary hardware mechanism for triggering the ADC is the PDB. The PDB itself can be triggered by other peripherals. For example: RTC (Alarm, Seconds) signal is connected to the PDB. The PDB trigger can receive the RTC (alarm/seconds) trigger input forcing ADC conversions in run mode (where PDB is enabled). On the other hand, the ADC can conduct conversions in low power modes, not triggered by PDB. This allows the ADC to do conversions in low power mode and store the output in the result register. The ADC generates interrupt when the data is ready in the result register that wakes the system from low power mode. The PDB can also be bypassed by using the ADCxTRGSEL bits in the SOPT7 register.

For operation of triggers in different modes, refer to Power Management chapter.

3.7.1.11 Alternate clock

For this device, the alternate clock is connected to OSC0ERCLK.

NOTE

> This clock option is only usable when OSC0ERCLK is in the MHz range. A system with OSC0ERCLK in the kHz range has the optional clock source below minimum ADC clock operating frequency.

3.7.1.12 ADC low-power modes

This table shows the ADC low-power modes and the corresponding chip low-power modes.

Table 3-49. ADC low-power modes

Module mode	Chip mode
Wait	Wait, VLPW
Normal Stop	Stop, VLPS
Low Power Stop	LLS, VLLS3, VLLS2, VLLS1

3.7.1.13 PGA Integration

- No additional external pins are required for the PGA as it is part of the ADC and is selected as a separate channel
- Each PGA connects to the differential ADC channels
- The PGA outputs differential pairs that are connected to ADC differential input
- When the PGA is used, differential input from the pins is connected to differential input channel 2 on ADCx

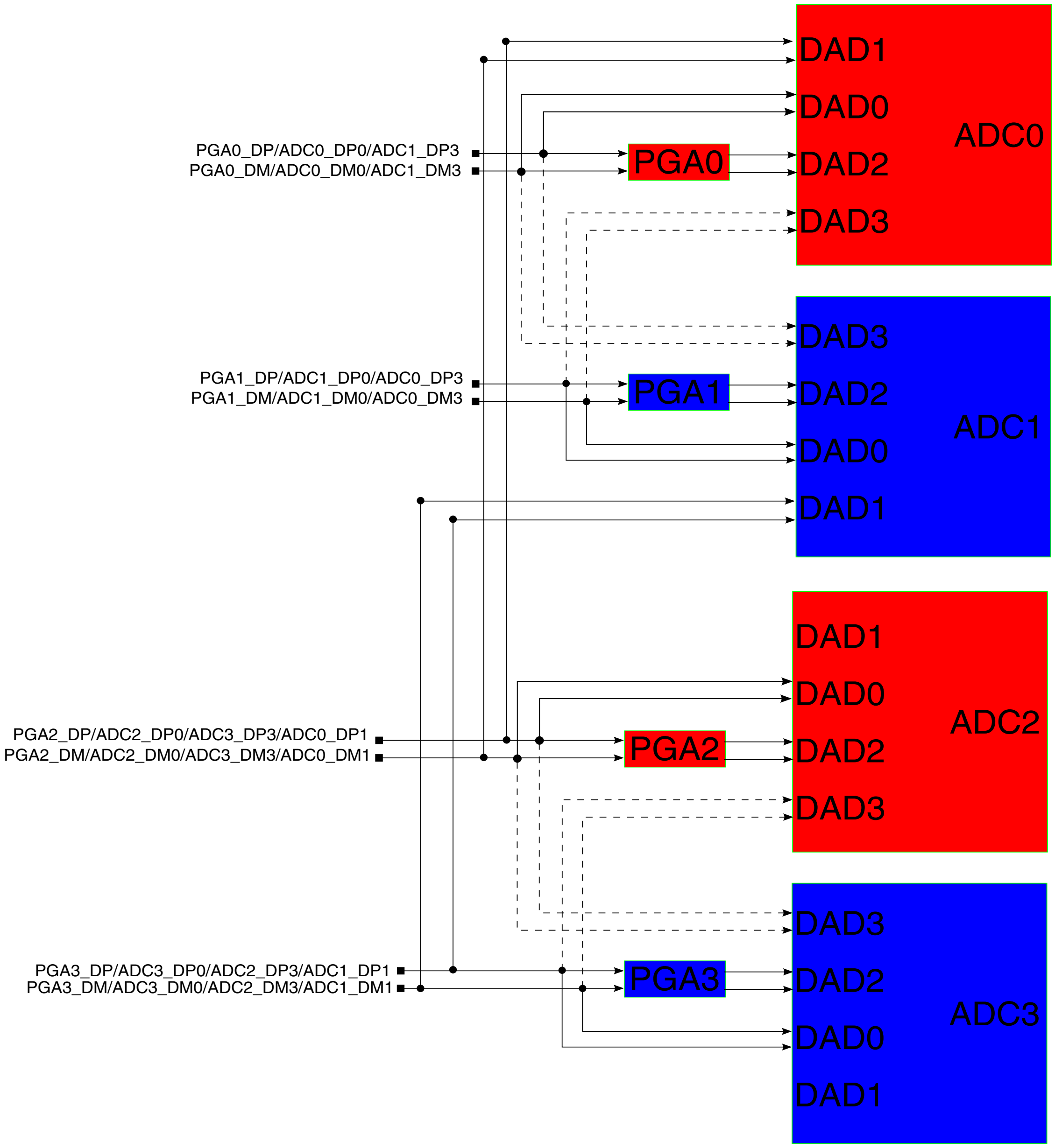

PGA0_DP/ADC0_DP0/ADC1_DP3
PGA0_DM/ADC0_DM0/ADC1_DM3
PGA1_DP/ADC1_DP0/ADC0_DP3
PGA1_DM/ADC1_DM0/ADC0_DM3
PGA2_DP/ADC2_DP0/ADC3_DP3/ADC0_DP1
PGA2_DM/ADC2_DM0/ADC3_DM3/ADC0_DM1
PGA3_DP/ADC3_DP0/ADC2_DP3/ADC1_DP1
PGA3_DM/ADC3_DM0/ADC2_DM3/ADC1_DM1
PGA0
PGA1
PGA2
PGA3
ADC0
DAD1
DAD0
DAD2
DAD3
ADC1
DAD3
DAD2
DAD0
DAD1
ADC2
DAD1
DAD0
DAD2
DAD3
ADC3
DAD3
DAD2
DAD0
DAD1

Figure 3-39. PGA Integration

3.7.2 CMP Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

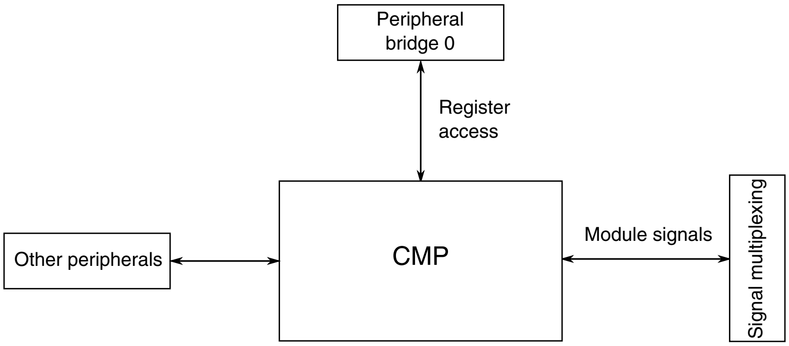


Figure 3-40. CMP configuration

Table 3-50. Reference links to related information

Topic	Related module	Reference
Full description	Comparator (CMP)	Comparator
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.7.2.1 CMP input connections

The following table shows the fixed internal connections to the CMP.

Table 3-51. CMP input connections

CMP Inputs	CMP0	CMP1	CMP2	CMP3
IN0	CMP0_IN0	CMP1_IN0	CMP2_IN0	CMP3_IN0
IN1	CMP0_IN1	CMP1_IN1	CMP2_IN1	CMP3_IN1
IN2	CMP0_IN2	ADC0SE16/CMP1_IN2	ADC1SE16/CMP2_IN2	CMP3_IN2
IN3	CMP0_IN3	12b DAC0 Reference/ CMP1_IN3	12b DAC1 Reference/ CMP2_IN3	12b DAC0Reference/ CMP2_IN3
IN4	12b DAC1 Reference	—	CMP2_IN4	CMP3_IN4
IN5	VREF Output/ CMP0_IN5	VREF Output/ CMP1_IN5	CMP2_IN5	CMP3_IN5

Table continues on the next page...

Table 3-51. CMP input connections (continued)

CMP Inputs	CMP0	CMP1	CMP2	CMP3
IN6	Bandgap	Bandgap	Bandgap	Bandgap
IN7	6b DAC0 Reference	6b DAC1 Reference	6b DAC2 Reference	6b DAC3 Reference

3.7.2.2 CMP external references

The 6-bit DAC sub-block supports selection of two references. For this device, the references are connected as follows:

- VREF_OUT - V_{in1} input
- VDD - V_{in2} input

3.7.2.3 External window/sample input

Individual PDB pulse-out signals control each CMP Sample/Window timing.

3.7.3 12-bit DAC Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

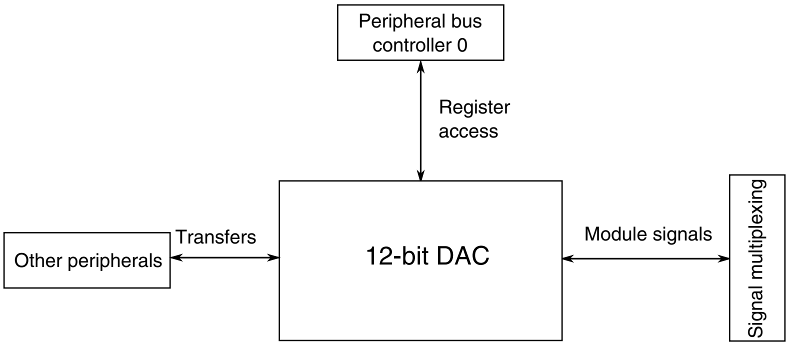


Figure 3-41. 12-bit DAC configuration

Table 3-52. Reference links to related information

Topic	Related module	Reference
Full description	12-bit DAC	12-bit DAC
System memory map		System memory map
Clocking		Clock distribution

Table continues on the next page...

Table 3-52. Reference links to related information (continued)

Topic	Related module	Reference
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.7.3.1 12-bit DAC Overview

This device contains two 12-bit digital-to-analog converters (DAC) with programmable reference generator output. The DAC includes a FIFO for DMA support.

3.7.3.2 12-bit DAC 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.

3.7.3.3 12-bit DAC Reference

For this device VREF_OUT and VDDA are selectable as the DAC reference. VREF_OUT is connected to the DACREF_1 input and VDDA is connected to the DACREF_2 input. Use DACx_C0[DACRFS] control bit to select between these two options.

Be aware that if the DAC and ADC use the VREF_OUT reference simultaneously, some degradation of ADC accuracy is to be expected due to DAC switching.

3.7.4 VREF Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

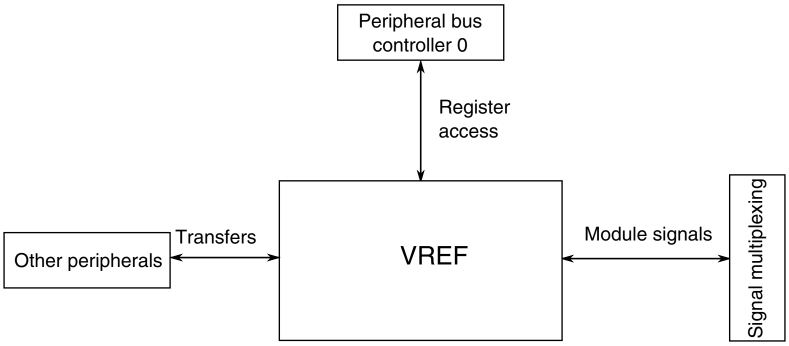


Figure 3-42. VREF configuration

Table 3-53. Reference links to related information

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

3.7.4.1 VREF Overview

This device includes a voltage reference (VREF) to supply an accurate 1.2 V voltage output.

The voltage reference can provide a reference voltage to external peripherals or a reference to analog peripherals, such as the ADC, DAC, or CMP.

NOTE

PMC_REGSC[BGEN] bit must be set for VREF operation in VLPx modes.

NOTE

For either an internal or external reference if the VREF_OUT functionality is being used, VREF_OUT signal must be connected to an output load capacitor. Refer the device data sheet for more details.

3.8 Timers

3.8.1 PDB Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

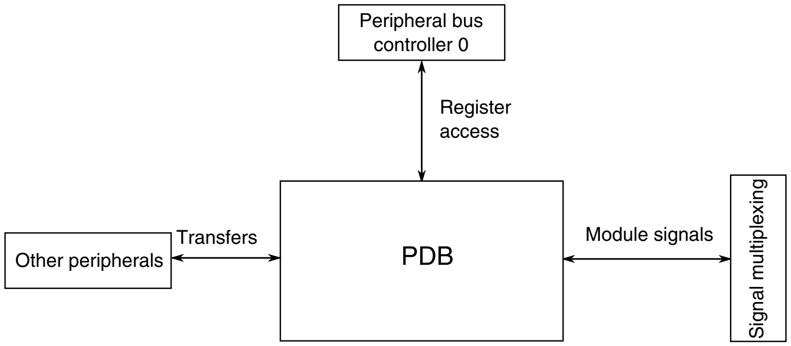


Figure 3-43. PDB configuration

Table 3-54. Reference links to related information

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

3.8.1.1 PDB Instantiation

3.8.1.1.1 PDB Output Triggers

Table 3-55. PDB output triggers

Number of PDB channels for ADC trigger	4
Number of pre-triggers per PDB channel	2
Number of DAC triggers	2
Number of PulseOut	4

3.8.1.1.2 PDB Input Trigger Connections

Table 3-56. PDB Input Trigger Options

PDB Trigger	PDB Input
0000	External Trigger
0001	CMP 0
0010	CMP 1
0011	CMP 2
0100	PIT Ch 0 Output
0101	PIT Ch 1 Output
0110	PIT Ch 2 Output
0111	PIT Ch 3 Output
1000	FTM0 Init and Ext Trigger Outputs
1001	FTM1 Init and Ext Trigger Outputs
1010	FTM2 Init and Ext Trigger Outputs
1011	FTM3 Init and Ext Trigger Outputs
1100	RTC Alarm
1101	RTC Seconds
1110	LPTMR Output
1111	Software Trigger

3.8.1.2 PDB Module Interconnections

PDB trigger outputs	Connection
Channel 0 triggers	ADC0 trigger
Channel 1 triggers	ADC1 trigger and synchronous input 1 of FTM0
Channel 2 triggers	ADC2 trigger
Channel 3 triggers	ADC3 trigger and synchronous input 1 of FTM3
DAC triggers	DAC0 and DAC1 trigger
Pulse-out	Pulse-out connected to each CMP module's sample/window input to control sample operation

3.8.1.3 Back-to-back acknowledgement connections

In this MCU, the following PDB back-to-back operation acknowledgment connections are implemented based on SIM_MCR[PDBLOOP] bit setting.

When SIM_MCR[PDBLOOP]=0:

- PDB channel 0 pre-trigger 0 acknowledgement input: ADC1SC1B_COCO

- PDB channel 0 pre-trigger 1 acknowledgement input: ADC0SC1A_COCO
- PDB channel 1 pre-trigger 0 acknowledgement input: ADC0SC1B_COCO
- PDB channel 1 pre-trigger 1 acknowledgement input: ADC1SC1A_COCO

- PDB channel 2 pre-trigger 0 acknowledgement input: ADC3SC1B_COCO
- PDB channel 2 pre-trigger 1 acknowledgement input: ADC2SC1A_COCO
- PDB channel 3 pre-trigger 0 acknowledgement input: ADC2SC1B_COCO
- PDB channel 3 pre-trigger 1 acknowledgement input: ADC3SC1A_COCO

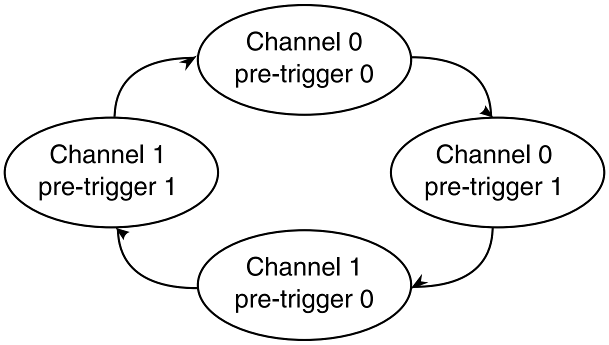


Figure 3-44. PDB back-to-back chain 1

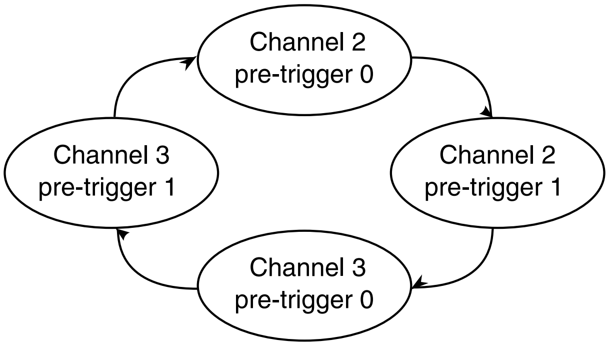


Figure 3-45. PDB back-to-back chain 2

When SIM_MCR[PDBLOOP]=1:

- PDB channel 0 pre-trigger 0 acknowledgement input: ADC3SC1B_COCO
- PDB channel 0 pre-trigger 1 acknowledgement input: ADC0SC1A_COCO
- PDB channel 1 pre-trigger 0 acknowledgement input: ADC0SC1B_COCO
- PDB channel 1 pre-trigger 1 acknowledgement input: ADC1SC1A_COCO
- PDB channel 2 pre-trigger 0 acknowledgement input: ADC1SC1B_COCO
- PDB channel 2 pre-trigger 1 acknowledgement input: ADC2SC1A_COCO
- PDB channel 3 pre-trigger 0 acknowledgement input: ADC2SC1B_COCO
- PDB channel 3 pre-trigger 1 acknowledgement input: ADC3SC1A_COCO

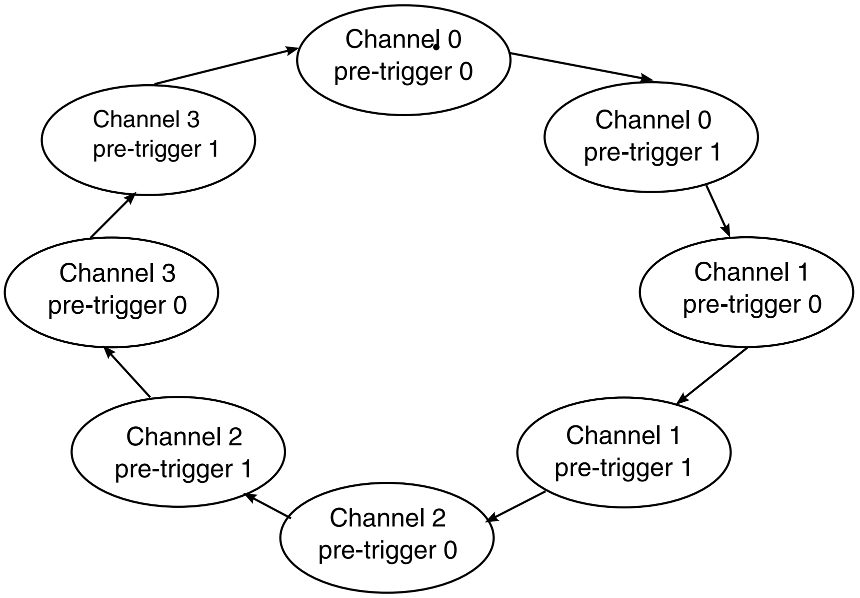


Figure 3-46. PDB back-to-back chain

The application code can set the PDB*x*_CH*n*C1[BB] bits to configure the PDB pre-triggers as a single chain or several chains.

3.8.1.4 PDB Interval Trigger Connections to DAC

In this MCU, PDB interval trigger connections to DAC are implemented as follows.

- PDB interval trigger 0 connects to DAC0 hardware trigger input.
- PDB interval trigger 1 connects to DAC1 hardware trigger input.

3.8.1.5 DAC External Trigger Input Connections

In this MCU, two DAC external trigger inputs are implemented.

- DAC external trigger input 0: ADC0SC1A_COCO
- DAC external trigger input 1: ADC1SC1A_COCO

NOTE

Application code can set the PDBx_DACINTCn[EXT] bit to allow DAC external trigger input when the corresponding ADC Conversion complete flag, ADCx_SC1n[COCO], is set.

3.8.1.6 Pulse-Out Connection

Individual PDB Pulse-Out signals are connected to each CMP block and used for sample window.

3.8.1.7 Pulse-Out Enable Register Implementation

The following table shows the comparison of pulse-out enable register at the module and chip level.

Table 3-57. PDB pulse-out enable register

Register	Module implementation	Chip implementation
POnEN	7:0 - POEN 31:8 - Reserved	0 - POEN[0] for CMP0 1 - POEN[1] for CMP1 2 - POEN[2] for CMP2 3 - POEN[3] for CMP3 31:4 - Reserved

3.8.2 FlexTimer Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

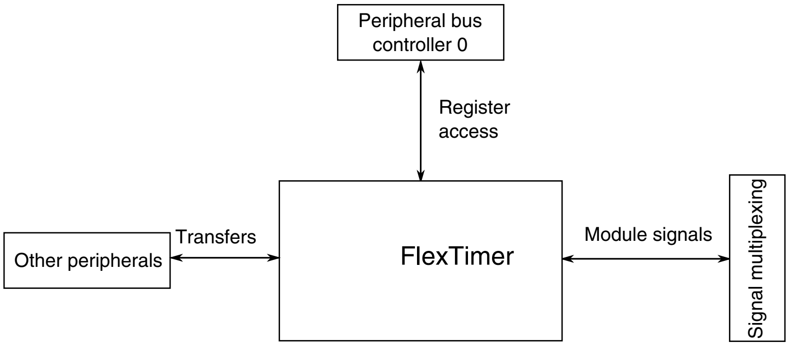


Figure 3-47. FlexTimer configuration

Table 3-58. Reference links to related information

Topic	Related module	Reference
Full description	FlexTimer	FlexTimer
System memory map		System memory map
Clocking		Clock distribution

Table continues on the next page...

Table 3-58. Reference links to related information (continued)

Topic	Related module	Reference
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.8.2.1 Instantiation Information

This device contains four FlexTimer modules.

The following table shows how these modules are configured.

Table 3-59. FTM Instantiations

FTM instance	Number of channels	Features/usage
FTM0	8	3-phase motor + 2 general purpose or stepper motor
FTM1	2	Quadrature decoder or general purpose
FTM2	2	Quadrature decoder or general purpose
FTM3	8	3-phase motor + 2 general purpose or stepper motor

Compared with the FTM0 and FTM3 configuration, the FTM1 and FTM2 configuration adds the Quadrature decoder feature and reduces the number of channels.

3.8.2.2 External Clock Options

By default each FTM is clocked by the internal bus clock (the FTM refers to it as system clock). Each module contains a register setting that allows the module to be clocked from an external clock instead. There are two external FTM_CLKINx pins that can be selected by any FTM module via the SOPT4 register in the SIM module.

3.8.2.3 Fixed frequency clock

The fixed frequency clock for each FTM is MCGFFCLK.

3.8.2.4 FTM Interrupts

The FlexTimer has multiple sources of interrupt. However, these sources are OR'd together to generate a single interrupt request per FTM module to the interrupt controller. When an FTM interrupt occurs, read the FTM status registers (FMS, SC, and STATUS) to determine the exact interrupt source.

3.8.2.5 FTM Fault Detection Inputs

The following fault detection input options for the FTM modules are selected via the SOPT4 register in the SIM module. The external pin option is selected by default.

- FTM0 FAULT0 = FTM0_FLT0 pin or CMP0 output
- FTM0 FAULT1 = FTM0_FLT1 pin or CMP1 output
- FTM0 FAULT2 = FTM0_FLT2 pin or CMP2 output
- FTM0 FAULT3 = FTM0_FLT3 pin

- FTM1 FAULT0 = FTM1_FLT0 pin or CMP0 output
- FTM1 FAULT1 = CMP1 output
- FTM1 FAULT2 = CMP2 output

- FTM2 FAULT0 = FTM2_FLT0 pin or CMP0 output
- FTM2 FAULT1 = CMP1 output
- FTM2 FAULT2 = CMP2 output

- FTM3 FAULT0 = FTM3_FLT0 pin or CMP0 output
- FTM3 FAULT1 = CMP2 output
- FTM3 FAULT2 = CMP3 output

3.8.2.6 FTM Hardware Triggers

The FTM synchronization hardware triggers are connected in the chip as follows:

- FTM0 hardware trigger 0 = CMP0 Output or FTM1 Match
- FTM0 hardware trigger 1 = PDB channel 1 Trigger Output or FTM2 Match
- FTM0 hardware trigger 2 = FTM0_FLT0 pin

- FTM1 hardware trigger 0 = CMP0 Output
- FTM1 hardware trigger 1 = CMP1 Output
- FTM1 hardware trigger 2 = FTM1_FLT0 pin

- FTM2 hardware trigger 0 = CMP0 Output

- FTM2 hardware trigger 1 = CMP2 Output
- FTM2 hardware trigger 2 = FTM2_FLT0 pin

- FTM3 hardware trigger 0 = CMP3 Output or FTM1 Match
- FTM3 hardware trigger 1 = PDB channel 3 Trigger Output or FTM2 Match
- FTM3 hardware trigger 2 = FTM3_FLT0 pin

For the triggers with more than one option, the SOPT4 register in the SIM module controls the selection.

3.8.2.7 Input capture options for FTM module instances

The following channel 0 input capture source options are selected via the SOPT4 register in the SIM module. The external pin option is selected by default.

- FTM1 channel 0 input capture = FTM1_CH0 pin or CMP0 output or CMP1 output or USB start of frame pulse
- FTM2 channel 0 input capture = FTM2_CH0 pin or CMP0 output or CMP1 output

3.8.2.8 FTM output triggers for other modules

FTM output triggers can be selected as input triggers for the PDB and ADC modules. See PDB Instantiation and ADC triggers.

3.8.2.9 FTM Global Time Base

This chip provides the optional FTM global time base feature (see Global Time Base (GTB)).

FTM0 provides the only source for the FTM global time base. The other FTM modules can share the time base as shown in the following figure:

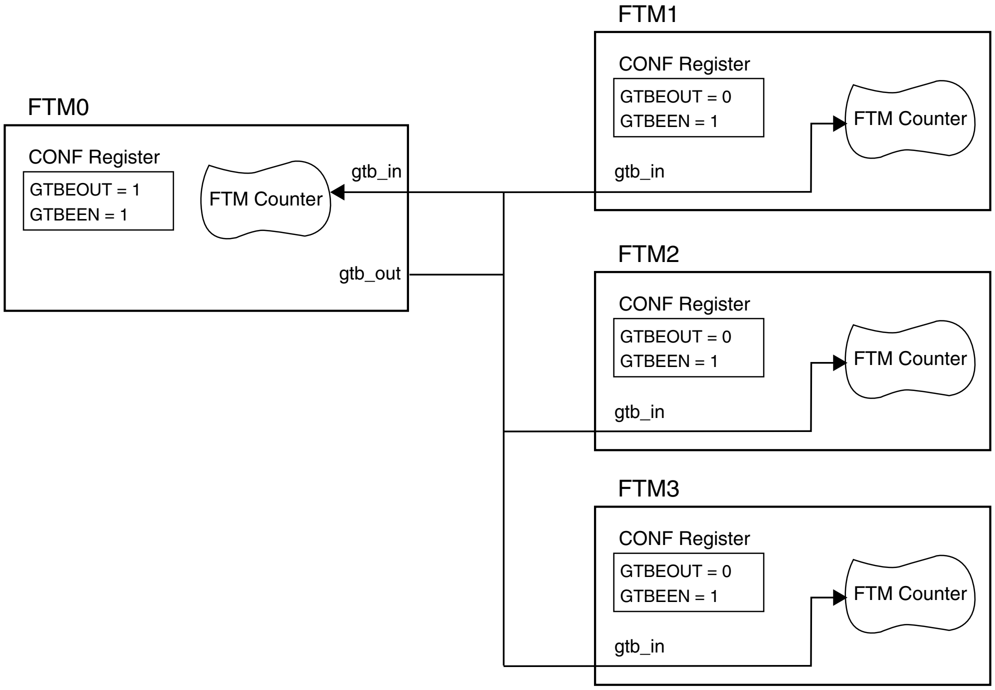


Figure 3-48. FTM Global Time Base Configuration

3.8.2.10 FTM BDM and debug halt mode

In the FTM chapter, references to the chip being in "BDM" are the same as the chip being in “debug halt mode".

3.8.3 PIT Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

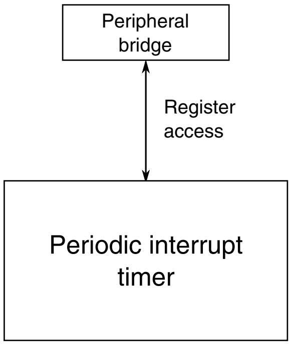


Figure 3-49. PIT configuration

Table 3-60. Reference links to related information

Topic	Related module	Reference
Full description	PIT	PIT
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management

3.8.3.1 PIT/DMA Periodic Trigger Assignments

The PIT generates periodic trigger events to the DMA Mux as shown in the table below.

Table 3-61. PIT channel assignments for periodic DMA triggering

DMA Channel Number	PIT Channel
DMA Channel 0	PIT Channel 0
DMA Channel 1	PIT Channel 1
DMA Channel 2	PIT Channel 2
DMA Channel 3	PIT Channel 3

3.8.3.2 PIT/ADC Triggers

PIT triggers are selected as ADCx trigger sources using the SOPT7[ADCxTRGSEL] bits in the SIM module. For more details, refer to SIM chapter.

3.8.4 Low-power timer configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

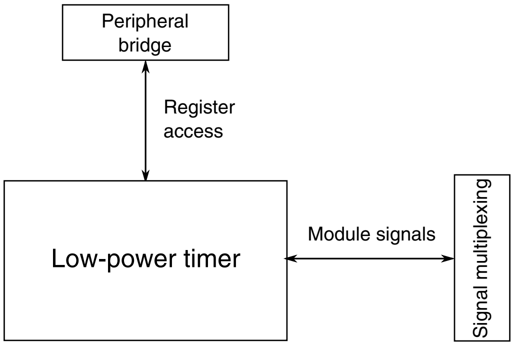


Figure 3-50. LPT configuration

Table 3-62. Reference links to related information

Topic	Related module	Reference
Full description	Low-power timer	Low-power timer
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management
Signal Multiplexing	Port control	Signal Multiplexing

3.8.4.1 LPTMR prescaler/glitch filter clocking options

The prescaler and glitch filter of the LPTMR module can be clocked from one of four sources determined by the LPTMR0_PSR[PCS] bitfield. The following table shows the chip-specific clock assignments for this bitfield.

NOTE

The chosen clock must remain enabled if the LPTMR is to continue operating in all required low-power modes.

LPTMR0_PSR[PCS]	Prescaler/glitch filter clock number	Chip clock
00	0	MCGIRCLK — internal reference clock (not available in VLPS/LLS/VLLS modes)
01	1	LPO — 1 kHz clock
10	2	ERCLK32K — secondary external reference clock
11	3	OSC0ERCLK — external reference clock

See Clock Distribution for more details on these clocks.

3.8.4.2 LPTMR pulse counter input options

The LPTMR_CSR[TPS] bitfield configures the input source used in pulse counter mode. The following table shows the chip-specific input assignments for this bitfield.

LPTMR_CSR[TPS]	Pulse counter input number	Chip input
00	0	CMP0 output
01	1	LPTMR_ALT1 pin
10	2	LPTMR_ALT2 pin
11	3	Reserved

3.8.5 CMT Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

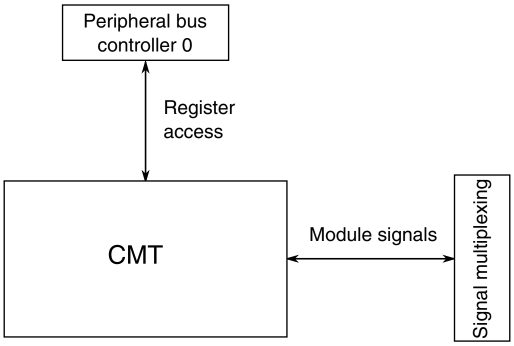


Figure 3-51. CMT configuration

Table 3-63. Reference links to related information

Topic	Related module	Reference
Full description	Carrier modulator transmitter (CMT)	CMT
System memory map		System memory map
Clocking		Clock distribution
Power management		Power management
Signal multiplexing	Port control	Signal multiplexing

3.8.5.1 Instantiation Information

This device contains one CMT module.

3.8.5.2 IRO Drive Strength

The IRO pad requires higher current drive than can be obtained from a single pad. For this device, the pin associated with the CMT_IRO signal is doubled bonded to two pads.

The SOPT2[CMTUARTPAD] field in SIM module can be used to configure the pin associated with the CMT_IRO signal as a higher current output port pin.

3.8.6 RTC configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

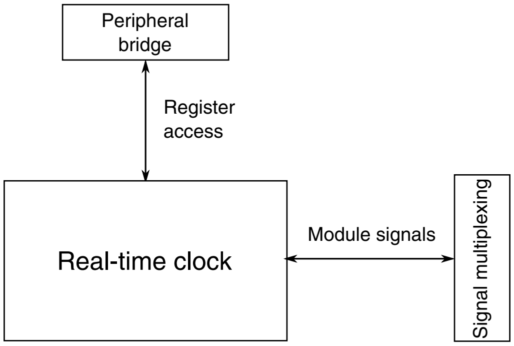


Figure 3-52. RTC configuration

Table 3-64. Reference links to related information

Topic	Related module	Reference
Full description	RTC	RTC
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management

3.8.6.1 RTC_CLKOUT signal

When the RTC is enabled and the port control module selects the RTC_CLKOUT function, the RTC_CLKOUT signal outputs a 1 Hz or 32 kHz output derived from RTC oscillator as shown below.

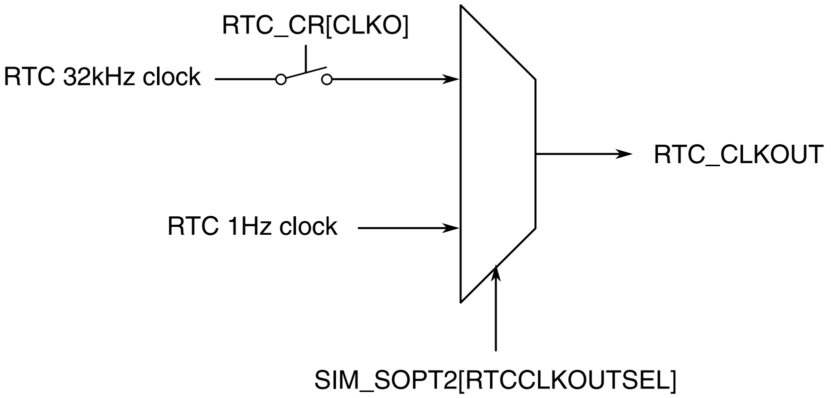


Figure 3-53. RTC_CLKOUT generation

3.9 Communication interfaces

3.9.1 Universal Serial Bus (USB) FS Subsystem

The USB FS subsystem includes these components:

- Dual-role USB OTG-capable (On-The-Go) controller that supports a full-speed (FS) device or FS/LS host. The module complies with the USB 2.0 specification.
- USB transceiver that includes internal 15 kΩ pulldowns on the D+ and D- lines for host mode functionality.
- A 3.3 V regulator.
- USB device charger detection module.
- VBUS detect signal: To detect a valid VBUS in device mode, use a GPIO signal that can wake the chip in all power modes.

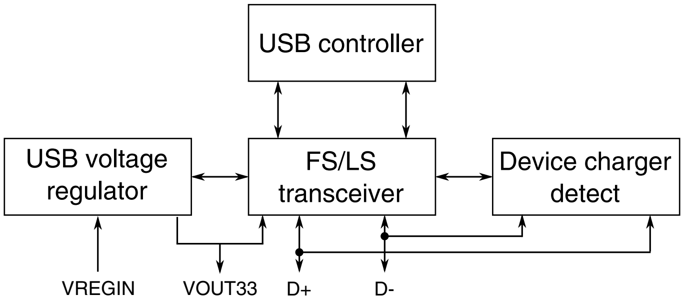


Figure 3-54. USB Subsystem Overview

3.9.1.1 USB Wakeup

When the USB detects that there is no activity on the USB bus for more than 3 ms, the INT_STAT[SLEEP] bit is set. This bit can cause an interrupt and software decides the appropriate action.

Waking from a low power mode (except in LLS/VLLS mode where USB is not powered) occurs through an asynchronous interrupt triggered by activity on the USB bus. Setting the USBTRC0[USBRESMEN] bit enables this function.

3.9.1.2 USB Power Distribution

This chip includes an internal 5 V to 3.3 V USB regulator that powers the USB transceiver or the MCU (depending on the application).

3.9.1.2.1 AA/AAA cells power supply

The chip can be powered by two AA/AAA cells. In this case, the MCU is powered through VDD which is within the 1.8 to 3.0 V range. After USB cable insertion is detected, the USB regulator is enabled to power the USB transceiver.

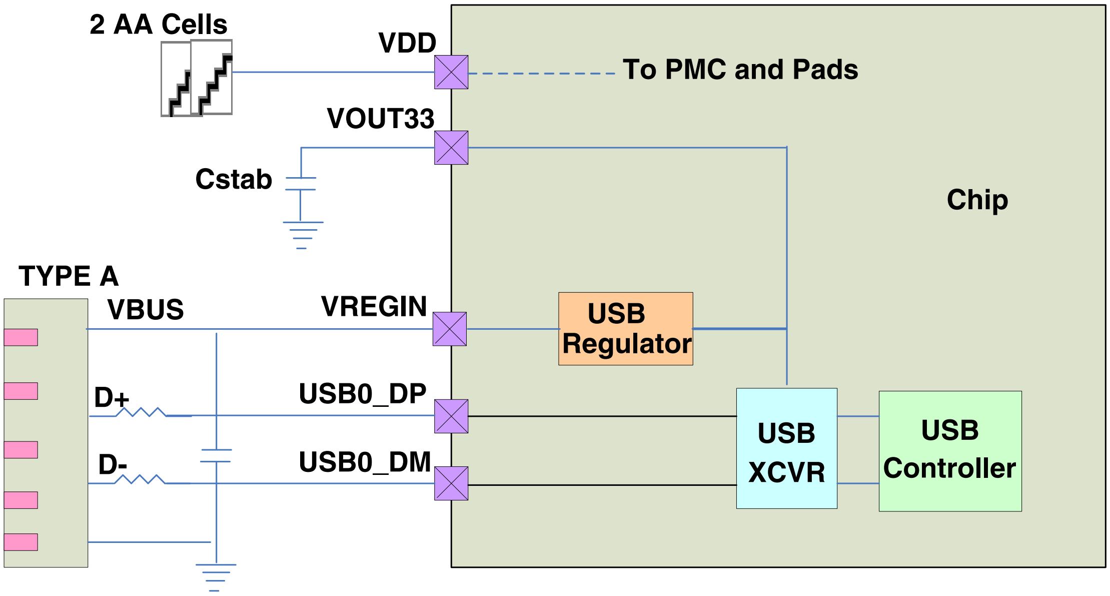


Figure 3-55. USB regulator AA cell usecase

3.9.1.2.2 Li-Ion battery power supply

The chip can also be powered by a single Li-ion battery. In this case, VOUT33 is connected to VDD. The USB regulator must be enabled by default to power the MCU. When connected to a USB host, the input source of this regulator is switched to the USB bus supply from the Li-ion battery. To charge the battery, the MCU can configure the battery charger according to the charger detection information.

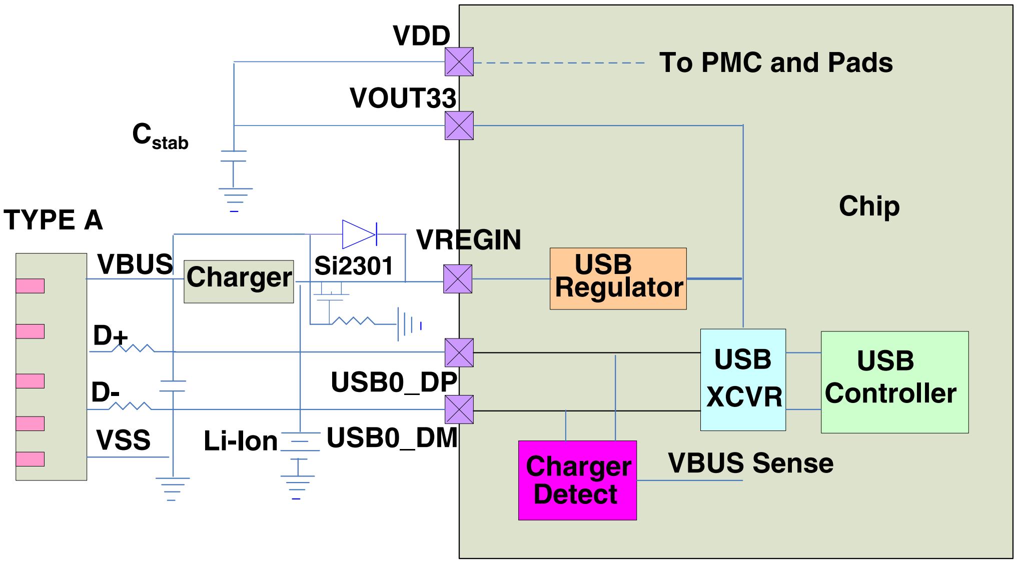


Figure 3-56. USB regulator Li-ion usecase

3.9.1.2.3 USB bus power supply

The chip can also be powered by the USB bus directly. In this case, VOUT33 is connected to VDD. The USB regulator must be enabled by default to power the MCU, then to power USB transceiver or external sensor.

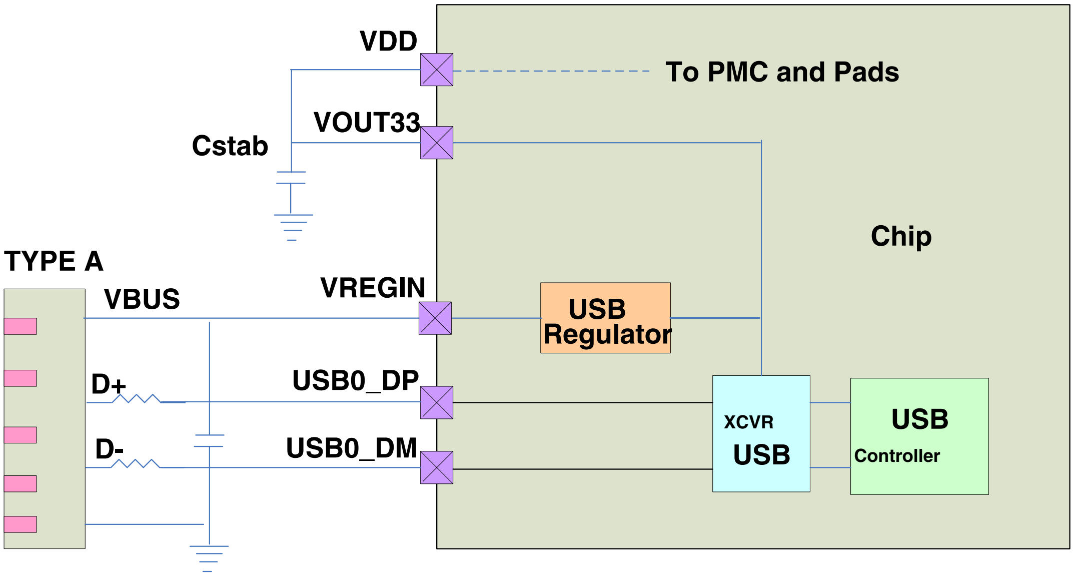


Figure 3-57. USB regulator bus supply

3.9.1.3 USB power management

The regulator should be put into STANDBY mode whenever the chip is in Stop mode.

3.9.1.4 USB controller configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

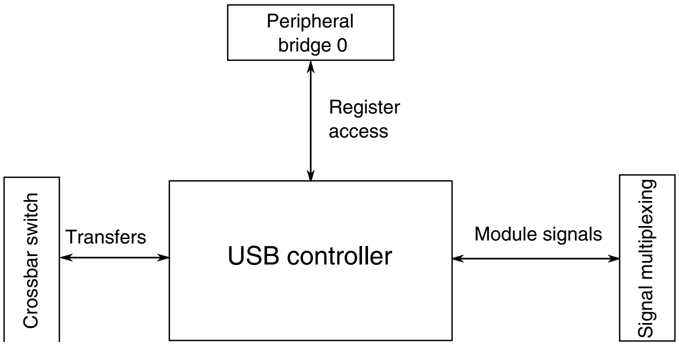


Figure 3-58. USB controller configuration

Table 3-65. Reference links to related information

Topic	Related module	Reference
Full description	USB controller	USB controller

Table continues on the next page...

Table 3-65. Reference links to related information (continued)

Topic	Related module	Reference
System memory map		System memory map
Clocking		Clock Distribution
Transfers	Crossbar switch	Crossbar switch
Signal Multiplexing	Port control	Signal Multiplexing

NOTE

When USB is not used in the application, it is recommended that the USB regulator VREGIN and VOUT33 pins remain floating.

3.9.1.5 USB DCD Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

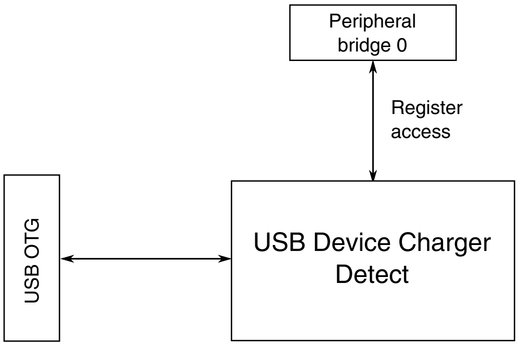


Figure 3-59. USB DCD configuration

Table 3-66. Reference links to related information

Topic	Related module	Reference
Full description	USB DCD	USB DCD
System memory map		System memory map
Clocking		Clock Distribution
	USB controller	USB controller

3.9.1.6 USB Voltage Regulator Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

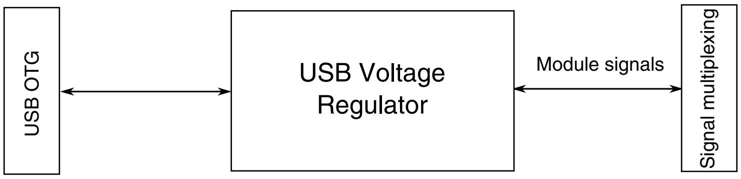


Figure 3-60. USB Voltage Regulator configuration

Table 3-67. Reference links to related information

Topic	Related module	Reference
Full description	USB Voltage Regulator	USB Voltage Regulator
System memory map		System memory map
Clocking		Clock Distribution
	USB controller	USB controller
Signal Multiplexing	Port control	Signal Multiplexing

NOTE

When USB is not used in the application, it is recommended that the USB regulator VREGIN and VOUT33 pins remain floating.

3.9.2 USB HS controller configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

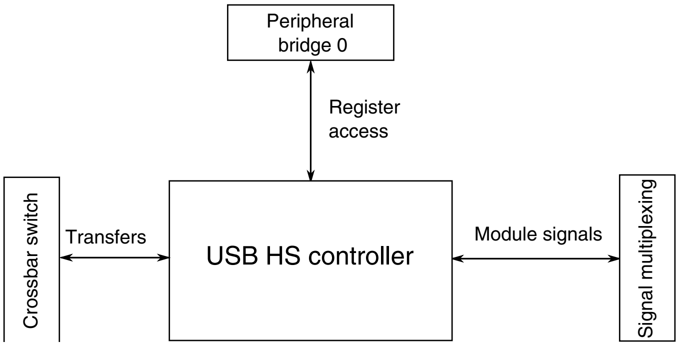


Figure 3-61. USB HS controller configuration

Table 3-68. Reference links to related information

Topic	Related module	Reference
Full description	USB HS controller	USB HS controller
System memory map		System memory map

Table continues on the next page...

Table 3-68. Reference links to related information (continued)

Topic	Related module	Reference
Clocking		Clock Distribution
Transfers	Crossbar switch	Crossbar switch
Signal Multiplexing	Port control	Signal Multiplexing

3.9.2.1 ULPI interface

To operate the USB HS OTG controller, connect the USB HS controller to an external ULPI PHY/transceiver via the ULPI interface.

The following figure illustrates the connection of the ULPI transceiver. The ULPI transceiver is an implementation of the HS/FS/LS physical layer which encapsulates the 60+ pin UTMI+ interface using a 12-pin digital interface.

The board-level implementation of a ULPI-based product is dependent on the PHY vendor. The ULPI PHY manages USB clocking, DP/DM bias resistors, and the OTG VBUS charge pump. For OTG applications requiring full host power (100 – 500 mA downstream current), an additional USB power-switch chip may be used. This OTG configuration may be used as a USB device, host, or dual-role device under firmware control.

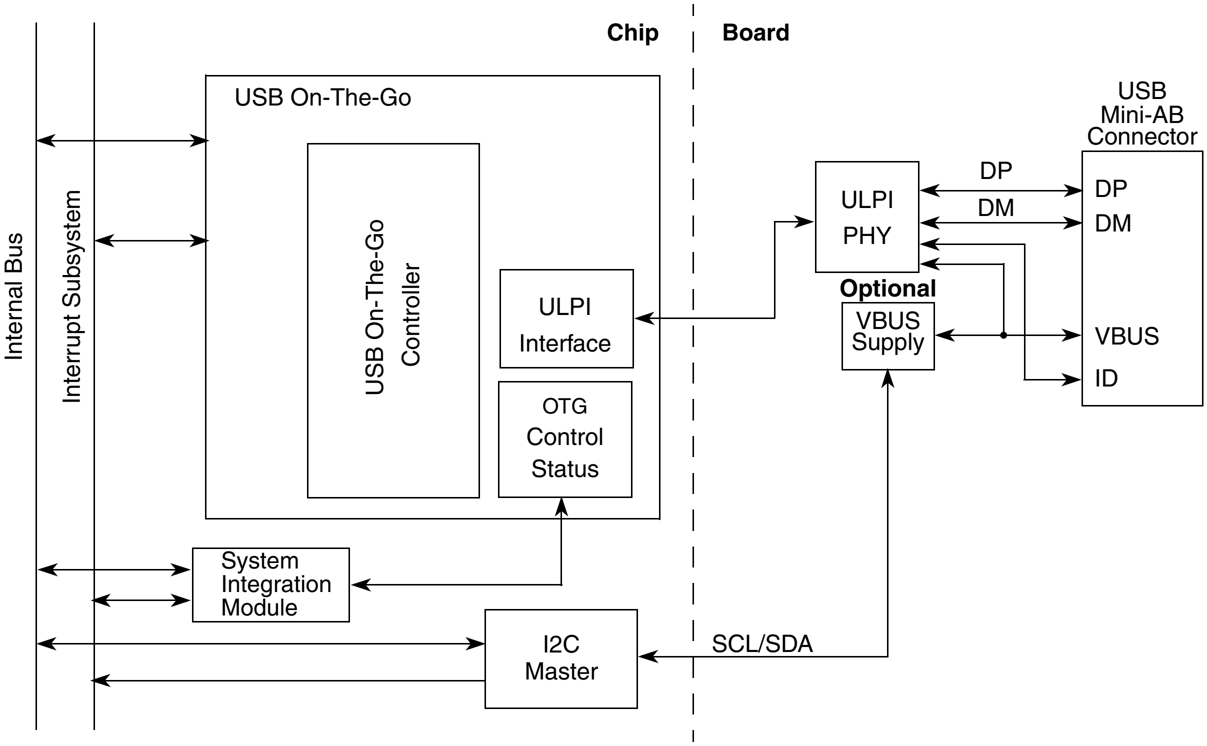


Figure 3-62. USB HS OTG module and ULPI PHY

3.9.3 CAN Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

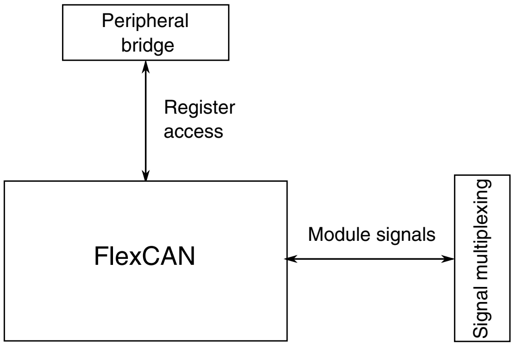


Figure 3-63. CAN configuration

Table 3-69. Reference links to related information

Topic	Related module	Reference
Full description	CAN	CAN
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management
Signal Multiplexing	Port control	Signal Multiplexing

3.9.3.1 Number of FlexCAN modules

This device contains 2 identical FlexCAN modules.

3.9.3.2 Reset value of MDIS bit

The CAN_MCR[MDIS] bit is set after reset. Therefore, FlexCAN module is disabled following a reset.

3.9.3.3 Number of message buffers

Each FlexCAN module contains 16 message buffers. Each message buffer is 16 bytes.

3.9.3.4 FlexCAN Clocking

3.9.3.4.1 Clocking Options

The FlexCAN module has a register bit CANCTRL[CLK_SRC] that selects between clocking the FlexCAN from the internal bus clock or the input clock (EXTAL).

3.9.3.4.2 Clock Gating

The clock to each CAN module can be gated on and off using the SCGC*n*[CAN*x*] bits. These bits are cleared after any reset, which disables the clock to the corresponding module. The appropriate clock enable bit should be set by software at the beginning of the FlexCAN initialization routine to enable the module clock before attempting to initialize any of the FlexCAN registers.

3.9.3.5 FlexCAN Interrupts

The FlexCAN has multiple sources of interrupt requests. However, some of these sources are OR'd together to generate a single interrupt request. See below for the mapping of the individual interrupt sources to the interrupt request:

Request	Sources
Message buffer	Message buffers 0-15
Bus off	Bus off
Error	• Bit1 error • Bit0 error • Acknowledge error • Cyclic redundancy check (CRC) error • Form error • Stuffing error • Transmit error warning • Receive error warning
Transmit Warning	Transmit Warning
Receive Warning	Receive Warning
Wake-up	Wake-up

3.9.3.6 FlexCAN Operation in Low Power Modes

The FlexCAN module is operational in VLPR and VLPW modes. With the 2 MHz bus clock, the fastest supported FlexCAN transfer rate is 256 kbps. The bit timing parameters in the module must be adjusted for the new frequency, but full functionality is possible.

The FlexCAN module can be configured to generate a wakeup interrupt in STOP and VLPS modes. When the FlexCAN is configured to generate a wakeup, a recessive to dominant transition on the CAN bus generates an interrupt.

3.9.3.7 FlexCAN Doze Mode

The Doze mode for the FlexCAN module is the same as the Wait and VLPW modes for the chip.

3.9.4 SPI configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

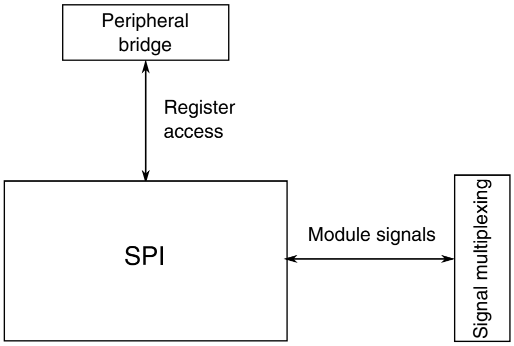


Figure 3-64. SPI configuration

Table 3-70. Reference links to related information

Topic	Related module	Reference
Full description	SPI	SPI
System memory map		System memory map
Clocking		Clock Distribution
Signal Multiplexing	Port control	Signal Multiplexing

3.9.4.1 SPI Modules Configuration

This device contains three SPI modules.

3.9.4.2 SPI clocking

The SPI module is clocked by the internal bus clock (the DSPI refers to it as system clock). The module has an internal divider, with a minimum divide is two. So, the SPI can run at a maximum frequency of bus clock/2.

3.9.4.3 Number of CTARs

SPI CTAR registers define different transfer attribute configurations. The SPI module supports up to eight CTAR registers. This device supports two CTARs on all instances of the SPI.

In master mode, the CTAR registers define combinations of transfer attributes, such as frame size, clock phase, clock polarity, data bit ordering, baud rate, and various delays. In slave mode only CTAR0 is used, and a subset of its bitfields sets the slave transfer attributes.

3.9.4.4 TX FIFO size

Table 3-71. SPI transmit FIFO size

SPI Module	Transmit FIFO size
SPI0	4
SPI1	4
SPI2	4

3.9.4.5 RX FIFO Size

SPI supports up to 16-bit frame size during reception.

Table 3-72. SPI receive FIFO size

SPI Module	Receive FIFO size
SPI0	4
SPI1	4
SPI2	4

3.9.4.6 Number of PCS signals

The following table shows the number of peripheral chip select signals available per SPI module.

Table 3-73. SPI PCS signals

SPI Module	PCS Signals
SPI0	SPI_PCS[5:0]
SPI1	SPI_PCS[3:0]
SPI2	SPI_PCS[1:0]

3.9.4.7 SPI Operation in Low Power Modes

In VLPR and VLPW modes the SPI is functional; however, the reduced system frequency also reduces the max frequency of operation for the SPI. In VLPR and VLPW modes the max SPI_CLK frequency is 2MHz.

In stop and VLPS modes, the clocks to the SPI module are disabled. The module is not functional, but it is powered so that it retains state.

There is one way to wake from stop mode via the SPI, which is explained in the following section.

3.9.4.7.1 Using GPIO Interrupt to Wake from stop mode

Here are the steps to use a GPIO to create a wakeup upon reception of SPI data in slave mode:

1. Point the GPIO interrupt vector to the desired interrupt handler.
2. Enable the GPIO input to generate an interrupt on either the rising or falling edge (depending on the polarity of the chip select signal).
3. Enter Stop or VLPS mode and Wait for the GPIO interrupt.

NOTE

It is likely that in using this approach the first word of data from the SPI host might not be received correctly. This is dependent on the transfer rate used for the SPI, the delay between chip select assertion and presentation of data, and the system interrupt latency.

3.9.4.8 SPI Doze Mode

The Doze mode for the SPI module is the same as the Wait and VLPW modes for the chip.

3.9.4.9 SPI Interrupts

The SPI has multiple sources of interrupt requests. However, these sources are OR'd together to generate a single interrupt request per SPI module to the interrupt controller. When an SPI interrupt occurs, read the SPI_SR to determine the exact interrupt source.

3.9.4.10 SPI clocks

This table shows the SPI module clocks and the corresponding chip clocks.

Table 3-74. SPI clock connections

Module clock	Chip clock
System Clock	Bus Clock

3.9.5 I2C Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

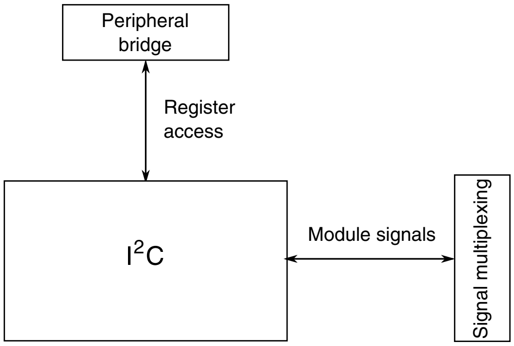


Figure 3-65. I2C configuration

Table 3-75. Reference links to related information

Topic	Related module	Reference
Full description	I^{2}C	I^{2}C
System memory map		System memory map

Table continues on the next page...

Table 3-75. Reference links to related information (continued)

Topic	Related module	Reference
Clocking		Clock Distribution
Power management		Power management
Signal Multiplexing	Port control	Signal Multiplexing

3.9.5.1 Number of I2C modules

This device has two I^{2}C modules.

3.9.6 UART Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

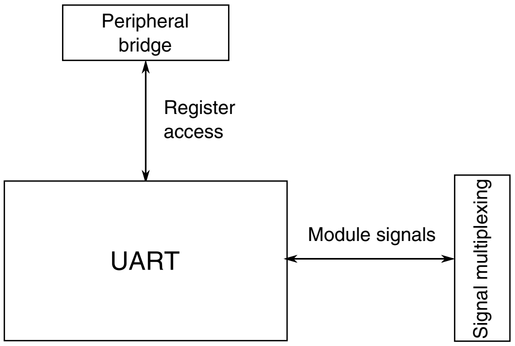


Figure 3-66. UART configuration

Table 3-76. Reference links to related information

Topic	Related module	Reference
Full description	UART	UART
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management
Signal Multiplexing	Port control	Signal Multiplexing

3.9.6.1 UART configuration information

This device contains six UART modules. This section describes how each module is configured on this device.

1. Standard features of all UARTs:
   - RS-485 support
   - Hardware flow control (RTS/CTS)
   - 9-bit UART to support address mark with parity
   - MSB/LSB configuration on data
2. UART0 and UART1 are clocked from the core clock, the remaining UARTs are clocked on the bus clock. The maximum baud rate is 1/16 of related source clock frequency.
3. IrDA is available on all UARTs
4. UART0 contains the standard features plus ISO7816
5. UART1 contains the standard features plus ISO7816
6. AMR support on all UARTs. The pin control and interrupts (PORT) module supports open-drain for all I/O.
7. UART0 and UART1 contains 8-entry transmit and 8-entry receive FIFOs
8. All other UARTs contain a 1-entry transmit and receive FIFOs
9. CEA709.1-B (LON) is available in UART0

3.9.6.2 UART wakeup

The UART can be configured to generate an interrupt/wakeup on the first active edge that it receives.

3.9.6.3 UART interrupts

The UART has multiple sources of interrupt requests. However, some of these sources are OR'd together to generate a single interrupt request. See below for the mapping of the individual interrupt sources to the interrupt request:

The status interrupt combines the following interrupt sources:

Source	UART 0	UART 1	UART 2	UART 3	UART 4	UART 5
Transmit data empty	x	x	x	x	x	x
Transmit complete	x	x	x	x	x	x
Idle line	x	x	x	x	x	x
Receive data full	x	x	x	x	x	x
LIN break detect	x	x	x	x	x	x

Table continues on the next page...

Source	UART 0	UART 1	UART 2	UART 3	UART 4	UART 5
RxD pin active edge	x	x	x	x	x	x
Initial character detect	x	x	—	—	—	—

The error interrupt combines the following interrupt sources:

Source	UART 0	UART 1	UART 2	UART 3	UART 4	UART 5
Receiver overrun	x	x	x	x	x	x
Noise flag	x	x	x	x	x	x
Framing error	x	x	x	x	x	x
Parity error	x	x	x	x	x	x
Transmitter buffer overflow	x	x	x	x	x	x
Receiver buffer underflow	x	x	x	x	x	x
Transmit threshold (ISO7816)	x	x	—	—	—	—
Receiver threshold (ISO7816)	x	x	—	—	—	—
Wait timer (ISO7816)	x	x	—	—	—	—
Character wait timer (ISO7816)	x	x	—	—	—	—
Block wait timer (ISO7816)	x	x	—	—	—	—
Guard time violation (ISO7816)	x	x	—	—	—	—
Wbase expire after beta1 time slots (LON)	x		—	—	—	—
Package received (LON)	x		—	—	—	—
Package transmitted (LON)	x		—	—	—	—
Package cycle time expired (LON)	x		—	—	—	—
Preamble start (LON)	x		—	—	—	—
Transmission fail (LON)	x		—	—	—	—

3.9.7 SDHC Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

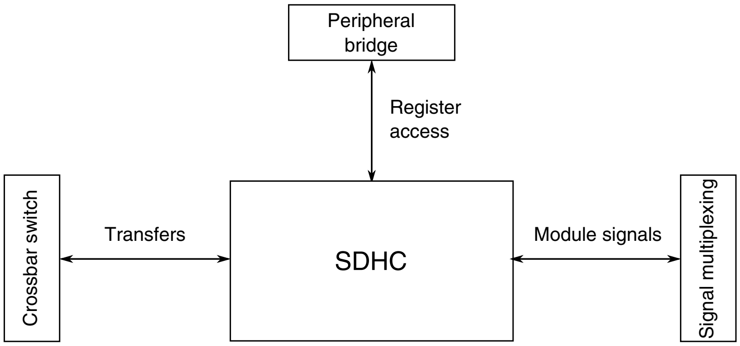


Figure 3-67. SDHC configuration

Table 3-77. Reference links to related information

Topic	Related module	Reference
Full description	SDHC	SDHC
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management
Transfers	Crossbar switch	Crossbar switch
Signal Multiplexing	Port control	Signal Multiplexing

3.9.7.1 SDHC clocking

In addition to the system clock, the SDHC needs a clock for the base for the external card clock. There are four possible clock sources for this clock, selected by the SIM's SOPT2 register:

- Core/system clock
- MCGPLLCLK or MCGFLLCLK
- EXTAL
- Bypass clock from off-chip (SDHC0_CLKIN)

3.9.7.2 SD bus pullup/pulldown constraints

The SD standard requires the SD bus signals (except the SD clock) to be pulled up during data transfers. The SDHC also provides a feature of detecting card insertion/removal, by detecting voltage level changes on DAT[3] of the SD bus. To support this DAT[3] must be pulled down. To avoid a situation where the SDHC detects voltage changes due to normal data transfers on the SD bus as card insertion/removal, the interrupt relating to this event must be disabled after the card has been inserted and detected. It can be re-enabled after the card is removed.

3.9.8 I^{2}S configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

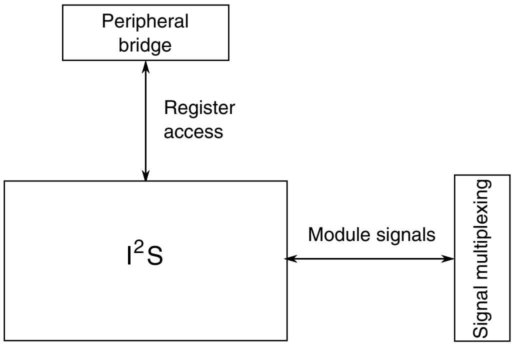


Figure 3-68. I^{2}S configuration

Table 3-78. Reference links to related information

Topic	Related module	Reference
Full description	I^{2}S	I2S
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management
Signal multiplexing	Port control	Signal Multiplexing

3.9.8.1 Instantiation information

This device contains two I^{2}S modules.

As configured on the device, module features include:

- TX data lines per module: 2
- RX data lines per module: 2

- FIFO size (words): 8
- Maximum words per frame: 32
- Maximum bit clock divider: 512

3.9.8.2 I^{2}S/SAI clocking

3.9.8.2.1 Audio Master Clock

The audio master clock (MCLK) is used to generate the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The audio master clock can also be output to or input from a pin. The transmitter and receiver have the same audio master clock inputs.

3.9.8.2.2 Bit Clock

The I^{2}S/SAI transmitter and receiver support asynchronous bit clocks (BCLKs) that can be generated internally from the audio master clock or supplied externally. The module also supports the option for synchronous operation between the receiver and transmitter or between two separate I^{2}S/SAI peripherals.

3.9.8.2.3 Bus Clock

The bus clock is used by the control registers and to generate synchronous interrupts and DMA requests.

3.9.8.2.4 I^{2}S/SAI clock generation

Each SAI peripheral can control the input clock selection, pin direction and divide ratio of one audio master clock.

The MCLK Input Clock Select bit of the MCLK Control Register (MCR[MICS]) selects the clock input to the I^{2}S/SAI module's MCLK divider.

On this device, I2S0 and I2S1 have identical input clock selection options.

Table 3-79. I2S0 and I2S1 MCLK input clock selection

MCR[MICS]	Clock Selection
00	System clock
01	OSC0ERCLK
10	OSC1ERCLK

Table continues on the next page...

Table 3-79. I2S0 and I2S1 MCLK input clock selection (continued)

MCR[MICS]	Clock Selection
11	MCGPLLCLK

The module's MCLK Divide Register (MDR) configures the MCLK divide ratio.

The module's MCLK Output Enable bit of the MCLK Control Register (MCR[MOE]) controls the direction of the MCLK pin. The pin is the input from the pin when MOE is 0, and the pin is the output from the clock divider when MOE is 1.

The transmitter and receiver can independently select between the bus clock and the audio master clocks to generate the bit clock. Each module's Clocking Mode field of the Transmit Configuration 2 Register and Receive Configuration 2 Register (TCR2[MSEL] and RCR2[MSEL]) selects the master clock.

The following tables show the TCR2[MSEL] and RCR2[MSEL] field settings for the module instances on this device.

Table 3-80. I2S0 master clock settings

TCR2[MSEL], RCR2[MSEL]	Master Clock
00	Bus Clock
01	When MOE is 1: MCLK is generated internally from PLL, OSC, or system clock When MOE is 0: I2S0_MCLK
10	I2S1_MCLK
11	I2S1_MCLK

Table 3-81. I2S1 master clock settings

TCR2[MSEL], RCR2[MSEL]	Master Clock
00	Bus Clock
01	When MOE is 1: MCLK is generated internally from PLL, OSC, or system clock When MOE is 0: I2S1_MCLK
10	I2S0_MCLK
11	I2S0_MCLK

3.9.8.2.5 Clock gating and I^{2}S/SAI initialization

The clock to the I^{2}S/SAI module can be gated using a bit in the SIM. To minimize power consumption, these bits are cleared after any reset, which disables the clock to the corresponding module. The clock enable bit should be set by software at the beginning of the module initialization routine to enable the module clock before initialization of any of the I^{2}S/SAI registers.

3.9.8.3 Multiple SAI Synchronous Mode

Multiple SAI Synchronous Mode is supported in both I2S0 and I2S1.

- For I2S1 to use I2S0's bit clock and frame sync, configure I2S1 as synchronous to another SAI peripheral, and configure I2S0 as asynchronous.
- For I2S0 to use I2S1's bit clock and frame sync, configure I2S0 as synchronous to another SAI peripheral, and configure I2S1 as asynchronous.

Note that transmit and receive are configured independently.

3.9.8.4 I^{2}S/SAI operation in low power modes

3.9.8.4.1 Stop and very low power modes

In stop mode, the SAI transmitter and/or receiver can continue operating provided the 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 VLPS mode, the module behaves as it does in stop mode if VLPS mode is entered from run mode. However, if VLPS mode is entered from VLPR mode, the FIFO might underflow or overflow before wakeup from stop mode due to the limits in bus bandwidth. In VLPW and VLPR modes, the module is limited by the maximum bus clock frequencies.

When operating from an internally generated bit clock or Audio Master Clock that is disabled in stop modes:

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.

3.9.8.4.2 Low-leakage modes

When entering low-leakage 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.

3.10 Human-machine interfaces (HMI)

3.10.1 GPIO configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

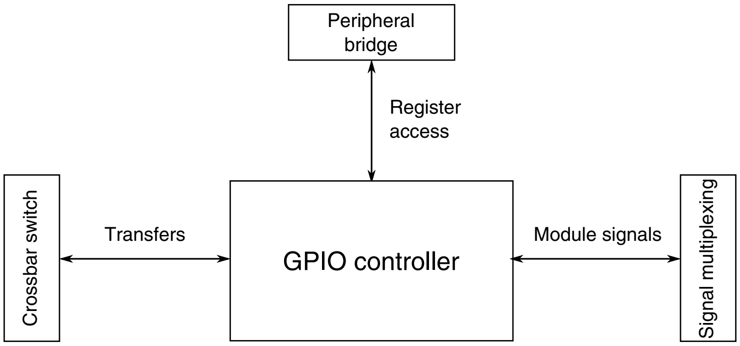


Figure 3-69. GPIO configuration

Table 3-82. Reference links to related information

Topic	Related module	Reference
Full description	GPIO	GPIO
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management
Transfers	Crossbar switch	Clock Distribution
Signal Multiplexing	Port control	Signal Multiplexing

3.10.1.1 GPIO access protection

The GPIO module does not have access protection because it is not connected to a peripheral bridge slot and is not protected by the MPU.

3.10.1.2 Number of GPIO signals

The number of GPIO signals available on the devices covered by this document are detailed in Orderable part numbers.

3.10.2 TSI Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module’s dedicated chapter.

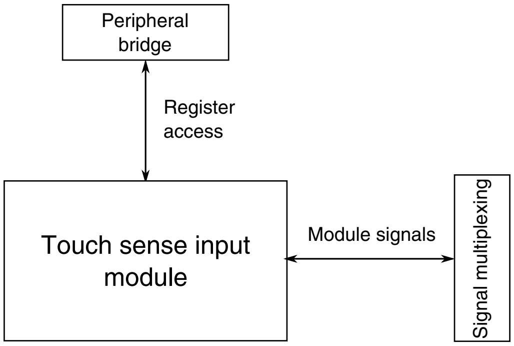


Figure 3-70. TSI configuration

Table 3-83. Reference links to related information

Topic	Related module	Reference
Full description	TSI	TSI
System memory map		System memory map
Clocking		Clock Distribution
Power management		Power management
Signal Multiplexing	Port control	Signal Multiplexing

3.10.2.1 Number of inputs

This device includes one TSI module containing 16 inputs. In low-power modes, one selectable pin is active.

3.10.2.2 TSI module functionality in MCU operation modes

Table 3-84. TSI module functionality in MCU operation modes

MCU operation mode	TSI clock sources	TSI operation mode when GENCS[TSIEN] is 1	Functional electrode pins	Required GENCS[STPE] state
Run	BUSCLK, MCGIRCLK, OSC0ERCLK	Active mode	All	Don't care
Wait	BUSCLK, MCGIRCLK, OSC0ERCLK	Active mode	All	Don't care
Stop	MCGIRCLK, OSC0ERCLK	Active mode	All	1
VLPR	BUSCLK, MCGIRCLK, OSC0ERCLK	Active mode	All	Don't care
VLPW	BUSCLK, MCGIRCLK, OSC0ERCLK	Active mode	All	Don't care
VLPS	OSC0ERCLK	Active mode	All	1
LLS	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1
VLLS3	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1
VLLS2	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1
VLLS1	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1

3.10.2.3 TSI clocks

This table shows the TSI clocks and the corresponding chip clocks.

Table 3-85. TSI clock connections

Module clock	Chip clock
BUSCLK	Bus clock
MCGIRCLK	MCGIRCLK
OSCERCLK	OSC0ERCLK
LPOCLK	1 kHz LPO clock
VLPOSCCLK	ERCLK32K

3.10.2.4 TSI Interrupts

The TSI has multiple sources of interrupt requests. However, these sources are OR'd together to generate a single interrupt request. When a TSI interrupt occurs, read the TSI status register to determine the exact interrupt source.

3.10.2.5 Shield drive signal

The shield drive signal is not supported on this device. Ignore this feature in the TSI chapter.

Chapter 4
Memory Map

4.1 Introduction

This device contains various memories and memory-mapped peripherals which are located in one 32-bit contiguous memory space. This chapter describes the memory and peripheral locations within that memory space.

4.2 System memory map

The following table shows the high-level device memory map.

Table 4-1. System memory map

System 32-bit Address Range	Destination Slave	Access	Slave Port
0x0000_0000–0x07FF_FFFF	Program flash and read-only data (Includes exception vectors in first 1024 bytes)	All masters	S0
0x0800_0000–0x0FFF_FFFF	DRAM Controller (Aliased Area)	Cortex-M4 core (M0) only	S5
0x1000_0000–0x13FF_FFFF	FlexNVM	All masters	S0
0x1000_0000–0x13FF_FFFF	Reserved	–	–
0x1400_0000–0x17FF_FFFF	For devices with FlexNVM: FlexRAM	All masters	S0
0x1400_0000–0x17FF_FFFF	For devices with program flash only: Programming acceleration RAM	–	S0
0x1800_0000–0x1BFF_FFFF	FlexBus (Aliased Area). 0x1800_0000-0x1BFF_FFFF are mapped to the same access space of 0x9800_0000-0x9BFF_FFFF.	Cortex-M4 core (M0) only	–
0x1C00_0000–0x1FFF_FFFF	SRAM_L: Lower SRAM (ICODE/DCODE)	All masters	–
0x2000_0000–0x200F_FFFF	SRAM_U: Upper SRAM bitband region	All masters	–
0x2010_0000–0x21FF_FFFF	Reserved	–	–

Table continues on the next page...

Table 4-1. System memory map (continued)

System 32-bit Address Range	Destination Slave	Access	Slave Port
0x2200_0000–0x23FF_FFFF	Aliased to TCMU SRAM bitband	Cortex-M4 core only	–
0x2400_0000–0x3FFF_FFFF	Reserved	–	–
0x4000_0000–0x4007_FFFF	Bitband region for AIPS0	Cortex-M4 core & DMA/EzPort	S2
0x4008_0000–0x400F_EFFF	Bitband region for AIPS1	Cortex-M4 core & DMA/EzPort	S3
0x400F_F000–0x400F_FFFF	Bitband region for GPIO	Cortex-M4 core & DMA/EzPort	S3
0x4010_0000–0x41FF_FFFF	Reserved	–	–
0x4200_0000–0x43FF_FFFF	Aliased to AIPS and GPIO bitband	Cortex-M4 core only	–
0x4400_0000–0x5FFF_FFFF	Reserved	–	–
0x6000_0000–0x6FFF_FFFF	Flexbus (External memory - Write-back)	All masters	S4
0x9000_0000–0x9FFF_FFFF	FlexBus (External memory - Write-through)	All masters	S4
0xA000_0000–0xDFFF_FFFF	FlexBus (External peripheral - not executable)	All masters	S4
0xE000_0000–0xE00F_FFFF	Private Peripherals	Cortex-M4 core only	–
0xE010_0000–0xFFFF_FFFF	Reserved	–	–

NOTE

1. EzPort master port is statically muxed with DMA master port. Access rights to AIPS-Lite peripheral bridges and general purpose input/output (GPIO) module address space is limited to the core, DMA, and EzPort.
2. ARM Cortex-M4 core access privileges also includes accesses via the debug interface.

4.2.1 Aliased bit-band regions

The SRAM_U, AIPS-Lite, and general purpose input/output (GPIO) module resources reside in the Cortex-M4 processor bit-band regions.

The processor also includes two 32 MB aliased bit-band regions associated with the two 1 MB bit-band spaces. Each 32-bit location in the 32 MB space maps to an individual bit in the bit-band region. A 32-bit write in the alias region has the same effect as a read-modify-write operation on the targeted bit in the bit-band region.

Bit 0 of the value written to the alias region determines what value is written to the target bit:

- Writing a value with bit 0 set writes a 1 to the target bit.
- Writing a value with bit 0 clear writes a 0 to the target bit.

A 32-bit read in the alias region returns either:

- a value of 0x0000_0000 to indicate the target bit is clear
- a value of 0x0000_0001 to indicate the target bit is set

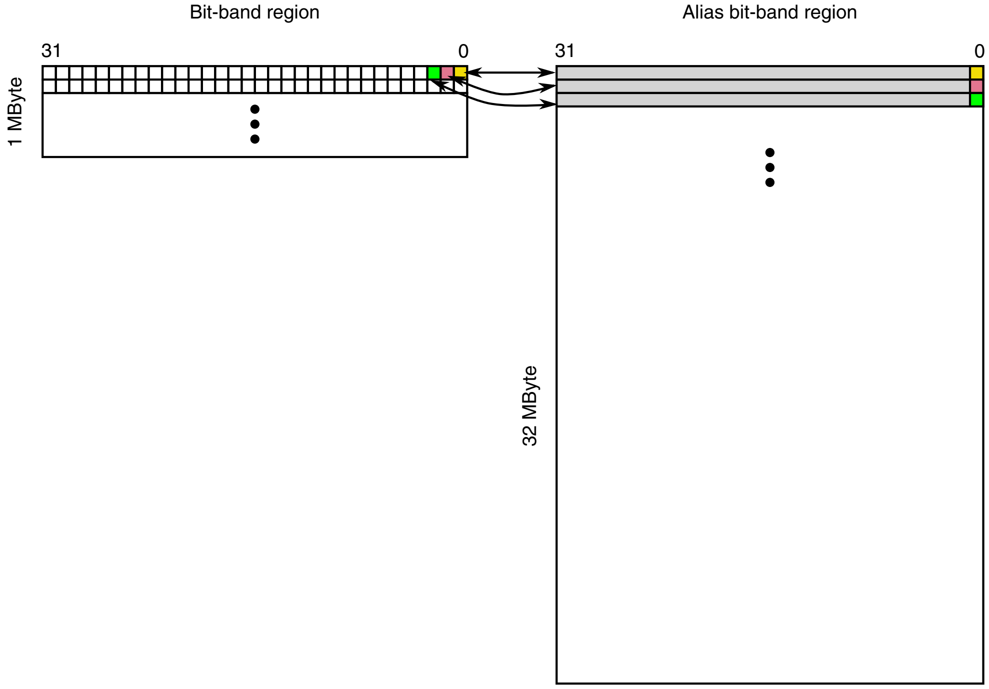


Figure 4-1. Alias bit-band mapping

NOTE

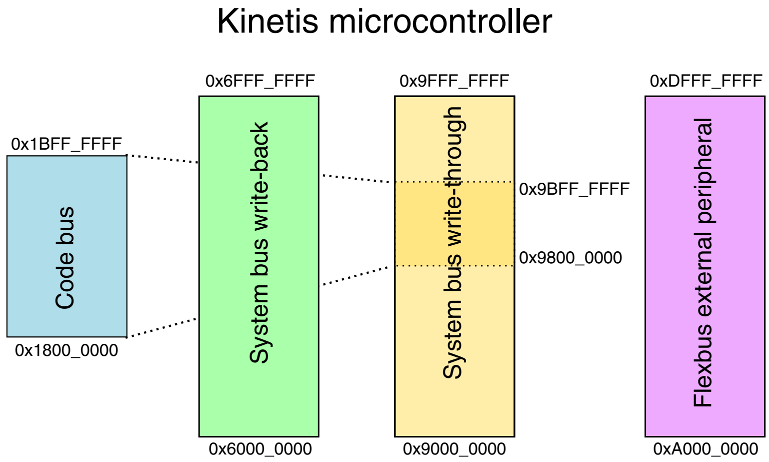
Each bit in bit-band region has an equivalent bit that can be manipulated through bit 0 in a corresponding long word in the alias bit-band region.

4.2.2 Aliased areas

The Flexbus has several aliased regions to allow the user to access the same physical area of memory with different cache attributes.

4.2.2.1 FlexBus aliased areas

For the FlexBus write-through, write-back and non-executable regions are available. The mapping from the internal memory map area to the external memory device is determined by the Flexbus register settings and the physical address signal connections.



Note: The aliased region accessible by the system bus is directly remapped from the I/D bus write-through region.

Figure 4-2. Flexbus aliased regions

NOTE

- In the proceeding figure, code bus alias region is mapped to system bus region:0x9800_0000-0x9BFF_FFFF.
- Set the FB_CSARx to 0x1800_0000 to use the address space from 0x9800_0000.

4.3 Flash Memory Map

The various flash memories and the flash registers are located at different base addresses as shown in the following figure. The base address for each is specified in System memory map.

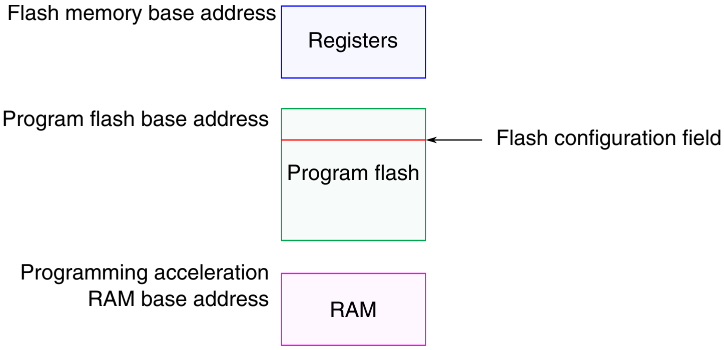

Flash memory base address
Registers
Program flash base address
Flash configuration field
Program flash
Programming acceleration
RAM base address
RAM

Figure 4-3. Flash memory map for devices containing only program flash

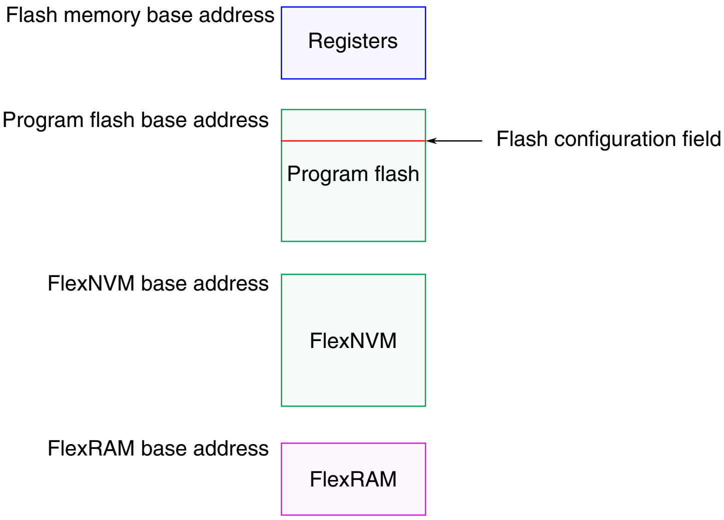

Flash memory base address
Registers
Program flash base address
Flash configuration field
Program flash
FlexNVM base address
FlexNVM
FlexRAM base address
FlexRAM

Figure 4-4. Flash memory map for devices containing FlexNVM

4.3.1 Alternate Non-Volatile IRC User Trim Description

The following non-volatile locations (4 bytes) are reserved for custom IRC user trim supported by some development tools. An alternate IRC trim to the factory loaded trim can be stored at this location. To override the factory trim, user software must load new values into the MCG trim registers.

Non-Volatile Byte Address	Alternate IRC Trim Value
0x0000_03FC	Reserved
0x0000_03FD	Reserved
0x0000_03FE (bit 0)	SCFTRIM
0x0000_03FE (bit 4:1)	FCTRIM
0x0000_03FF	SCTRIM

4.4 SRAM memory map

The on-chip RAM is split evenly among SRAM_L and SRAM_U. The RAM is also implemented such that the SRAM_L and SRAM_U ranges form a contiguous block in the memory map. See SRAM Arrays for details.

Accesses to the SRAM_L and SRAM_U memory ranges outside the amount of RAM on the device causes the bus cycle to be terminated with an error followed by the appropriate response in the requesting bus master.

4.5 Peripheral bridge (AIPS-Lite0 and AIPS-Lite1) memory maps

The peripheral memory map is accessible via two slave ports on the crossbar switch in the 0x4000_0000–0x400F_FFFF region. The device implements two peripheral bridges (AIPS-Lite 0 and 1):

- AIPS-Lite0 covers 512 KB
- AIPS-Lite1 covers 508 KB with 4 KB assigned to the general purpose input/output module (GPIO)

AIPS-Lite0 is connected to crossbar switch slave port 2, and is accessible at locations 0x4000_0000–0x4007_FFFF.

AIPS-Lite1 and the general purpose input/output module share the connection to crossbar switch slave port 3. The AIPS-Lite1 is accessible at locations 0x4008_0000–0x400F_EFFF. The general purpose input/output module is accessible in a 4-kbyte region at 0x400F_F000–0x400F_FFFF. Its direct connection to the crossbar switch provides master access without incurring wait states associated with accesses via the AIPS-Lite controllers.

Modules that are disabled via their clock gate control bits in the SIM registers disable the associated AIPS slots. Access to any address within an unimplemented or disabled peripheral bridge slot results in a transfer error termination.

For programming model accesses via the peripheral bridges, there is generally only a small range within the 4 KB slots that is implemented. Accessing an address that is not implemented in the peripheral results in a transfer error termination.

4.5.1 Peripheral Bridge 0 (AIPS-Lite 0) Memory Map

Table 4-2. Peripheral bridge 0 slot assignments

System 32-bit base address	Slot number	Module
0x4000_0000	0	Peripheral bridge 0 (AIPS-Lite 0)
0x4000_1000	1	—
0x4000_2000	2	—
0x4000_3000	3	—
0x4000_4000	4	Crossbar switch
0x4000_5000	5	—
0x4000_6000	6	—
0x4000_7000	7	—
0x4000_8000	8	DMA controller
0x4000_9000	9	DMA controller transfer control descriptors
0x4000_A000	10	—
0x4000_B000	11	—
0x4000_C000	12	FlexBus
0x4000_D000	13	MPU
0x4000_E000	14	—
0x4000_F000	15	—
0x4001_0000	16	—
0x4001_1000	17	—
0x4001_2000	18	—
0x4001_3000	19	—
0x4001_4000	20	—
0x4001_5000	21	—
0x4001_6000	22	—
0x4001_7000	23	—
0x4001_8000	24	—
0x4001_9000	25	—
0x4001_A000	26	—
0x4001_B000	27	—
0x4001_C000	28	—
0x4001_D000	29	—
0x4001_E000	30	—
0x4001_F000	31	Flash memory controller
0x4002_0000	32	Flash memory

Table continues on the next page...

Table 4-2. Peripheral bridge 0 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4002_1000	33	DMA channel mutiplexer 0
0x4002_2000	34	DMA channel mutiplexer 1
0x4002_3000	35	—
0x4002_4000	36	FlexCAN 0
0x4002_5000	37	—
0x4002_6000	38	—
0x4002_7000	39	—
0x4002_8000	40	—
0x4002_9000	41	—
0x4002_A000	42	—
0x4002_B000	43	—
0x4002_C000	44	SPI 0
0x4002_D000	45	SPI 1
0x4002_E000	46	—
0x4002_F000	47	I2S 0
0x4003_0000	48	—
0x4003_1000	49	—
0x4003_2000	50	CRC
0x4003_3000	51	—
0x4003_4000	52	USB OTG HS/FS/LS
0x4003_5000	53	USB DCD
0x4003_6000	54	Programmable delay block (PDB)
0x4003_7000	55	Periodic interrupt timers (PIT)
0x4003_8000	56	FlexTimer (FTM) 0
0x4003_9000	57	FlexTimer (FTM) 1
0x4003_A000	58	—
0x4003_B000	59	Analog-to-digital converter (ADC) 0
0x4003_C000	60	Analog-to-digital converter (ADC) 2
0x4003_D000	61	Real-time clock (RTC)
0x4003_E000	62	VBAT register file
0x4003_F000	63	
0x4004_0000	64	Low-power timer (LPTMR)
0x4004_1000	65	System register file
0x4004_2000	66	—

Table continues on the next page...

Table 4-2. Peripheral bridge 0 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4004_3000	67	—
0x4004_4000	68	—
0x4004_5000	69	Touch sense interface (TSI)
0x4004_6000	70	—
0x4004_7000	71	SIM low-power logic
0x4004_8000	72	System integration module (SIM)
0x4004_9000	73	Port A multiplexing control
0x4004_A000	74	Port B multiplexing control
0x4004_B000	75	Port C multiplexing control
0x4004_C000	76	Port D multiplexing control
0x4004_D000	77	Port E multiplexing control
0x4004_E000	78	Port F multiplexing control
0x4004_F000	79	—
0x4005_0000	80	—
0x4005_1000	81	—
0x4005_2000	82	Software watchdog
0x4005_3000	83	—
0x4005_4000	84	—
0x4005_5000	85	—
0x4005_6000	86	—
0x4005_7000	87	—
0x4005_8000	88	—
0x4005_9000	89	—
0x4005_A000	90	—
0x4005_B000	91	—
0x4005_C000	92	—
0x4005_D000	93	—
0x4005_E000	94	—
0x4005_F000	95	—
0x4006_0000	96	—
0x4006_1000	97	External watchdog
0x4006_2000	98	Carrier modulator timer (CMT)
0x4006_3000	99	—
0x4006_4000	100	Multi-purpose Clock Generator (MCG)

Table continues on the next page...

Table 4-2. Peripheral bridge 0 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4006_5000	101	System oscillator (OSC0)
0x4006_6000	102	I^{2}C 0
0x4006_7000	103	I^{2}C 1
0x4006_8000	104	
0x4006_9000	105	—
0x4006_A000	106	UART 0
0x4006_B000	107	UART 1
0x4006_C000	108	UART 2
0x4006_D000	109	UART 3
0x4006_E000	110	—
0x4006_F000	111	—
0x4007_0000	112	—
0x4007_1000	113	—
0x4007_2000	114	USB OTG FS/LS
0x4007_3000	115	Analog comparator (CMP) / 6-bit digital-to-analog converter (DAC)
0x4007_4000	116	Voltage reference (VREF)
0x4007_5000	117	—
0x4007_6000	118	—
0x4007_7000	119	—
0x4007_8000	120	—
0x4007_9000	121	—
0x4007_A000	122	—
0x4007_B000	123	—
0x4007_C000	124	Low-leakage wakeup unit (LLWU)
0x4007_D000	125	Power management controller (PMC)
0x4007_E000	126	System Mode controller (SMC)
0x4007_F000	127	Reset Control Module (RCM)

4.5.2 Peripheral Bridge 1 (AIPS-Lite 1) Memory Map

Table 4-3. Peripheral bridge 1 slot assignments

System 32-bit base address	Slot number	Module
0x4008_0000	0	Peripheral bridge 1 (AIPS-Lite 1)

Table continues on the next page...

Table 4-3. Peripheral bridge 1 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x4008_1000	1	—
0x4008_2000	2	—
0x4008_3000	3	—
0x4008_4000	4	—
0x4008_5000	5	—
0x4008_6000	6	—
0x4008_7000	7	—
0x4008_8000	8	—
0x4008_9000	9	—
0x4008_A000	10	—
0x4008_B000	11	—
0x4008_C000	12	—
0x4008_D000	13	—
0x4008_E000	14	—
0x4008_F000	15	—
0x4009_0000	16	—
0x4009_1000	17	—
0x4009_2000	18	—
0x4009_3000	19	—
0x4009_4000	20	—
0x4009_5000	21	—
0x4009_6000	22	—
0x4009_7000	23	—
0x4009_8000	24	—
0x4009_9000	25	—
0x4009_A000	26	—
0x4009_B000	27	—
0x4009_C000	28	—
0x4009_D000	29	—
0x4009_E000	30	—
0x4009_F000	31	—
0x400A_0000	32	—
0x400A_1000	33	—
0x400A_2000	34	—

Table continues on the next page...

Table 4-3. Peripheral bridge 1 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x400A_3000	35	—
0x400A_4000	36	FlexCAN 1
0x400A_5000	37	—
0x400A_6000	38	—
0x400A_7000	39	—
0x400A_8000	40	NAND flash controller
0x400A_9000	41	NAND flash controller
0x400A_A000	42	NAND flash controller
0x400A_B000	43	NAND flash controller
0x400A_C000	44	SPI 2
0x400A_D000	45	—
0x400A_E000	46	
0x400A_F000	47	I2S 1
0x400B_0000	48	—
0x400B_1000	49	SDHC
0x400B_2000	50	—
0x400B_3000	51	—
0x400B_4000	52	—
0x400B_5000	53	—
0x400B_6000	54	
0x400B_7000	55	—
0x400B_8000	56	FlexTimer (FTM) 2
0x400B_9000	57	FlexTimer (FTM) 3
0x400B_A000	58	—
0x400B_B000	59	Analog-to-digital converter (ADC) 1
0x400B_C000	60	Analog-to-digital converter (ADC) 3
0x400B_D000	61	—
0x400B_E000	62	—
0x400B_F000	63	—
0x400C_0000	64	—
0x400C_1000	65	—
0x400C_2000	66	—
0x400C_3000	67	—
0x400C_4000	68	—

Table continues on the next page...

Table 4-3. Peripheral bridge 1 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x400C_5000	69	—
0x400C_6000	70	—
0x400C_7000	71	—
0x400C_8000	72	—
0x400C_9000	73	—
0x400C_A000	74	—
0x400C_B000	75	—
0x400C_C000	76	12-bit digital-to-analog converter (DAC) 0
0x400C_D000	77	12-bit digital-to-analog converter (DAC) 1
0x400C_E000	78	—
0x400C_F000	79	—
0x400D_0000	80	—
0x400D_1000	81	—
0x400D_2000	82	—
0x400D_3000	83	—
0x400D_4000	84	—
0x400D_5000	85	—
0x400D_6000	86	—
0x400D_7000	87	—
0x400D_8000	88	—
0x400D_9000	89	—
0x400D_A000	90	—
0x400D_B000	91	—
0x400D_C000	92	—
0x400D_D000	93	—
0x400D_E000	94	—
0x400D_F000	95	—
0x400E_0000	96	—
0x400E_1000	97	—
0x400E_2000	98	—
0x400E_3000	99	—
0x400E_4000	100	—
0x400E_5000	101	System oscillator 1 (OSC1)
0x400E_6000	102	—

Table continues on the next page...

Table 4-3. Peripheral bridge 1 slot assignments (continued)

System 32-bit base address	Slot number	Module
0x400E_7000	103	—
0x400E_8000	104	—
0x400E_9000	105	—
0x400E_A000	106	UART 4
0x400E_B000	107	UART 5
0x400E_C000	108	—
0x400E_D000	109	—
0x400E_E000	110	—
0x400E_F000	111	—
0x400F_0000	112	—
0x400F_1000	113	—
0x400F_2000	114	—
0x400F_3000	115	—
0x400F_4000	116	—
0x400F_5000	117	—
0x400F_6000	118	—
0x400F_7000	119	—
0x400F_8000	120	—
0x400F_9000	121	—
0x400F_A000	122	—
0x400F_B000	123	—
0x400F_C000	124	—
0x400F_D000	125	—
0x400F_E000	126	—
0x400F_F000	Not an AIPS-Lite slot. The 32-bit general purpose input/output module that shares the crossbar switch slave port with the AIPS-Lite is accessed at this address.	

4.6 Private Peripheral Bus (PPB) memory map

The PPB is part of the defined ARM bus architecture and provides access to select processor-local modules. These resources are only accessible from the core; other system masters do not have access to them.

Table 4-4. PPB memory map

System 32-bit Address Range	Resource
0xE000_0000–0xE000_0FFF	Instrumentation Trace Macrocell (ITM)
0xE000_1000–0xE000_1FFF	Data Watchpoint and Trace (DWT)
0xE000_2000–0xE000_2FFF	Flash Patch and Breakpoint (FPB)
0xE000_3000–0xE000_DFFF	Reserved
0xE000_E000–0xE000_EFFF	System Control Space (SCS) (for NVIC and FPU)
0xE000_F000–0xE003_FFFF	Reserved
0xE004_0000–0xE004_0FFF	Trace Port Interface Unit (TPIU)
0xE004_1000–0xE004_1FFF	Embedded Trace Macrocell (ETM)
0xE004_2000–0xE004_2FFF	Embedded Trace Buffer (ETB)
0xE004_3000–0xE004_3FFF	Embedded Trace Funnel
0xE004_4000–0xE007_FFFF	Reserved
0xE008_0000–0xE008_0FFF	Miscellaneous Control Module (MCM)(including ETB Almost Full)
0xE008_1000–0xE008_1FFF	Reserved
0xE008_2000–0xE008_2FFF	Cache Controller
0xE008_3000–0xE00F_EFFF	Reserved
0xE00F_F000–0xE00F_FFFF	ROM Table - allows auto-detection of debug components

Chapter 5
Clock Distribution

5.1 Introduction

The MCG module controls which clock source is used to derive the system clocks. The clock generation logic divides the selected clock source into a variety of clock domains, including the clocks for the system bus masters, system bus slaves, and flash memory. The clock generation logic also implements module-specific clock gating to allow granular shutoff of modules.

The primary clocks for the system are generated from the MCGOUTCLK clock. The clock generation circuitry provides several clock dividers that allow different portions of the device to be clocked at different frequencies. This allows for trade-offs between performance and power dissipation.

Various modules, such as the USB OTG Controller, have module-specific clocks that can be generated from the MCGPLLCLK or MCGFLLCLK clock. In addition, there are various other module-specific clocks that have other alternate sources. Clock selection for most modules is controlled by the SOPT registers in the SIM module.

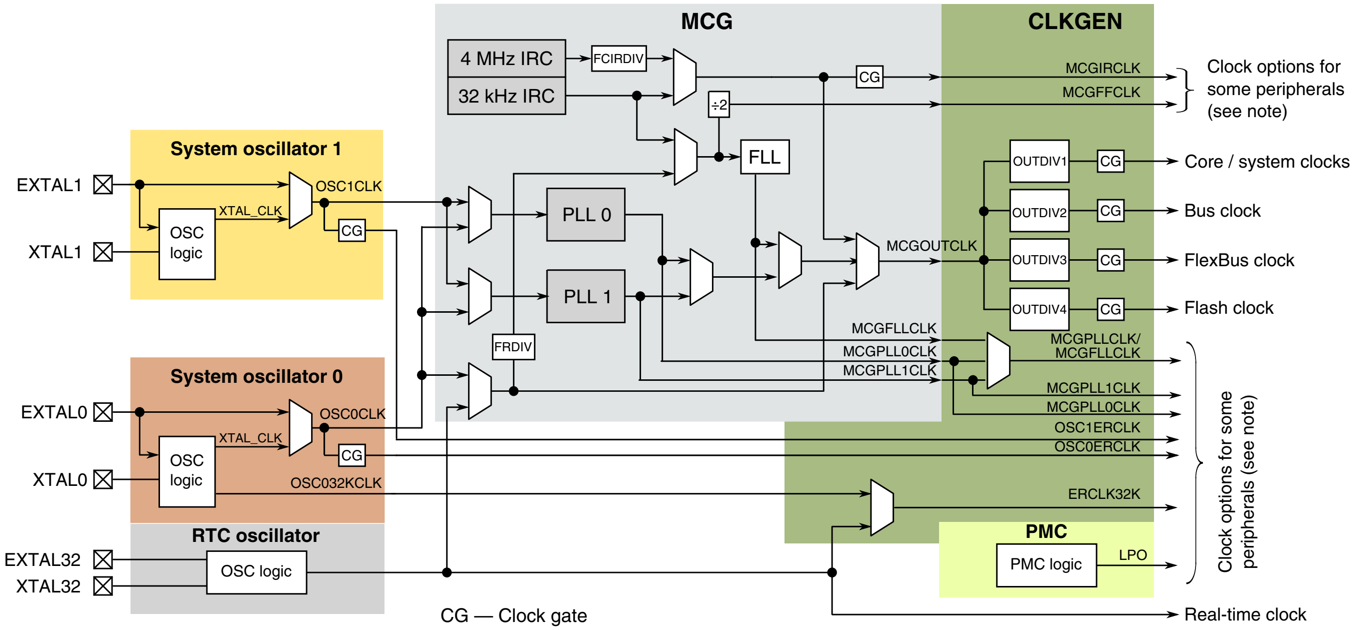
5.2 Programming model

The selection and multiplexing of system clock sources is controlled and programmed via the MCG module. The setting of clock dividers and module clock gating for the system are programmed via the SIM module. Reference those sections for detailed register and bit descriptions.

5.3 High-level device clocking diagram

The following system oscillator, MCG, and SIM module registers control the multiplexers, dividers, and clock gates shown in the below figure:

	OSC	MCG	SIM
Muliplexers	MCG_C*x*	MCG_C*x*	SIM_SOPT1, SIM_SOPT2
Dividers	—	MCG_C*x*	SIM_CLKDIV*x*
Clock gates	OSC_CR	MCG_C1	SIM_SCGC*x*



CG — Clock gate

Note: See subsequent sections for details on where these clocks are used.

Figure 5-1. Clocking diagram

5.4 Clock definitions

The following table describes the clocks in the previous block diagram.

Clock name	Description
Core clock	MCGOUTCLK divided by OUTDIV1 clocks the ARM Cortex-M4 core
System clock	MCGOUTCLK divided by OUTDIV1 clocks the crossbar switch and bus masters directly connected to the crossbar. In addition, this clock is used for UART0 and UART1.
Bus clock	MCGOUTCLK divided by OUTDIV2 clocks the bus slaves and peripheral (excluding memories)
FlexBus clock	MCGOUTCLK divided by OUTDIV3 clocks the external FlexBus interface
Flash clock	MCGOUTCLK divided by OUTDIV4 clocks the flash memory
MCGIRCLK	MCG output of the slow or fast internal reference clock
MCGFFCLK	MCG output of the slow internal reference clock or a divided MCG external reference clock.

Table continues on the next page...

Clock name	Description
MCGOUTCLK	MCG output of either IRC, MCGFLLCLK, MCGPLLCLK, or MCG's external reference clock that sources the core, system, bus, FlexBus, and flash clock. It is also an option for the debug trace clock.
MCGFLLCLK	MCG output of the FLL. MCGFLLCLK or MCGPLLCLK may clock some modules.
MCGPLL0CLK	MCG output of the PLL0. MCGFLLCLK or MCGPLLCLK may clock some modules.
MCGPLL1CLK	MCG output of the PLL1. MCGFLLCLK or MCGPLLCLK may clock some modules.
OSC0CLK, OSC1CLK	System oscillator output of the internal oscillators or sourced directly from EXTAL
OSC0ERCLK, OSC1ERCLK	System oscillator output sourced from OSC0CLK or OSC1CLKthat may clock some on-chip modules
OSC032KCLK	System oscillator 0 32kHz output
ERCLK32K	Clock source for some modules that is chosen as OSC0_32KCLK or the RTC clock
RTC clock	RTC oscillator output for the RTC module
LPO	PMC 1kHz output

5.4.1 Device clock summary

The following table provides more information regarding the on-chip clocks.

Table 5-1. Clock Summary

Clock name	Run mode clock frequency	VLPR mode clock frequency	Clock source	Clock is disabled when…
MCGOUTCLK	Up to 120 MHz	Up to 4 MHz	MCG	In all stop modes
Core clock	Up to 120 MHz	Up to 4 MHz	MCGOUTCLK clock divider	In all wait and stop modes
System clock	Up to 120 MHz	Up to 4 MHz	MCGOUTCLK clock divider	In all stop modes
Bus clock	Up to 75 MHz	Up to 4 MHz	MCGOUTCLK clock divider	In all stop modes
FlexBus clock (FB_CLK)	Up to 50 MHz	Up to 4 MHz	MCGOUTCLK clock divider	In all stop modes or FlexBus disabled
Flash clock	Up to 25 MHz	Up to 1 MHz	MCGOUTCLK clock divider	In all stop modes

Table continues on the next page...

Table 5-1. Clock Summary (continued)

Clock name	Run mode clock frequency	VLPR mode clock frequency	Clock source	Clock is disabled when…
Internal reference (MCGIRCLK)	30-40 kHz or 4 MHz	4 MHz only	MCG	MCG_C1[IRCLKEN] cleared, Stop mode and MCG_C1[IREFSTEN] cleared, or VLPS/LLS/VLLS mode
External reference (OSC0ERCLK, OSC1ERCLK)	Up to 50 MHz (bypass), 30-40 kHz, or 4-32 MHz (crystal)	Up to 8 MHz (bypass), 30-40 kHz (low-range crystal) or Up to 4 MHz (high-range crystal)	System OSC	System OSC's OSC_CR[ERCLKEN] cleared, or Stop mode and OSC_CR[EREFSTEN] cleared
External reference 32kHz (ERCLK32K)	30-40 kHz	30-40 kHz	System OSC0 or RTC OSC depending on SIM_SOPT1[OSC32KSEL]	System OSC's OSC_CR[ERCLKEN] cleared or RTC's RTC_CR[OSCE] cleared
RTC_CLKOUT	1 Hz or 32 kHz	1 Hz or 32 kHz	RTC clock	Clock is disabled in LLS and VLLSx modes
LPO	1 kHz	1 kHz	PMC	Available in all power modes
USB HS clock	60 MHz	N/A	Bus clock, MCGPLL0CLK, MCGPLL1CLK, or OSC0ERCLK with fractional clock divider, or ULPI_CLK	USB HS OTG is disabled
USB FS clock	48 MHz	N/A	MCGPLL0CLK, MCGPLL1CLK, or MCGFLLCLK with fractional clock divider, or USB_CLKIN	USB FS OTG is disabled
I2S master clock	Up to 25 MHz	Up to 12.5 MHz	System clock, MCGPLLCLK, OSC0ERCLK, or OSC1ERCLK with fractional clock divider, or I2S_CLKIN	I^{2}S is disabled

Table continues on the next page...

Table 5-1. Clock Summary (continued)

Clock name	Run mode clock frequency	VLPR mode clock frequency	Clock source	Clock is disabled when…
SDHC clock	Up to 50 MHz	N/A	System clock, MCGPLLCLK/ MCGFLLCLK, or OSC0ERCLK	SDHC is disabled
TRACE clock	Up to 100 MHz	Up to 4 MHz	System clock or MCGOUTCLK	Trace is disabled

5.5 Internal clocking requirements

Each system clock divider is programmed via the SIM module’s CLKDIV registers. Each divider is programmable from a divide-by-1 through divide-by-16 setting. The following requirements must be met when configuring the system clocks for this device:

1. The core and system clock frequencies frequency must be 120 MHz or slower.
2. The bus clock frequency must be 75 MHz or slower and an integer divide of the core clock.
3. The flash clock frequency must be 25 MHz or slower and an integer divide of the bus clock.

The following are examples of valid clock configurations for this device:

Clock	Frequency
Core clock	120 MHz
Bus clock	60 MHz
Flash clock	20 MHz

5.5.1 Clock divider values after reset

Each clock divider is programmed via the SIM module’s CLKDIV*n* registers. The flash memory's FTFE_FOPT[LPBOOT] bit controls the reset value of the core clock, system clock, bus clock, and flash clock dividers as shown below:

FTFE_FOPT [LPBOOT]	Core/system clock	Bus clock	FlexBus clock	Flash clock	Description
0	0x7 (divide by 8)	0x7 (divide by 8)	0xF (divide by 16)	0xF (divide by 16)	Low power boot
1	0x0 (divide by 1)	0x0 (divide by 1)	0x1 (divide by 2)	0x1 (divide by 2)	Fast clock boot

This gives the user flexibility for a lower frequency, low-power boot option. The flash erased state defaults to fast clocking mode, since where the low power boot (FTFE_FOPT[LPBOOT]) bit resides in flash is logic 1 in the flash erased state.

To enable the low power boot option program FTFE_FOPT[LPBOOT] to zero. During the reset sequence, if LPBOOT is cleared, the system is in a slow clock configuration. Upon any system reset, the clock dividers return to this configurable reset state.

5.5.2 VLPR mode clocking

The clock dividers cannot be changed while in VLPR mode. They must be programmed prior to entering VLPR mode to guarantee:

- the core/system, FlexBus, and bus clocks are less than or equal to 4 MHz, and
- the flash memory clock is less than or equal to 1 MHz

5.6 Clock Gating

The clock to each module can be individually gated on and off using the SIM module's SCGC*x* registers. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing a module, set the corresponding bit in SCGC*x* register to enable the clock. Before turning off the clock, make sure to disable the module.

Any bus access to a peripheral that has its clock disabled generates an error termination.

5.7 Module clocks

The following table summarizes the clocks associated with each module.

Table 5-2. Module clocks

Module	Bus interface clock	Internal clocks	I/O interface clocks
Core modules			
ARM Cortex-M4 core	System clock	Core clock	—
NVIC	System clock	—	—
DAP	System clock	—	—
ITM	System clock	—	—

Table continues on the next page...

Table 5-2. Module clocks (continued)

Module	Bus interface clock	Internal clocks	I/O interface clocks
ETM	System clock	TRACE clock	TRACE_CLKOUT
ETB	System clock	—	—
cJTAG, JTAGC	—	—	JTAG_CLK
System modules			
DMA	System clock	—	—
DMA Mux	Bus clock	—	—
Port control	Bus clock	LPO	—
Crossbar Switch	System clock	—	—
Peripheral bridges	System clock	Bus clock	—
MPU	System clock	—	—
LLWU, PMC, SIM, RCM	Bus clock	LPO	—
Mode controller	Bus clock	—	—
MCM	System clock	—	—
EWM	Bus clock	LPO	—
Watchdog timer	Bus clock	LPO	—
Clocks			
MCG	Bus clock	MCGOUTCLK, MCGPLLCLK, MCGFLLCLK, MCGIRCLK, OSC0ERCLK, OSC1ERCLK, EXTAL32K	—
OSC	Bus clock	OSC0ERCLK, OSC1ERCLK	—
Memory and memory interfaces			
Flash Controller	System clock	Flash clock	—
Flash memory	Flash clock	—	—
FlexBus	System clock	—	FB_CLKOUT
EzPort	System clock	—	EZP_CLK
Security			
CRC	Bus clock	—	—
Analog			
ADC	Bus clock	OSC0ERCLK	—
CMP	Bus clock	—	—
DAC	Bus clock	—	—
VREF	Bus clock	—	—
Timers			
PDB	Bus clock	—	—
FlexTimers	Bus clock	MCGFFCLK	FTM_CLKIN*x*

Table continues on the next page...

Table 5-2. Module clocks (continued)

Module	Bus interface clock	Internal clocks	I/O interface clocks
PIT	Bus clock	—	—
LPTMR	Bus clock	LPO, OSC0ERCLK, MCGIRCLK, ERCLK32K	—
CMT	Bus clock	—	—
RTC	Bus clock	EXTAL32	—
Communication interfaces			
USB HS OTG	System clock	USB HS clock	ULPI_CLK
USB FS OTG	System clock	USB FS clock	—
USB DCD	Bus clock	—	—
FlexCAN	Bus clock	OSC0ERCLK	—
DSPI	Bus clock	—	DSPI_SCK
I^{2}C	Bus clock	—	I2C_SCL
UART0, UART1	System clock	—	—
UART2-5	Bus clock	—	—
SDHC	System clock	SDHC clock	SDHC_DCLK
I^{2}S	Bus clock	I^{2}S master clock	I2S_TX_BCLK, I2S_RX_BCLK
Human-machine interfaces			
GPIO	System clock	—	—
TSI	Bus clock	LPO, ERCLK32K, MCGIRCLK	—

5.7.1 PMC 1-kHz LPO clock

The Power Management Controller (PMC) generates a 1-kHz clock that is enabled in all modes of operation, including all low power modes. This 1-kHz source is commonly referred to as LPO clock or 1-kHz LPO clock.

5.7.2 WDOG clocking

The WDOG may be clocked from two clock sources as shown in the following figure.

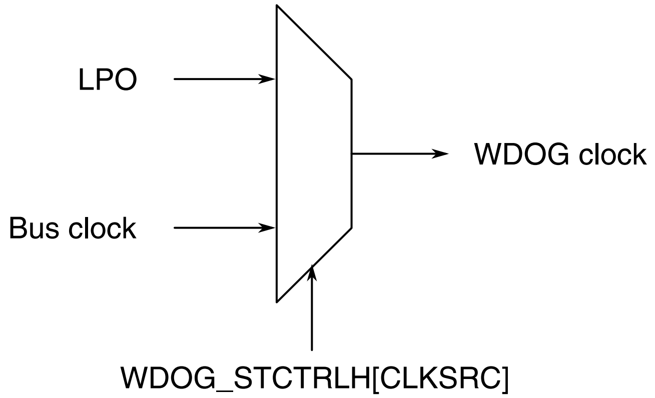


Figure 5-2. WDOG clock generation

5.7.3 Debug trace clock

The debug trace clock source can be clocked as shown in the following figure.

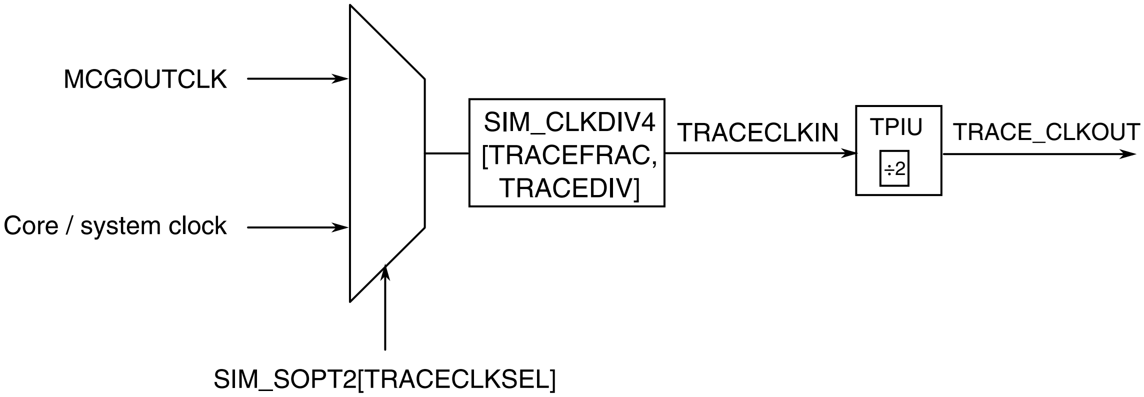


Figure 5-3. Trace clock generation

NOTE

The trace clock frequency observed at the TRACE_CLKOUT pin will be half that of the selected clock source.

5.7.4 PORT digital filter clocking

The digital filters in each of the PORT*x* modules can be clocked as shown in the following figure.

NOTE

In stop mode, the digital input filters are bypassed unless they are configured to run from the 1 kHz LPO clock source.

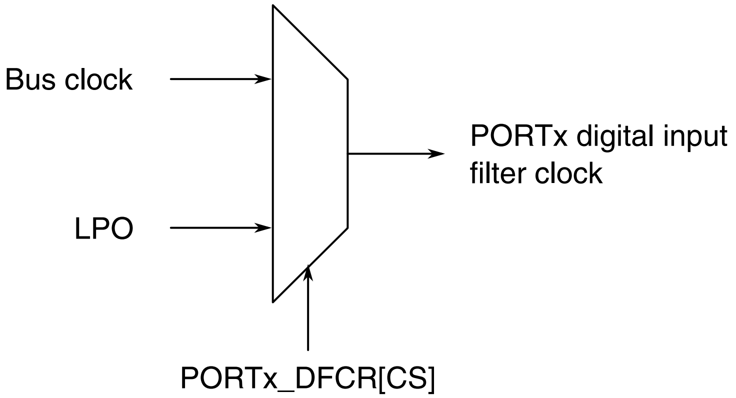


Figure 5-4. PORTx digital input filter clock generation

5.7.5 NAND Flash Controller (NFC) clocking

The NAND Flash Controller's clock source is determined by the SIM_SOPT2[NFCSRC] bitfield and the SIM_CLKDIV4[NFCFRAC, NFCDIV] bitfields as shown below.

> **NOTE**
>
> The reciprocal of ((NFCFRAC+1)/(NFCDIV+1)) must be a multiple of 0.5. For example, NFCFRAC = 1 and NFCDIV = 2 is a valid setting, since the reciprocal is 1.5. However, NFCFRAC = 2 and NFCDIV=7 is not a valid setting, since the reciprocal is 2.6667.

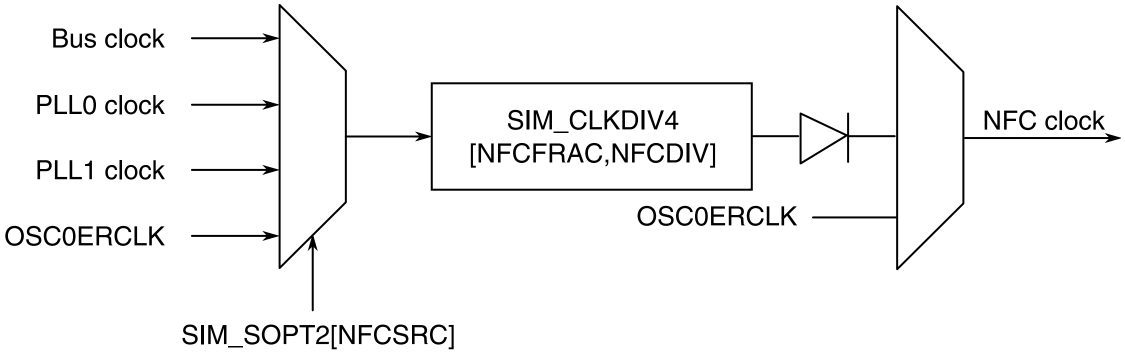


Figure 5-5. NFC clock generation

5.7.6 LPTMR clocking

The prescaler and glitch filters in each of the LPTMR*x* modules can be clocked as shown in the following figure.

NOTE

The chosen clock must remain enabled if the LPTMR*x* is to continue operating in all required low-power modes.

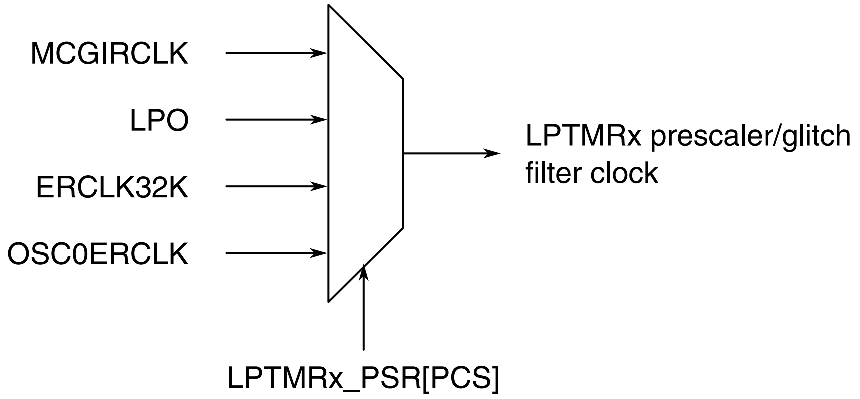


Figure 5-6. LPTMRx prescaler/glitch filter clock generation

5.7.7 USB FS OTG Controller clocking

The USB FS OTG controller clocking options are shown in the following figure.

NOTE

For the USB FS OTG controller to operate, the minimum system clock frequency is 20 MHz.

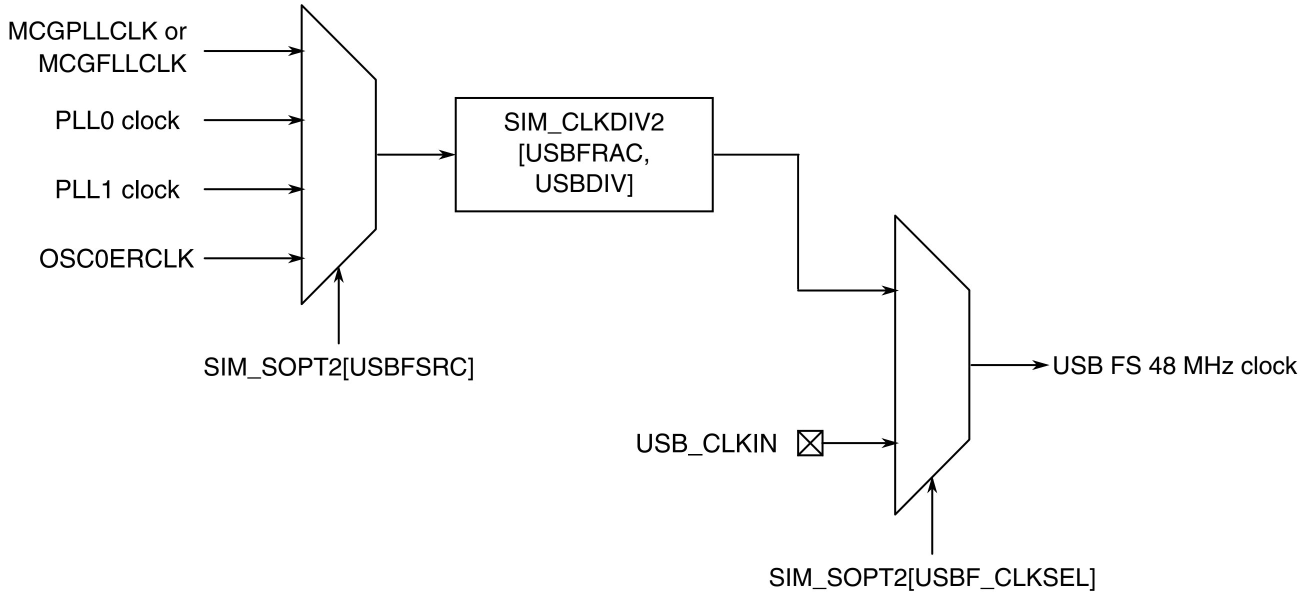


Figure 5-7. USB FS clock generation

NOTE

The MCGFLLCLK does not meet the USB jitter specifications for certification.

5.7.8 USB HS OTG Controller clocking

The USB HS OTG controller clocking options are shown in the following figure.

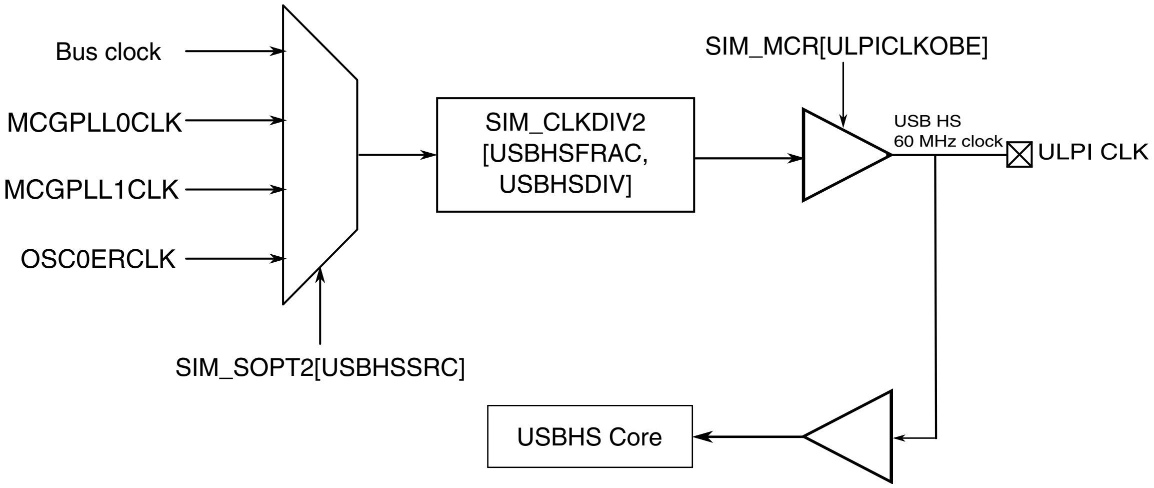


Figure 5-8. USB HS clock generation

5.7.9 FlexCAN clocking

The clock for the FlexCAN's protocol engine can be selected as shown in the following figure.

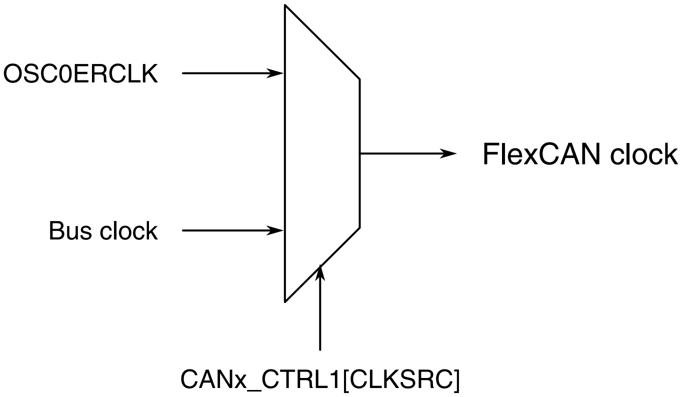


Figure 5-9. FlexCAN clock generation

5.7.10 UART clocking

UART0 and UART1 modules operate from the core/system clock, which provides higher performance level for these modules. All other UART modules operate from the bus clock.

5.7.11 SDHC clocking

The SDHC module has four possible clock sources for the external clock source, as shown in the following figure.

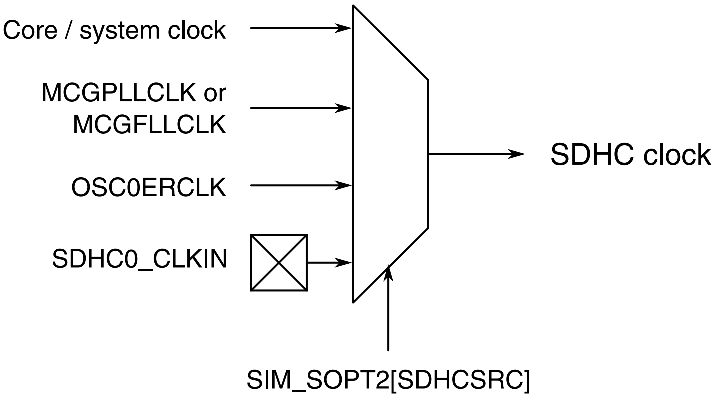


Figure 5-10. SDHC clock generation

5.7.12 I²S/SAI clocking

The audio master clock (MCLK) is used to generate the bit clock when the receiver or transmitter is configured for an internally generated bit clock. The audio master clock can also be output to or input from a pin. The transmitter and receiver have the same audio master clock inputs.

Each SAI peripheral can control the input clock selection, pin direction and divide ratio of one audio master clock.

The I²S/SAI transmitter and receiver support asynchronous bit clocks (BCLKs) that can be generated internally from the audio master clock or supplied externally. The module also supports the option for synchronous operation between the receiver and transmitter or between two separate I²S/SAI peripherals.

The transmitter and receiver can independently select between the bus clock and the audio master clocks to generate the bit clock.

The MCLK and BCLK source options appear in the following figure.

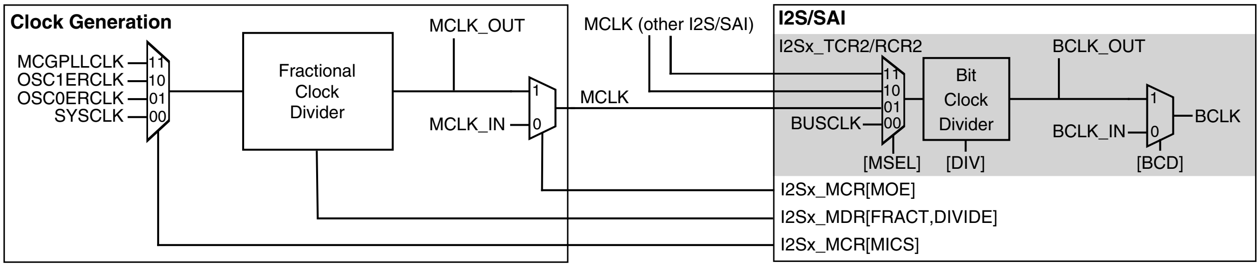


Figure 5-11. I^{2}S/SAI clock generation

5.7.13 TSI clocking

In active mode, the TSI can be clocked as shown in the following figure.

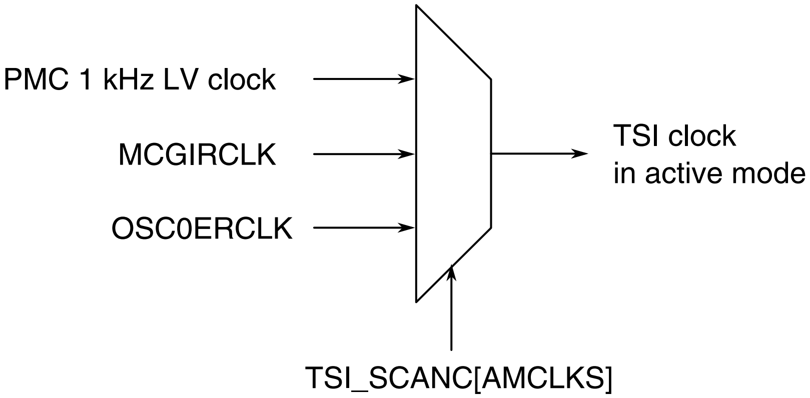


Figure 5-12. TSI clock generation

In low-power mode, the TSI can be clocked as shown in the following figure.

NOTE

In the TSI chapter, these two clocks are referred to as LPOCLK and VLPOSCCLK.

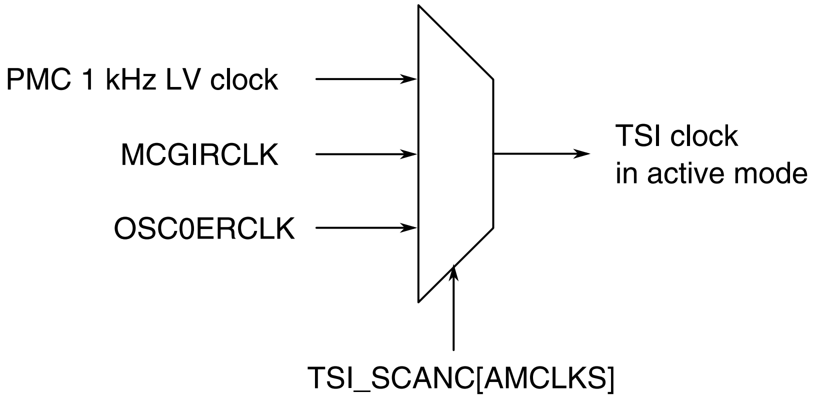


Figure 5-13. TSI low-power clock generation

Chapter 6
Reset and Boot

6.1 Introduction

The following reset sources are supported in this MCU:

Table 6-1. Reset sources

Reset sources	Description
POR reset	• Power-on reset (POR)
System resets	• External pin reset (PIN) • Low-voltage detect (LVD) • Computer operating properly (COP) watchdog reset • Low leakage wakeup (LLWU) reset • Multipurpose clock generator loss of clock (LOC) reset • Stop mode acknowledge error (SACKERR) • Software reset (SW) • Lockup reset (LOCKUP) • EzPort reset • MDM DAP system reset
Debug reset	• JTAG reset • nTRST reset

Each of the system reset sources, with the exception of the EzPort and MDM-AP reset, has an associated bit in the system reset status (SRS) registers. See the Reset Control Module for register details.

The MCU exits reset in functional mode that is controlled by $\overline{\text{EZP\_CS}}$ pin to select between the single chip (default) or serial flash programming (EzPort) modes. See Boot options for more details.

6.2 Reset

This section discusses basic reset mechanisms and sources. Some modules that cause resets can be configured to cause interrupts instead. Consult the individual peripheral chapters for more information.

6.2.1 Power-on reset (POR)

When power is initially applied to the MCU or when the supply voltage drops below the power-on reset re-arm voltage level (V_{POR}), the POR circuit causes a POR reset condition.

As the supply voltage rises, the LVD circuit holds the MCU in reset until the supply has risen above the LVD low threshold (V_{LVDL}). The POR and LVD bits in SRS0 register are set following a POR.

6.2.2 System reset sources

Resetting the MCU provides a way to start processing from a known set of initial conditions. System reset begins with the on-chip regulator in full regulation and system clocking generation from an internal reference. When the processor exits reset, it performs the following:

- Reads the start SP (SP_main) from vector-table offset 0
- Reads the start PC from vector-table offset 4
- LR is set to 0xFFFF_FFFF

The on-chip peripheral modules are disabled and the non-analog I/O pins are initially configured as disabled. The pins with analog functions assigned to them default to their analog function after reset.

During and following a reset, the JTAG pins have their associated input pins configured as:

- TDI in pull-up (PU)
- TCK in pull-down (PD)
- TMS in PU

and associated output pin configured as:

- TDO with no pull-down or pull-up

Note that the nTRST signal is initially configured as disabled, however once configured to its JTAG functionality its associated input pin is configured as:

- nTRST in PU

6.2.2.1 External pin reset (PIN)

On this device, $\overline{\text{RESET}}$ is a dedicated pin. This pin is open drain and has an internal pullup device. Asserting $\overline{\text{RESET}}$ wakes the device from any mode. During a pin reset, the RCM's SRS0[PIN] bit is set.

6.2.2.1.1 Reset pin filter

The $\overline{\text{RESET}}$ pin filter supports filtering from both the 1 kHz LPO clock and the bus clock. A separate filter is implemented for each clock source. In stop and VLPS mode operation, this logic either switches to bypass operation or has continued filtering operation depending on the filtering mode selected. In low leakage stop modes, a separate LPO filter in the LLWU can continue filtering the $\overline{\text{RESET}}$ pin.

The RPFC[RSTFLTSS], RPFC[RSTFLTSRW], and RPFW[RSTFLTSEL] fields in the reset control (RCM) register set control this functionality; see the RCM chapter. The filters are asynchronously reset by Chip POR. The reset value for each filter assumes the RESET pin is negated.

The two clock options for the $\overline{\text{RESET}}$ pin filter when the chip is not in low leakage modes are the LPO (1 kHz) and bus clock. For low leakage modes VLLS3, VLLS2, VLLS1, the LLWU provides control (in the LLWU_RST register) of an optional fixed digital filter running the LPO.

The LPO filter has a fixed filter value of 3. Due to a synchronizer on the input data, there is also some associated latency (2 cycles). As a result, 5 cycles are required to complete a transition from low to high or high to low.

The bus filter initializes to off (logic 1) when the bus filter is not enabled. The bus clock is used when the filter selects bus clock, and the number of counts is controlled by the RCM's RPFW[RSTFLTSEL] field.

6.2.2.2 Low-voltage detect (LVD)

The chip includes a system for managing low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system consists of a power-on reset (POR) circuit and an LVD circuit with a user-selectable trip voltage. The LVD system is always enabled in normal run, wait, or stop mode. The LVD system is disabled when entering VLPx, LLS, or VLLSx modes.

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting the PMC's LVDSC1[LVDRE] bit to 1. The low voltage detection threshold is determined by the PMC's LVDSC1[LVDV] field. After an LVD reset has occurred, the LVD system holds the MCU in reset until the supply voltage has risen above the low voltage detection threshold. The RCM's SRS0[LVD] bit is set following either an LVD reset or POR.

6.2.2.3 Computer operating properly (COP) watchdog timer

The computer operating properly (COP) watchdog timer (WDOG) monitors the operation of the system by expecting periodic communication from the software. This communication is generally known as servicing (or refreshing) the COP watchdog. If this periodic refreshing does not occur, the watchdog issues a system reset. The COP reset causes the RCM's SRS0[WDOG] bit to set.

6.2.2.4 Low leakage wakeup (LLWU)

The LLWU module provides the means for a number of external pins, the $\overline{\text{RESET}}$ pin, and a number of internal peripherals to wake the MCU from low leakage power modes. The LLWU module is functional only in low leakage power modes.

- In LLS mode, only the $\overline{\text{RESET}}$ pin via the LLWU can generate a system reset.
- In VLLSx modes, all enabled inputs to the LLWU can generate a system reset.

After a system reset, the LLWU retains the flags indicating the input source of the last wakeup until the user clears them.

NOTE

Some flags are cleared in the LLWU and some flags are required to be cleared in the peripheral module. Refer to the individual peripheral chapters for more information.

6.2.2.5 Multipurpose clock generator loss-of-clock (LOC)

The MCG module supports an external reference clock.

If the C6[CME] bit in the MCG module is set, the clock monitor is enabled. If the external reference falls below f_{loc_low} or f_{loc_high}, as controlled by the C2[RANGE] field in the MCG module, the MCU resets. The RCM's SRS0[LOC] bit is set to indicate this reset source.

NOTE

This reset source does not cause a reset if the chip is in any stop mode.

6.2.2.6 Stop mode acknowledge error (SACKERR)

This reset is generated if the core attempts to enter stop mode, but not all modules acknowledge stop mode within 1025 cycles of the 1 kHz LPO clock.

A module might not acknowledge the entry to stop mode if an error condition occurs. The error can be caused by a failure of an external clock input to a module.

6.2.2.7 Software reset (SW)

The SYSRESETREQ bit in the NVIC application interrupt and reset control register can be set to force a software reset on the device. (See ARM's NVIC documentation for the full description of the register fields, especially the VECTKEY field requirements.) Setting SYSRESETREQ generates a software reset request. This reset forces a system reset of all major components except for the debug module. A software reset causes the RCM's SRS1[SW] bit to set.

6.2.2.8 Lockup reset (LOCKUP)

The LOCKUP gives immediate indication of seriously errant kernel software. This is the result of the core being locked because of an unrecoverable exception following the activation of the processor’s built in system state protection hardware.

The LOCKUP condition causes a system reset and also causes the RCM's SRS1[LOCKUP] bit to set.

6.2.2.9 EzPort reset

The EzPort supports a system reset request via EzPort signaling. The EzPort generates a system reset request following execution of a Reset Chip (RESET) command via the EzPort interface. This method of reset allows the chip to boot from flash memory after it has been programmed by an external source. The EzPort is enabled or disabled by the $\overline{\text{EZP\_CS}}$ pin.

An EzPort reset causes the RCM's SRS1[EZPT] bit to set.

6.2.2.10 MDM-AP system reset request

Set the system reset request bit in the MDM-AP control register to initiate a system reset. This is the primary method for resets via the JTAG/SWD interface. The system reset is held until this bit is cleared.

Set the core hold reset bit in the MDM-AP control register to hold the core in reset as the rest of the chip comes out of system reset.

6.2.3 MCU Resets

A variety of resets are generated by the MCU to reset different modules.

6.2.3.1 VBAT POR

The VBAT POR asserts on a VBAT POR reset source. It affects only the modules within the VBAT power domain: RTC and VBAT Register File. These modules are not affected by the other reset types.

6.2.3.2 POR Only

The POR Only reset asserts on the POR reset source only. It resets the PMC and System Register File.

The POR Only reset also causes all other reset types (except VBAT POR) to occur.

6.2.3.3 Chip POR not VLLS

The Chip POR not VLLS reset asserts on POR and LVD reset sources. It resets parts of the SMC and SIM. It also resets the LPTMR.

The Chip POR not VLLS reset also causes these resets to occur: Chip POR, Chip Reset not VLLS, and Chip Reset (including Early Chip Reset).

6.2.3.4 Chip POR

The Chip POR asserts on POR, LVD, and VLLS Wakeup reset sources. It resets the Reset Pin Filter registers and parts of the SIM and MCG.

The Chip POR also causes the Chip Reset (including Early Chip Reset) to occur.

6.2.3.5 Chip Reset not VLLS

The Chip Reset not VLLS reset asserts on all reset sources except a VLLS Wakeup that does not occur via the $\overline{\text{RESET}}$ pin. It resets parts of the SMC, LLWU, and other modules that remain powered during VLLS mode.

The Chip Reset not VLLS reset also causes the Chip Reset (including Early Chip Reset) to occur.

6.2.3.6 Early Chip Reset

The Early Chip Reset asserts on all reset sources. It resets only the flash memory module. It negates before flash memory initialization begins ("earlier" than when the Chip Reset negates).

6.2.3.7 Chip Reset

Chip Reset asserts on all reset sources and only negates after flash initialization has completed and the $\overline{\text{RESET}}$ pin has also negated. It resets the remaining modules (the modules not reset by other reset types).

6.2.4 Reset Pin

For all reset sources except a VLLS Wakeup that does not occur via the $\overline{\text{RESET}}$ pin, the $\overline{\text{RESET}}$ pin is driven low by the MCU for at least 128 bus clock cycles and until flash initialization has completed.

After flash initialization has completed, the $\overline{\text{RESET}}$ pin is released, and the internal Chip Reset negates after the $\overline{\text{RESET}}$ pin is pulled high. Keeping the $\overline{\text{RESET}}$ pin asserted externally delays the negation of the internal Chip Reset.

6.2.5 Debug resets

The following sections detail the debug resets available on the device.

6.2.5.1 JTAG reset

The JTAG module generate a system reset when certain IR codes are selected. This functional reset is asserted when EzPort, EXTEST, HIGHZ and CLAMP instructions are active. The reset source from the JTAG module is released when any other IR code is selected. A JTAG reset causes the RCM's SRS1[JTAG] bit to set.

6.2.5.2 nTRST reset

The nTRST pin causes a reset of the JTAG logic when asserted. Asserting the nTRST pin allows the debugger to gain control of the TAP controller state machine (after exiting LLS or VLLSx) without resetting the state of the debug modules.

The nTRST pin does not cause a system reset.

6.2.5.3 Resetting the Debug subsystem

Use the CDBGRSTREQ bit within the SWJ-DP CTRL/STAT register to reset the debug modules. However, as explained below, using the CDBGRSTREQ bit does not reset all debug-related registers.

CDBGRSTREQ resets the debug-related registers within the following modules:

- SWJ-DP
- AHB-AP
- ETM

- ATB replicators
- ATB upsizers
- ATB funnels
- ETB
- TPIU
- MDM-AP (MDM control and status registers)
- MCM (ETB "Almost Full" logic)

CDBGRSTREQ does not reset the debug-related registers within the following modules:

- CM4 core (core debug registers: DHCSR, DCRSR, DCRDR, DEMCR)
- FPB
- DWT
- ITM
- NVIC
- Crossbar bus switch[1]
- AHB-AP[1]
- Private peripheral bus[1]

6.3 Boot

This section describes the boot sequence, including sources and options.

6.3.1 Boot sources

This device only supports booting from internal flash. Any secondary boot must go through an initialization sequence in flash.

6.3.2 Boot options

The device's functional mode is controlled by the state of the EzPort chip select ($\overline{\text{EZP\_CS}}$) pin during reset.

[1] CDBGRSTREQ does not affect AHB resources so that debug resources on the private peripheral bus are available during System Reset.

The device can be in single chip (default) or serial flash programming mode (EzPort). While in single chip mode the device can be in run or various low power modes mentioned in Power mode transitions.

Table 6-2. Mode select decoding

EzPort chip select ($\overline{\text{EZP\_CS}}$)	Description
0	Serial flash programming mode (EzPort)
1	Single chip (default)

6.3.3 FOPT boot options

The flash option register (FOPT) in flash memory module (FTFE) allows the user to customize the operation of the MCU at boot time. The register contains read-only bits that are loaded from the NVM's option byte in the flash configuration field. The user can reprogram the option byte in flash to change the FOPT values that are used for subsequent resets. For more details on programming the option byte, refer to the flash memory chapter.

The MCU uses the FTFE_FOPT register bits to configure the device at reset as shown in the following table.

Table 6-3. Flash Option Register (FTFE_FOPT) Bit Definitions

Bit Num	Field	Value	Definition
7-2	Reserved		Reserved for future expansion.
1	EZPORT_DIS	0	EzPort operation is disabled. The device always boots to normal CPU execution and the state of $\overline{\text{EZP\_CS}}$ signal during reset is ignored. This option avoids inadvertent resets into EzPort mode if the $\overline{\text{EZP\_CS}}$/NMI pin is used for its NMI function.
		1	EzPort operation is enabled. The state of $\overline{\text{EZP\_CS}}$ pin during reset determines if device enters EzPort mode.
0	LPBOOT	0	Low-power boot: OUTDIVx values in SIM_CLKDIV1 register are auto-configured at reset exit for higher divide values that produce lower power consumption at reset exit. • Core and system clock divider (OUTDIV1) and bus clock divider (OUTDIV2) are 0x7 (divide by 8) • Flash clock divider (OUTDIV4) and FlexBus clock divider (OUTDIV3) are 0xF (divide by 16)
		1	Normal boot: OUTDIVx values in SIM_CLKDIV1 register are auto-configured at reset exit for higher frequency values that produce faster operating frequencies at reset exit. • Core and system clock divider (OUTDIV1) and bus clock divider (OUTDIV2) are 0x0 (divide by 1) • Flash clock divider (OUTDIV4) and FlexBus clock divider (OUTDIV3) are 0x1 (divide by 2)

6.3.4 Boot sequence

At power up, the on-chip regulator holds the system in a POR state until the input supply is above the POR threshold. The system continues to be held in this static state until the internally regulated supplies have reached a safe operating voltage as determined by the LVD. The Mode Controller reset logic then controls a sequence to exit reset.

1. A system reset is held on internal logic, the $\overline{\text{RESET}}$ pin is driven out low, and the MCG is enabled in its default clocking mode.
2. Required clocks are enabled (Core Clock, System Clock, Flash Clock, and any Bus Clocks that do not have clock gate control).
3. The system reset on internal logic continues to be held, but the Flash Controller is released from reset and begins initialization operation while the Mode Control logic continues to drive the $\overline{\text{RESET}}$ pin out low for a count of ~128 Bus Clock cycles.
4. The $\overline{\text{RESET}}$ pin is released, but the system reset of internal logic continues to be held until the Flash Controller finishes initialization. EzPort mode is selected instead of the normal CPU execution if $\overline{\text{EZP\_CS}}$ is low when the internal reset is deasserted. EzPort mode can be disabled by programming the FOPT[EZPORT_DIS] field in the Flash Memory module.
5. When Flash Initialization completes, the $\overline{\text{RESET}}$ pin is observed. If $\overline{\text{RESET}}$ continues to be asserted (an indication of a slow rise time on the $\overline{\text{RESET}}$ pin or external drive in low), the system continues to be held in reset. Once the $\overline{\text{RESET}}$ pin is detected high, the system is released from reset.
6. At release of system reset, clocking is switched to a slow clock if the FOPT[LPBOOT] field in the Flash Memory module is configured for Low Power Boot
7. When the system exits reset, the processor sets up the stack, program counter (PC), and link register (LR). The processor reads the start SP (SP_main) from vector-table offset 0. The core reads the start PC from vector-table offset 4. LR is set to 0xFFFF_FFFF. The CPU begins execution at the PC location. EzPort mode is entered instead of the normal CPU execution if Ezport mode was latched during the sequence.
8. If FlexNVM is enabled, the flash controller continues to restore the FlexNVM data. This data is not available immediately out of reset and the system should not access this data until the flash controller completes this initialization step as indicated by the EEERDY flag.

Subsequent system resets follow this reset flow beginning with the step where system clocks are enabled.

Chapter 7
Power Management

7.1 Introduction

This chapter describes the various chip power modes and functionality of the individual modules in these modes.

7.2 Power modes

The power management controller (PMC) provides multiple power options to allow the user to optimize power consumption for the level of functionality needed.

Depending on the stop requirements of the user application, a variety of stop modes are available that provide state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. The following table compares the various power modes available.

For each run mode there is a corresponding wait and stop mode. Wait modes are similar to ARM sleep modes. Stop modes (VLPS, STOP) are similar to ARM sleep deep mode. The very low power run (VLPR) operating mode can drastically reduce runtime power when the maximum bus frequency is not required to handle the application needs.

The three primary modes of operation are run, wait and stop. The WFI instruction invokes both wait and stop modes for the chip. The primary modes are augmented in a number of ways to provide lower power based on application needs.

Table 7-1. Chip power modes

Chip mode	Description	Core mode	Normal recovery method
Normal run	Allows maximum performance of chip. Default mode out of reset; on-chip voltage regulator is on.	Run	-

Table continues on the next page...

Table 7-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
Normal Wait - via WFI	Allows peripherals to function while the core is in sleep mode, reducing power. NVIC remains sensitive to interrupts; peripherals continue to be clocked.	Sleep	Interrupt
Normal Stop - via WFI	Places chip in static state. Lowest power mode that retains all registers while maintaining LVD protection. NVIC is disabled; AWIC is used to wake up from interrupt; peripheral clocks are stopped.	Sleep Deep	Interrupt
VLPR (Very Low Power Run)	On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. Reduced frequency Flash access mode (1 MHz); LVD off; internal oscillator provides a low power 4 MHz source for the core, the bus and the peripheral clocks.	Run	Interrupt
VLPW (Very Low Power Wait) -via WFI	Same as VLPR but with the core in sleep mode to further reduce power; NVIC remains sensitive to interrupts (FCLK = ON). On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency.	Sleep	Interrupt
VLPS (Very Low Power Stop)-via WFI	Places chip in static state with LVD operation off. Lowest power mode with ADC and pin interrupts functional. Peripheral clocks are stopped, but LPTimer, RTC, CMP, TSI, DAC can be used. NVIC is disabled (FCLK = OFF); AWIC is used to wake up from interrupt. On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. All SRAM is operating (content retained and I/O states held).	Sleep Deep	Interrupt
LLS (Low Leakage Stop)	State retention power mode. Most peripherals are in state retention mode (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI, DAC can be used. NVIC is disabled; LLWU is used to wake up. **NOTE:** The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery. All SRAM is operating (content retained and I/O states held).	Sleep Deep	Wakeup Interrupt[1]
VLLS3 (Very Low Leakage Stop3)	Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI, DAC can be used. NVIC is disabled; LLWU is used to wake up. SRAM_U and SRAM_L remain powered on (content retained and I/O states held).	Sleep Deep	Wakeup Reset[2]
VLLS2 (Very Low Leakage Stop2)	Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI, DAC can be used. NVIC is disabled; LLWU is used to wake up. SRAM_L is powered off. A portion of SRAM_U remains powered on (content retained and I/O states held).	Sleep Deep	Wakeup Reset[2]
VLLS1 (Very Low Leakage Stop1)	Most peripherals are disabled (with clocks stopped), but LLWU, LPTimer, RTC, CMP, TSI, DAC can be used. NVIC is disabled; LLWU is used to wake up. All of SRAM_U and SRAM_L are powered off. The 32-byte system register file and the 32-byte VBAT register file remain powered for customer-critical data.	Sleep Deep	Wakeup Reset[2]

Table continues on the next page...

Table 7-1. Chip power modes (continued)

Chip mode	Description	Core mode	Normal recovery method
BAT (backup battery only)	The chip is powered down except for the VBAT supply. The RTC and the 32-byte VBAT register file for customer-critical data remain powered.	Off	Power-up Sequence

1. Resumes normal run mode operation by executing the LLWU interrupt service routine.
2. Follows the reset flow with the LLWU interrupt flag set for the NVIC.

7.3 Entering and exiting power modes

The WFI instruction invokes wait and stop modes for the chip. The processor exits the low-power mode via an interrupt. The Nested Vectored Interrupt Controller (NVIC) describes interrupt operation and what peripherals can cause interrupts.

NOTE

The WFE instruction can have the side effect of entering a low-power mode, but that is not its intended usage. See ARM documentation for more on the WFE instruction.

Recovery from VLLSx is through the wake-up Reset event. The chip wake-ups from VLLSx by means of reset, an enabled pin or enabled module. See the table "LLWU inputs" in the LLWU configuration section for a list of the sources.

The wake-up flow from VLLSx is through reset. The wakeup bit in the SRS registers in the RCM is set indicating that the chip is recovering from a low power mode. Code execution begins; however, the I/O pins are held in their pre low power mode entry states, and the system oscillator and MCG registers are reset (even if EREFSTEN had been set before entering VLLSx). Software must clear this hold by writing a 1 to the ACKISO bit in the Regulator Status and Control Register in the PMC module.

NOTE

To avoid unwanted transitions on the pins, software must re-initialize the I/O pins to their pre-low-power mode entry states *before* releasing the hold.

If the oscillator was configured to continue running during VLLSx modes, it must be re-configured before the ACKISO bit is cleared. The oscillator configuration within the MCG is cleared after VLLSx recovery and the oscillator will stop when ACKISO is cleared unless the register is re-configured.

7.4 Power mode transitions

The following figure shows the power mode transitions. Any reset always brings the chip back to the normal run state. In run, wait, and stop modes active power regulation is enabled. The VLPx modes are limited in frequency, but offer a lower power operating mode than normal modes. The LLS and VLLSx modes are the lowest power stop modes based on amount of logic or memory that is required to be retained by the application.

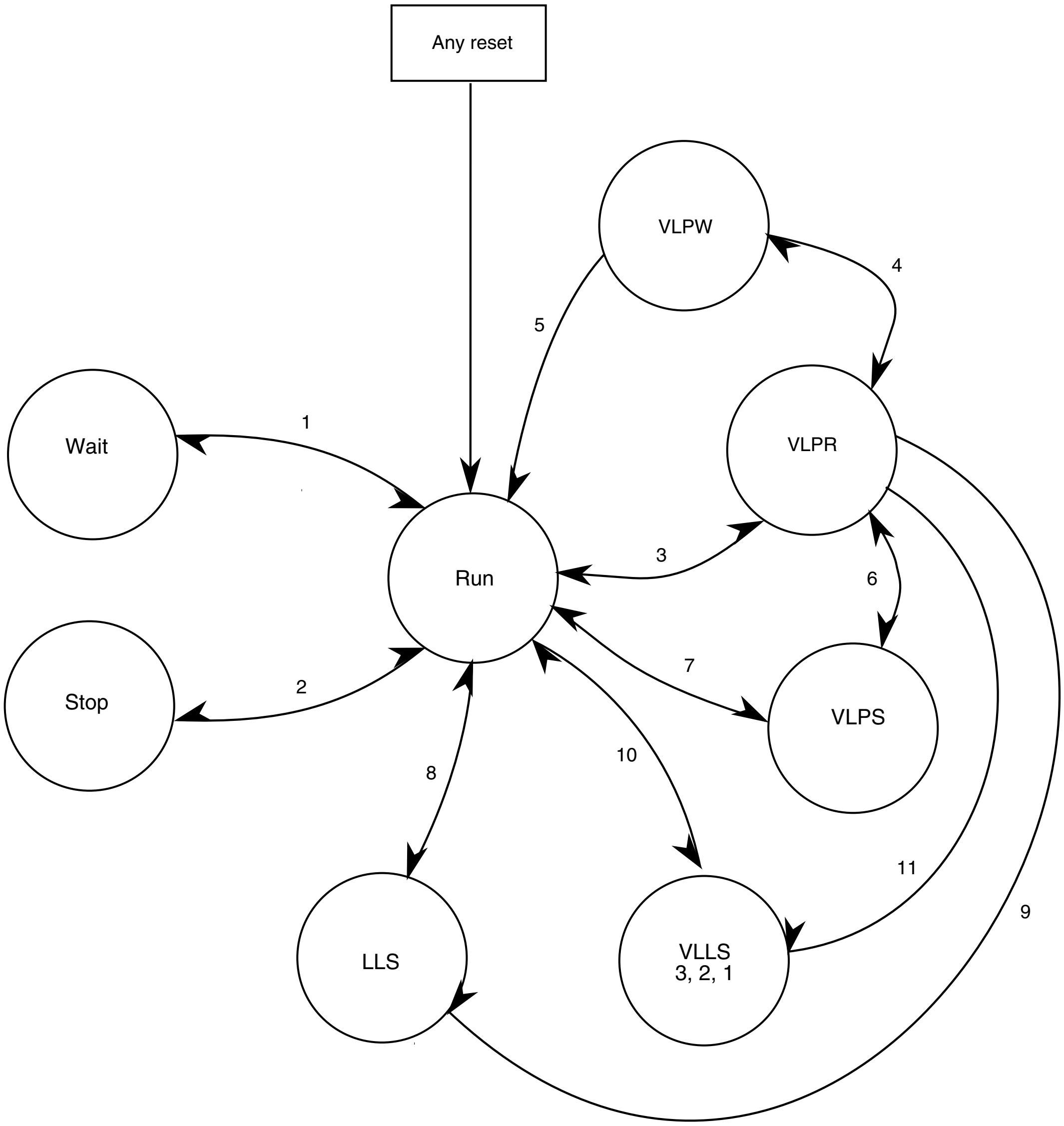


Figure 7-1. Power mode state transition diagram

7.5 Power modes shutdown sequencing

When entering stop or other low-power modes, the clocks are shut off in an orderly sequence to safely place the chip in the targeted low-power state. All low-power entry sequences are initiated by the core executing an WFI instruction. The ARM core's outputs, SLEEPDEEP and SLEEPING, trigger entry to the various low-power modes:

- System level wait and VLPW modes equate to: SLEEPING & $\overline{\text{SLEEPDEEP}}$
- All other low power modes equate to: SLEEPING & SLEEPDEEP

When entering the non-wait modes, the chip performs the following sequence:

- Shuts off Core Clock and System Clock to the ARM Cortex-M4 core immediately.
- Polls stop acknowledge indications from the non-core crossbar masters (DMA), supporting peripherals (SPI, PIT) and the Flash Controller for indications that System Clocks, Bus Clock and/or Flash Clock need to be left enabled to complete a previously initiated operation, effectively stalling entry to the targeted low power mode. When all acknowledges are detected, System Clock, Bus Clock and Flash Clock are turned off at the same time.
- MCG and Mode Controller shut off clock sources and/or the internal supplies driven from the on-chip regulator as defined for the targeted low power mode.

In wait modes, most of the system clocks are not affected by the low power mode entry. The Core Clock to the ARM Cortex-M4 core is shut off. Some modules support stop-in-wait functionality and have their clocks disabled under these configurations.

The debugger modules support a transition from stop, wait, VLPS, and VLPW back to a halted state when the debugger is enabled. This transition is initiated by setting the Debug Request bit in MDM-AP control register. As part of this transition, system clocking is re-established and is equivalent to normal run/VLPR mode clocking configuration.

7.6 Module Operation in Low Power Modes

The following table illustrates the functionality of each module while the chip is in each of the low power modes. (Debug modules are discussed separately; see Debug in Low Power Modes.) Number ratings (such as 2 MHz and 1 Mbps) represent the maximum frequencies or maximum data rates per mode. Also, these terms are used:

- FF = Full functionality. In VLPR and VLPW the system frequency is limited, but if a module does not have a limitation in its functionality, it is still listed as FF.
- static = Module register states and associated memories are retained.

- powered = Memory is powered to retain contents.
- low power = Flash has a low power state that retains configuration registers to support faster wakeup.
- OFF = Modules are powered off; module is in reset state upon wakeup.
- wakeup = Modules can serve as a wakeup source for the chip.

Table 7-2. Module operation in low power modes

Modules	Stop	VLPR	VLPW	VLPS	LLS	VLLSx
Core modules						
NVIC	static	FF	FF	static	static	OFF
System modules						
Mode Controller	FF	FF	FF	FF	FF	FF
LLWU[1]	static	static	static	static	FF	FF
Regulator	ON	low power	low power	low power	low power	low power
LVD	ON	disabled	disabled	disabled	disabled	disabled
Brown-out Detection	ON	ON	ON	ON	ON	ON
DMA	static	FF	FF	static	static	OFF
Watchdog	FF	FF	FF	FF	static	OFF
EWM	static	FF	static	static	static	OFF
Clocks						
1kHz LPO	ON	ON	ON	ON	ON	ON
System oscillator (OSC)	OSCERCLK optional	OSCERCLK max of 4MHz crystal	OSCERCLK max of 4MHz crystal	OSCERCLK max of 4MHz crystal	limited to low range/low power	limited to low range/low power
MCG	static - MCGIRCLK optional; PLL optionally on but gated	4 MHz IRC	4 MHz IRC	static - no clock output	static - no clock output	OFF
Core clock	OFF	4 MHz max	OFF	OFF	OFF	OFF
System clock	OFF	4 MHz max	4 MHz max	OFF	OFF	OFF
Bus clock	OFF	4 MHz max	4 MHz max	OFF	OFF	OFF
Memory and memory interfaces						
Flash	powered	1 MHz max access - no pgm	low power	low power	OFF	OFF
Portion of SRAM_U[2]	low power	low power	low power	low power	low power	low power in VLLS3,2
Remaining SRAM_U and all of SRAM_L	low power	low power	low power	low power	low power	low power in VLLS3

Table continues on the next page...

Table 7-2. Module operation in low power modes (continued)

Modules	Stop	VLPR	VLPW	VLPS	LLS	VLLSx
Cache	low power	low power	low power	low power	low power	OFF
FlexMemory	low power	low power[3]	low power	low power	low power	OFF
Register files[4]	powered	powered	powered	powered	powered	powered
NFC	static	FF	FF	static	static	OFF
FlexBus	static	FF	FF	static	static	OFF
EzPort	disabled	disabled	disabled	disabled	disabled	disabled
Communication interfaces						
USB FS/LS	static	static	static	static	static	OFF
USB DCD	static	FF	FF	static	static	OFF
USB Voltage Regulator	optional	optional	optional	optional	optional	optional
UART	static, wakeup on edge	125 kbps	125 kbps	static, wakeup on edge	static	OFF
SPI	static	1 Mbps	1 Mbps	static	static	OFF
I^2C	static, address match wakeup	100 kbps	100 kbps	static, address match wakeup	static	OFF
CAN	wakeup	256 kbps	256 kbps	wakeup	static	OFF
I^2S	FF with external clock[5]	FF	FF	FF with external clock[5]	static	OFF
SDHC	wakeup	FF	FF	wakeup	static	OFF
USB HS	static	static	static	static	static	OFF
Security						
CRC	static	FF	FF	static	static	OFF
Timers						
FTM	static	FF	FF	static	static	OFF
PIT	static	FF	FF	static	static	OFF
PDB	static	FF	FF	static	static	OFF
LPTMR	FF	FF	FF	FF	FF	FF
RTC - 32kHz OSC[4]	FF	FF	FF	FF	FF[6]	FF[6]
CMT	static	FF	FF	static	static	OFF
Analog						
16-bit ADC	ADC internal clock only	FF	FF	ADC internal clock only	static	OFF
CMP[7]	HS or LS compare	FF	FF	HS or LS compare	LS compare	LS compare
6-bit DAC	static	FF	FF	static	static	static

Table continues on the next page...

Table 7-2. Module operation in low power modes (continued)

Modules	Stop	VLPR	VLPW	VLPS	LLS	VLLSx
VREF	FF	FF	FF	FF	static	OFF
12-bit DAC	static	FF	FF	static	static	static
Human-machine interfaces						
GPIO	wakeup	FF	FF	wakeup	static, pins latched	OFF, pins latched
TSI	wakeup	FF	FF	wakeup	wakeup[8]	wakeup[8]

1. Using the LLWU module, the external pins available for this chip do not require the associated peripheral function to be enabled. It only requires the function controlling the pin (GPIO or peripheral) to be configured as an input to allow a transition to occur to the LLWU.
2. A 16KB portion of SRAM_U block is left powered on in low power mode VLLS2.
3. FlexRAM enabled as EEPROM is not writable in VLPR and writes are ignored. Read accesses to FlexRAM as EEPROM while in VLPR are allowed. There are no access restrictions for FlexRAM configured as traditional RAM.
4. These components remain powered in BAT power mode.
5. Use an externally generated bit clock or an externally generated audio master clock (including EXTAL).
6. RTC_CLKOUT is not available.
7. CMP in stop or VLPS supports high speed or low speed external pin to pin or external pin to DAC compares. CMP in LLS or VLLSx only supports low speed external pin to pin or external pin to DAC compares. Windowed, sampled & filtered modes of operation are not available while in stop, VLPS, LLS, or VLLSx modes.
8. TSI wakeup from LLS and VLLSx modes is limited to a single selectable pin.

7.7 Clock Gating

To conserve power, the clocks to most modules can be turned off using the SCGCx registers in the SIM module. These bits are cleared after any reset, which disables the clock to the corresponding module. Prior to initializing a module, set the corresponding bit in the SCGCx register to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution and SIM chapters.

Chapter 8
Security

8.1 Introduction

This device implements security based on the mode selected from the flash module. The following sections provide an overview of flash security and details the effects of security on non-flash modules.

8.2 Flash Security

The flash module provides security information to the MCU based on the state held by the FSEC[SEC] bits. The MCU, in turn, confirms the security request and limits access to flash resources. During reset, the flash module initializes the FSEC register using data read from the security byte of the flash configuration field.

NOTE

> The security features apply only to external accesses: debug and EzPort. CPU accesses to the flash are not affected by the status of FSEC.

In the unsecured state all flash commands are available to the programming interfaces (JTAG and EzPort), as well as user code execution of Flash Controller commands. When the flash is secured (FSEC[SEC] = 00, 01, or 11), programmer interfaces are only allowed to launch mass erase operations and have no access to memory locations.

Further information regarding the flash security options and enabling/disabling flash security is available in the Flash Memory Module.

8.3 Security Interactions with other Modules

The flash security settings are used by the SoC to determine what resources are available. The following sections describe the interactions between modules and the flash security settings or the impact that the flash security has on non-flash modules.

8.3.1 Security interactions with FlexBus

When flash security is enabled, SIM_SOPT2[FBSL] enables/disables off-chip accesses through the FlexBus interface. The FBSL bitfield also has an option to allow opcode and operand accesses or only operand accesses.

8.3.2 Security Interactions with EzPort

When flash security is active the MCU can still boot in EzPort mode. The EzPort holds the flash logic in NVM special mode and thus limits flash operation when flash security is active. While in EzPort mode and security is active, flash bulk erase (BE) can still be executed. The write FCCOB registers (WRFCCOB) command is limited to the mass erase (Erase All Blocks) and verify all 1s (Read 1s All Blocks) commands. Read accesses to internal memories via the EzPort are blocked when security is enabled.

The mass erase can be used to disable flash security, but all of the flash contents are lost in the process. A mass erase via the EzPort is allowed even when some memory locations are protected.

When mass erase has been disabled, mass erase via the EzPort is blocked and cannot be defeated.

8.3.3 Security Interactions with Debug

When flash security is active the JTAG port cannot access the memory resources of the MCU. Boundary scan chain operations work, but debugging capabilities are disabled so that the debug port cannot read flash contents.

Although most debug functions are disabled, the debugger can write to the Flash Mass Erase in Progress bit in the MDM-AP Control register to trigger a mass erase (Erase All Blocks) command. A mass erase via the debugger is allowed even when some memory locations are protected.

When mass erase is disabled, mass erase via the debugger is blocked.

Chapter 9
Debug

9.1 Introduction

This device's debug is based on the ARM coresight architecture and is configured in each device to provide the maximum flexibility as allowed by the restrictions of the pinout and other available resources.

Four debug interfaces are supported:

- IEEE 1149.1 JTAG
- IEEE 1149.7 JTAG (cJTAG)
- Serial Wire Debug (SWD)
- ARM Real-Time Trace Interface

The basic Cortex-M4 debug architecture is very flexible. The following diagram shows the topology of the core debug architecture and its components.

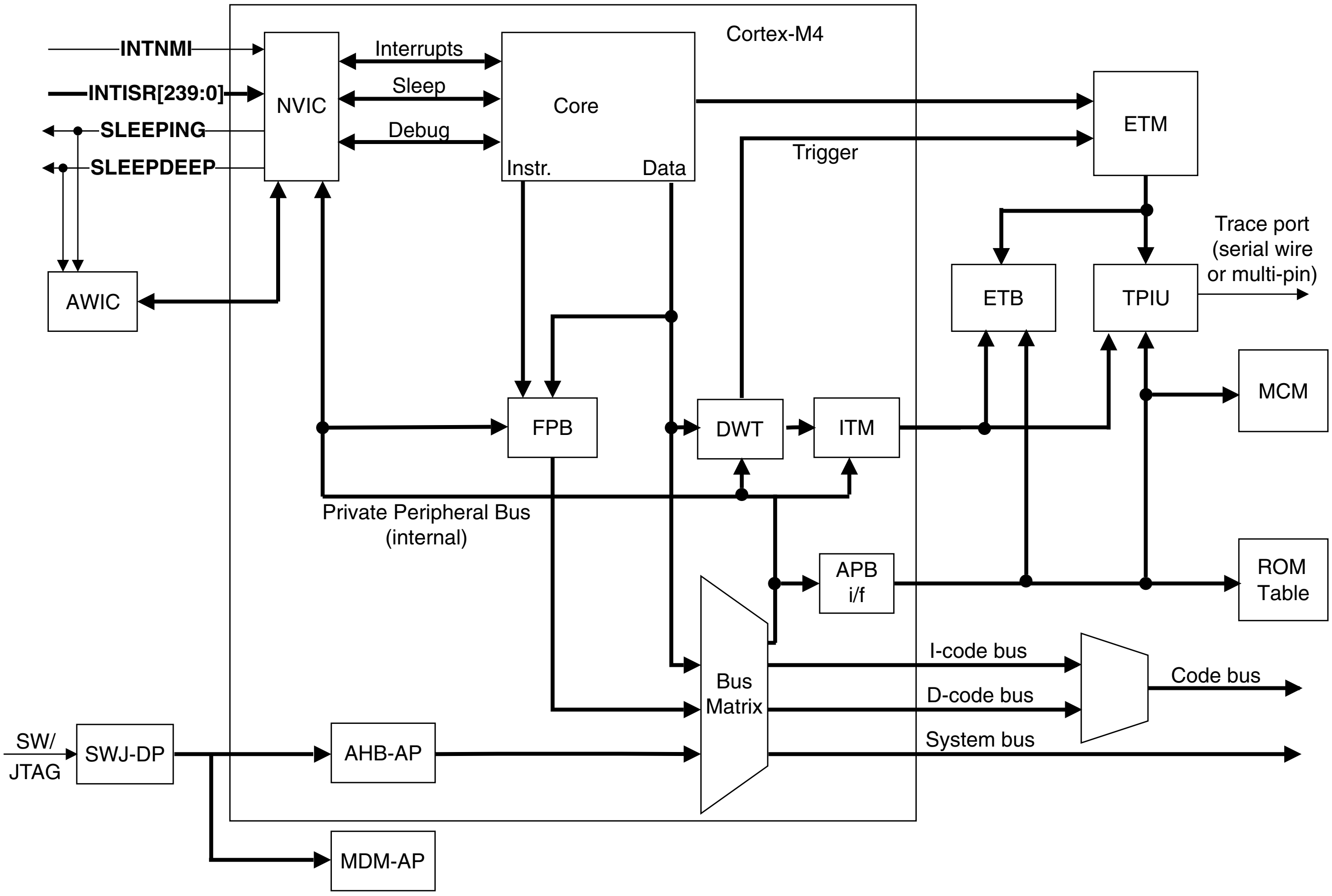


Figure 9-1. Cortex-M4 Debug Topology

The following table presents a brief description of each one of the debug components.

Table 9-1. Debug Components Description

Module	Description
SWJ-DP+ cJTAG	Modified Debug Port with support for SWD, JTAG, cJTAG
AHB-AP	AHB Master Interface from JTAG to debug module and SOC system memory maps
JTAG-AP	Bridge to DFT/BIST resources.
ROM Table	Identifies which debug IP is available.
Core Debug	Singlestep, Register Access, Run, Core Status
CoreSight Trace Funnel (not shown in figure)	The CSTF combines multiple trace streams onto a single ATB bus.
CoreSight Trace Replicator (not shown in figure)	The ATB replicator enables two trace sinks to be wired together and operate from the same incoming trace stream.
ETM (Embedded Trace Macrocell)	ETMv3.5 Architecture
CoreSight ETB (Embedded Trace Buffer)	Memory mapped buffer used to store trace data.
ITM	S/W Instrumentation Messaging + Simple Data Trace Messaging + Watchpoint Messaging

Table continues on the next page...

Table 9-1. Debug Components Description (continued)

Module	Description
DWT (Data and Address Watchpoints)	4 data and address watchpoints (configurable for less, but 4 seems to be accepted)
FPB (Flash Patch and Breakpoints)	The FPB implements hardware breakpoints and patches code and data from code space to system space. The FPB unit contains two literal comparators for matching against literal loads from Code space, and remapping to a corresponding area in System space. The FBP also contains six instruction comparators for matching against instruction fetches from Code space, and remapping to a corresponding area in System space. Alternatively, the six instruction comparators can individually configure the comparators to return a Breakpoint Instruction (BKPT) to the processor core on a match, so providing hardware breakpoint capability.
TPIU (Trace Port Inteface Unit)	Synchronous Mode (5-pin) = TRACE_D[3:0] + TRACE_CLKOUT Synchronous Mode (3-pin) = TRACE_D[1:0] + TRACE_CLKOUT Asynchronous Mode (1-pin) = TRACE_SWO (available on JTAG_TDO)
MCM (Miscellaneous Control Module)	The MCM provides miscellaneous control functions including control of the ETB and trace path switching.

9.1.1 References

For more information on ARM debug components, see these documents:

- ARMv7-M Architecture Reference Manual
- ARM Debug Interface v5.1
- ARM CoreSight Architecture Specification
- ARM ETM Architecture Specification v3.5

9.2 The Debug Port

The configuration of the cJTAG module, JTAG controller, and debug port is illustrated in the following figure:

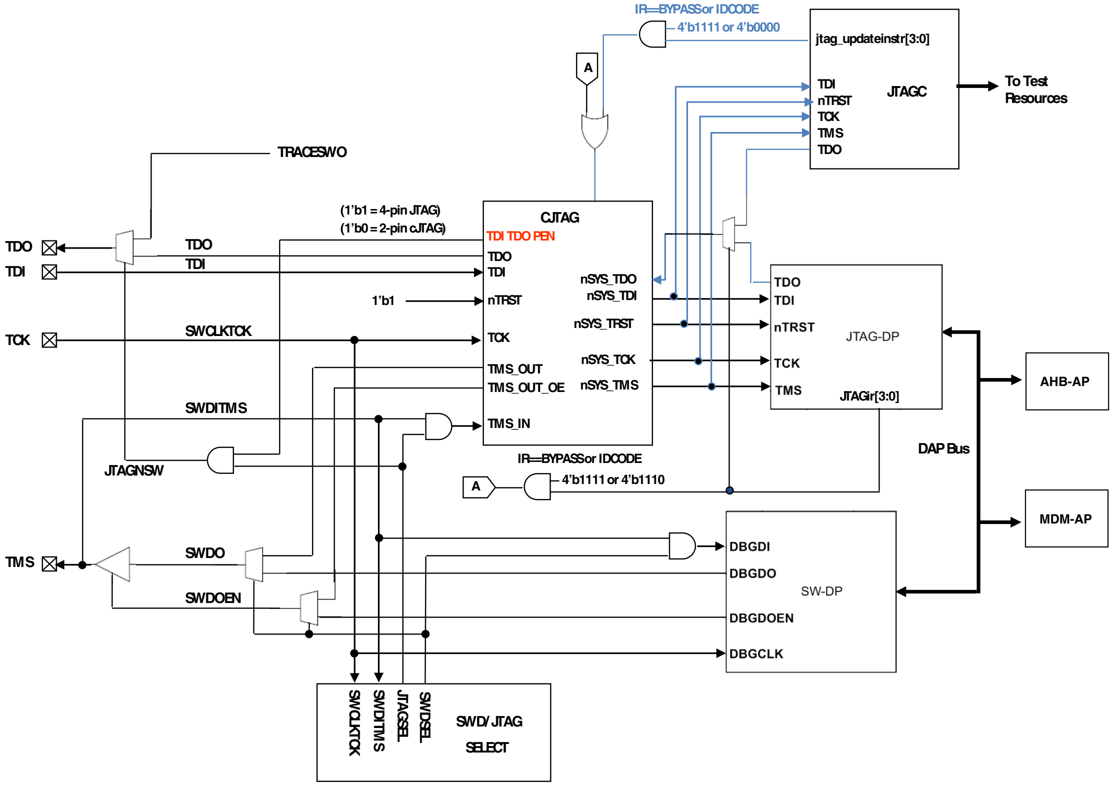


Figure 9-2. Modified Debug Port

The debug port comes out of reset in standard JTAG mode and is switched into either cJTAG or SWD mode by the following sequences. Once the mode has been changed, unused debug pins can be reassigned to any of their alternative muxed functions.

9.2.1 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

NOTE

See the ARM documentation for the CoreSight DAP Lite for restrictions.

9.2.2 JTAG-to-cJTAG change sequence

1. Reset the debug port

2. Set the control level to 2 via zero-bit scans
3. Execute the Store Format (STFMT) command (00011) to set the scan format register to 1149.7 scan format

9.3 Debug Port Pin Descriptions

The debug port pins default after POR to their JTAG functionality with the exception of JTAG_TRST_b and can be later reassigned to their alternate functionalities. In cJTAG and SWD modes JTAG_TDI and JTAG_TRST_b can be configured to alternate GPIO functions.

Table 9-2. Debug port pins

Pin Name	JTAG Debug Port		cJTAG Debug Port		SWD Debug Port		Internal Pull-up\Down
	Type	Description	Type	Description	Type	Description	
JTAG_TMS/ SWD_DIO	I/O	JTAG Test Mode Selection	I/O	cJTAG Data	I/O	Serial Wire Data	Pull-up
JTAG_TCLK/ SWD_CLK	I	JTAG Test Clock	I	cJTAG Clock	I	Serial Wire Clock	Pull-down
JTAG_TDI	I	JTAG Test Data Input	-	-	-	-	Pull-up
JTAG_TDO/ TRACE_SWO	O	JTAG Test Data Output	O	Trace output over a single pin	O	Trace output over a single pin	N/C
JTAG_TRST_b	I	JTAG Reset	I	cJTAG Reset	-	-	Pull-up

9.4 System TAP connection

The system JTAG controller is connected in parallel to the ARM TAP controller. The system JTAG controller IR codes overlay the ARM JTAG controller IR codes without conflict. Refer to the IR codes table for a list of the available IR codes. The output of the TAPs (TDO) are muxed based on the IR code which is selected. This design is fully JTAG compliant and appears to the JTAG chain as a single TAP. At power on reset, ARM's IDCODE (IR=4'b1110) is selected.

9.4.1 IR Codes

Table 9-3. 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 while applying preloaded values to output pins and asserting functional reset
HIGHZ	1001	Selects bypass register while three-stating all output pins and asserting functional reset
CLAMP	1100	Selects bypass register while applying preloaded values to output pins and asserting functional reset
EZPORT	1101	Enables the EZPORT function for the SoC and asserts functional reset.
ARM_IDCODE	1110	ARM JTAG-DP Instruction
BYPASS	1111	Selects bypass register for data operations
Factory debug reserved	0101, 0110, 0111	Intended for factory debug only
ARM JTAG-DP Reserved	1000, 1010, 1011, 1110	These instructions will go the ARM JTAG-DP controller. Please look at ARM JTAG-DP documentation for more information on these instructions.
Reserved [1]	All other opcodes	Decoded to select bypass register

1. The manufacturer reserves the right to change the decoding of reserved instruction codes in the future

9.5 JTAG status and control registers

Through the ARM Debug Access Port (DAP), the debugger has access to the status and control elements, implemented as registers on the DAP bus as shown in the following figure. These registers provide additional control and status for low power mode recovery and typical run-control scenarios. The status register bits also provide a means for the debugger to get updated status of the core without having to initiate a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

It is important to note that these DAP control and status registers are not memory mapped within the system memory map and are only accessible via the Debug Access Port (DAP) using JTAG, cJTAG, or SWD. The MDM-AP is accessible as Debug Access Port 1 with the available registers shown in the table below.

Table 9-4. MDM-AP Register Summary

Address	Register	Description
0x0100_0000	Status	See MDM-AP Status Register
0x0100_0004	Control	See MDM-AP Control Register
0x0100_00FC	ID	Read-only identification register that always reads as 0x001C_0000

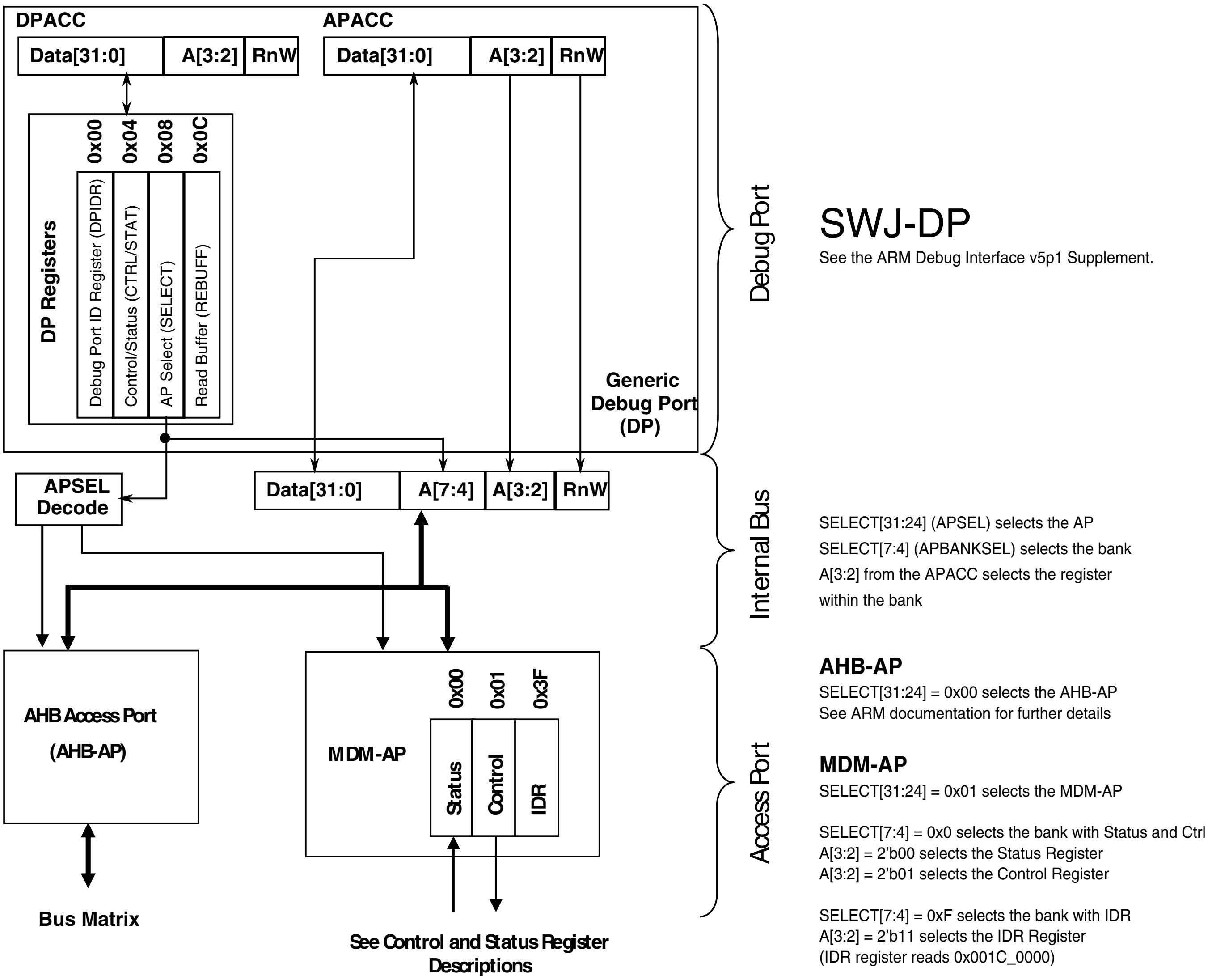

DPACC
Data[31:0]
A[3:2]
RnW
APACC
Data[31:0]
A[3:2]
RnW
0x00
0x04
0x08
0x0C
DP Registers
Debug Port ID Register (DPIDR)
Control/Status (CTRL/STAT)
AP Select (SELECT)
Read Buffer (REBUFF)
Generic Debug Port (DP)
Debug Port
SWJ-DP
See the ARM Debug Interface v5p1 Supplement.
APSEL Decode
Data[31:0]
A[7:4]
A[3:2]
RnW
Internal Bus
SELECT[31:24] (APSEL) selects the AP
SELECT[7:4] (APBANKSEL) selects the bank
A[3:2] from the APACC selects the register
within the bank
AHB Access Port
(AHB-AP)
MDM-AP
0x00
0x01
0x3F
Status
Control
IDR
Access Port
AHB-AP
SELECT[31:24] = 0x00 selects the AHB-AP
See ARM documentation for further details
MDM-AP
SELECT[31:24] = 0x01 selects the MDM-AP
SELECT[7:4] = 0x0 selects the bank with Status and Ctrl
A[3:2] = 2'b00 selects the Status Register
A[3:2] = 2'b01 selects the Control Register
SELECT[7:4] = 0xF selects the bank with IDR
A[3:2] = 2'b11 selects the IDR Register
(IDR register reads 0x001C_0000)
Bus Matrix
See Control and Status Register Descriptions

Figure 9-3. MDM AP Addressing

9.5.1 MDM-AP Control Register

Table 9-5. MDM-AP Control register assignments

Bit	Name	Secure[1]	Description
0	Flash Mass Erase in Progress	Y	Set to cause mass erase. Cleared by hardware after mass erase operation completes. When mass erase is disabled (via MEEN and SEC settings), the erase request does not occur and the Flash Mass Erase in Progress bit continues to assert until the next system reset.
1	Debug Disable	N	Set to disable debug. Clear to allow debug operation. When set it overrides the C_DEBUGEN bit within the DHCSR and force disables Debug logic.
2	Debug Request	N	Set to force the Core to halt. If the Core is in a stop or wait mode, this bit can be used to wakeup the core and transition to a halted state.
3	System Reset Request	N	Set to force a system reset. The system remains held in reset until this bit is cleared.
4	Core Hold Reset	N	Configuration bit to control Core operation at the end of system reset sequencing. 0 Normal operation - release the Core from reset along with the rest of the system at the end of system reset sequencing. 1 Suspend operation - hold the Core in reset at the end of reset sequencing. Once the system enters this suspended state, clearing this control bit immediately releases the Core from reset and CPU operation begins.
5	VLLSx Debug Request (VLLDBGREQ)	N	Set to configure the system to be held in reset after the next recovery from a VLLSx mode. This bit is ignored on a VLLS wakeup via the Reset pin. During a VLLS wakeup via the Reset pin, the system can be held in reset by holding the reset pin asserted allowing the debugger to re-initialize the debug modules. This bit holds the system in reset when VLLSx modes are exited to allow the debugger time to re-initialize debug IP before the debug session continues. The Mode Controller captures this bit logic on entry to VLLSx modes. Upon exit from VLLSx modes, the Mode Controller will hold the system in reset until VLLDBGACK is asserted. The VLLDBGREQ bit clears automatically due to the POR reset generated as part of the VLLSx recovery.
6	VLLSx Debug Acknowledge (VLLDBGACK)	N	Set to release a system being held in reset following a VLLSx recovery This bit is used by the debugger to release the system reset when it is being held on VLLSx mode exit. The debugger re-initializes all debug IP and then assert this control bit to allow the Mode Controller to release the system from reset and allow CPU operation to begin. The VLLDBGACK bit is cleared by the debugger or can be left set because it clears automatically due to the POR reset generated as part of the next VLLSx recovery.

Table continues on the next page...

Table 9-5. MDM-AP Control register assignments (continued)

Bit	Name	Secure[1]	Description
7	LLS, VLLSx Status Acknowledge	N	Set this bit to acknowledge the DAP LLS and VLLS Status bits have been read. This acknowledge automatically clears the status bits. This bit is used by the debugger to clear the sticky LLS and VLLSx mode entry status bits. This bit is asserted and cleared by the debugger.
8	Timestamp Disable	N	Set this bit to disable the 48-bit global trace timestamp counter during debug halt mode when the core is halted. 0 The timestamp counter continues to count assuming trace is enabled and the ETM is enabled. (default) 1 The timestamp counter freezes when the core has halted (debug halt mode).
9 – 31	Reserved for future use	N	

[1] Command available in secure mode

9.5.2 MDM-AP Status Register

Table 9-6. MDM-AP Status register assignments

Bit	Name	Description
0	Flash Mass Erase Acknowledge	The Flash Mass Erase Acknowledge bit is cleared after any system reset. The bit is also cleared at launch of a mass erase command due to write of Flash Mass Erase in Progress bit in MDM AP Control Register. The Flash Mass Erase Acknowledge is set after Flash control logic has started the mass erase operation. When mass erase is disabled (via MEEN and SEC settings), an erase request due to seting of Flash Mass Erase in Progress bit is not acknowledged.
1	Flash Ready	Indicate Flash has been initialized and debugger can be configured even if system is continuing to be held in reset via the debugger.
2	System Security	Indicates the security state. When secure, the debugger does not have access to the system bus or any memory mapped peripherals. This bit indicates when the part is locked and no system bus access is possible.
3	System Reset	Indicates the system reset state. 0 System is in reset 1 System is not in reset
4	Reserved	
5	Mass Erase Enable	Indicates if the MCU can be mass erased or not 0 Mass erase is disabled 1 Mass erase is enabled

Table continues on the next page...

Table 9-6. MDM-AP Status register assignments (continued)

Bit	Name	Description
6	Backdoor Access Key Enable	Indicates if the MCU has the backdoor access key enabled. 0 Disabled 1 Enabled
7	LP Enabled	Decode of LPLLSM control bits to indicate that VLPS, LLS, or VLLSx are the selected power mode the next time the ARM Core enters Deep Sleep. 0 Low Power Stop Mode is not enabled 1 Low Power Stop Mode is enabled Usage intended for debug operation in which Run to VLPS is attempted. Per debug definition, the system actually enters the Stop state. A debugger should interpret deep sleep indication (with SLEEPDEEP and SLEEPING asserted), in conjuntion with this bit asserted as the debugger-VLPS status indication.
8	Very Low Power Mode	Indicates current power mode is VLPx. This bit is not 'sticky' and should always represent whether VLPx is enabled or not. This bit is used to throttle JTAG TCK frequency up/down.
9	LLS Mode Exit	This bit indicates an exit from LLS mode has occurred. The debugger will lose communication while the system is in LLS (including access to this register). Once communication is reestablished, this bit indicates that the system had been in LLS. Since the debug modules held their state during LLS, they do not need to be reconfigured. This bit is set during the LLS recovery sequence. The LLS Mode Exit bit is held until the debugger has had a chance to recognize that LLS was exited and is cleared by a write of 1 to the LLS, VLLSx Status Acknowledge bit in MDM AP Control register.
10	VLLSx Modes Exit	This bit indicates an exit from VLLSx mode has occurred. The debugger will lose communication while the system is in VLLSx (including access to this register). Once communication is reestablished, this bit indicates that the system had been in VLLSx. Since the debug modules lose their state during VLLSx modes, they need to be reconfigured. This bit is set during the VLLSx recovery sequence. The VLLSx Mode Exit bit is held until the debugger has had a chance to recognize that a VLLS mode was exited and is cleared by a write of 1 to the LLS, VLLSx Status Acknowledge bit in MDM AP Control register.
11 – 15	Reserved for future use	Always read 0.
16	Core Halted	Indicates the Core has entered debug halt mode
17	Core SLEEPDEEP	Indicates the Core has entered a low power mode SLEEPING==1 and SLEEPDEEP==0 indicates wait or VLPW mode. SLEEPING==1 and SLEEPDEEP==1 indicates stop or VLPS mode.
18	Core SLEEPING	
19 – 31	Reserved for future use	Always read 0.

9.6 Debug Resets

The debug system receives the following sources of reset:

- JTAG_TRST_b from an external signal. This signal is optional and may not be available in all packages.
- Debug reset (CDBGRSTREQ bit within the SWJ-DP CTRL/STAT register) in the TCLK domain that allows the debugger to reset the debug logic.
- TRST asserted via the cJTAG escape command.
- System POR reset

Conversely the debug system is capable of generating system reset using the following mechanism:

- A system reset in the DAP control register which allows the debugger to hold the system in reset.
- SYSRESETREQ bit in the NVIC application interrupt and reset control register
- A system reset in the DAP control register which allows the debugger to hold the Core in reset.

9.7 AHB-AP

AHB-AP provides the debugger access to all memory and registers in the system, including processor registers through the NVIC. System access is independent of the processor status. AHB-AP does not do back-to-back transactions on the bus, so all transactions are non-sequential. AHB-AP can perform unaligned and bit-band transactions. AHB-AP transactions bypass the FPB, so the FPB cannot remap AHB-AP transactions. SWJ/SW-DP-initiated transaction aborts drive an AHB-AP-supported sideband signal called HABORT. This signal is driven into the Bus Matrix, which resets the Bus Matrix state, so that AHB-AP can access the Private Peripheral Bus for last ditch debugging such as read/stop/reset the core. AHB-AP transactions are little endian.

The MPU includes default settings and protections for the Region Descriptor 0 (RGD0) such that the Debugger always has access to the entire address space and those rights cannot be changed by the core or any other bus master.

For a short period at the start of a system reset event the system security status is being determined and debugger access to all AHB-AP transactions is blocked. The MDM-AP Status register is accessible and can be monitored to determine when this initial period is completed. After this initial period, if system reset is held via assertion of the RESET pin, the debugger has access via the bus matrix to the private peripheral bus to configure the debug IP even while system reset is asserted. While in system reset, access to other memory and register resources, accessed over the Crossbar Switch, is blocked.

9.8 ITM

The ITM is an application-driven trace source that supports printf style debugging to trace Operating System (OS) and application events, and emits diagnostic system information. The ITM emits trace information as packets. There are four sources that can generate packets. If multiple sources generate packets at the same time, the ITM arbitrates the order in which packets are output. The four sources in decreasing order of priority are:

1. Software trace -- Software can write directly to ITM stimulus registers. This emits packets.
2. Hardware trace -- The DWT generates these packets, and the ITM emits them.
3. Time stamping -- Timestamps are emitted relative to packets. The ITM contains a 21-bit counter to generate the timestamp. The Cortex-M4 clock or the bitclock rate of the Serial Wire Viewer (SWV) output clocks the counter.
4. Global system timestamping. Timestamps can optionally be generated using a system-wide 48-bit count value. The same count value can be used to insert timestamps in the ETM trace stream, allowing coarse-grain correlation.

9.9 Core Trace Connectivity

9.10 Embedded Trace Macrocell v3.5 (ETM)

The Cortex-M4 Embedded Trace Macrocell (ETM-M4) is a debug component that enables a debugger to reconstruct program execution. The CoreSight ETM-M4 supports only instruction trace. You can use it either with the Cortex-M4 Trace Port Interface Unit (M4-TPIU), or with the CoreSight ETB.

The main features of an ETM are:

- tracing of 16-bit and 32-bit Thumb instructions
- four EmbeddedICE watchpoint inputs
- a Trace Start/Stop block with EmbeddedICE inputs
- one reduced function counter
- two external inputs

- a 24-byte FIFO queue
- global timestamping

9.11 Coresight Embedded Trace Buffer (ETB)

The ETB provides on-chip storage of trace data using 32-bit RAM. The ETB accepts trace data from any CoreSight-compliant component trace source with an ATB master port, such as a trace source or a trace funnel. It is included in this device to remove dependencies from the trace pin pad speed, and enable low cost trace solutions. The TraceRAM size is 2 KB.

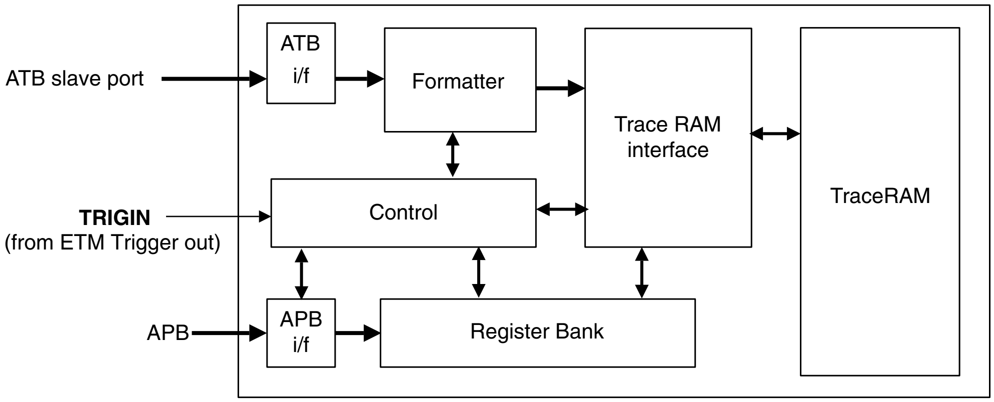


Figure 9-4. ETB Block Diagram

The ETB contains the following blocks:

- Formatter -- Inserts source ID signals into the data packet stream so that trace data can be re-associated with its trace source after the data is read back out of the ETB.
- Control -- Control registers for trace capture and flushing.
- APB interface -- Read, write, and data pointers provide access to ETB registers. In addition, the APB interface supports wait states through the use of a PREADYDBG signal output by the ETB. The APB interface is synchronous to the ATB domain.
- Register bank -- Contains the management, control, and status registers for triggers, flushing behavior, and external control.
- Trace RAM interface -- Controls reads and writes to the Trace RAM.

9.11.1 Performance Profiling with the ETB

To create a performance profile (e.g. gprof) for the target application, a means to collect trace over a long period of time is needed. The ETB buffer is too small to capture a meaningful profile in just one take. What is needed is to collect and concatenate data from the ETB buffer for multiple sequential runs. Using the ETB packet counter (described in Miscellaneous Control Module (MCM)), the trace analysis tool can capture multiple sequential runs by executing code until the ETB is almost full, and halting or executing an interrupt handler to allow the buffer to be emptied, and then continuing executing code. The target halts or executes an interrupt handler when the buffer is almost full to empty the data and then the debugger runs the target again.

9.11.2 ETB Counter Control

The ETB packet counter is controlled by the ETB counter control register, ETB reload register, and ETB counter value register implemented in the Miscellaneous Control Module (MCM) accessible via the Private Peripheral Bus. Via the ETB counter control register the ETB control logic can be configured to cause an MCM Alert Interrupt, an NMI Interrupt, or cause a Debug halt when the down counter reaches 0. Other features of the ETB control logic include:

- Down counter to count as many as 512 x 32-bit packets.
- Reload request transfers reload value to counter.
- ATB valid and ready signals used to form counter decrement.
- The counter disarms itself when the count reaches 0.

9.12 TPIU

The TPIU acts as a bridge between the on-chip trace data from the Embedded Trace Macrocell (ETM) and the Instrumentation Trace Macrocell (ITM), with separate IDs, to a data stream, encapsulating IDs where required, that is then captured by a Trace Port Analyzer (TPA). The TPIU is specially designed for low-cost debug.

9.13 DWT

The DWT is a unit that performs the following debug functionality:

- It contains four comparators that you can configure as a hardware watchpoint, an ETM trigger, a PC sampler event trigger, or a data address sampler event trigger. The first comparator, DWT_COMP0, can also compare against the clock cycle counter, CYCCNT. The second comparator, DWT_COMP1, can also be used as a data comparator.
- The DWT contains counters for:
  - Clock cycles (CYCCNT)
  - Folded instructions
  - Load store unit (LSU) operations
  - Sleep cycles
  - CPI (all instruction cycles except for the first cycle)
  - Interrupt overhead

NOTE

An event is emitted each time a counter overflows.

- The DWT can be configured to emit PC samples at defined intervals, and to emit interrupt event information.

9.14 Debug in Low Power Modes

In low power modes in which the debug modules are kept static or powered off, the debugger cannot gather any debug data for the duration of the low power mode. In the case that the debugger is held static, the debug port returns to full functionality as soon as the low power mode exits and the system returns to a state with active debug. In the case that the debugger logic is powered off, the debugger is reset on recovery and must be reconfigured once the low power mode is exited.

Power mode entry logic monitors Debug Power Up and System Power Up signals from the debug port as indications that a debugger is active. These signals can be changed in RUN, VLPR, WAIT and VLPW. If the debug signal is active and the system attempts to enter stop or VLPS, FCLK continues to run to support core register access. In these modes in which FCLK is left active the debug modules have access to core registers but not to system memory resources accessed via the crossbar.

With debug enabled, transitions from Run directly to VLPS are not allowed and result in the system entering Stop mode instead. Status bits within the MDM-AP Status register can be evaluated to determine this pseudo-VLPS state. Note with the debug enabled, transitions from Run--> VLPR --> VLPS are still possible but also result in the system entering Stop mode instead.

In VLLS mode all debug modules are powered off and reset at wakeup. In LLS mode, the debug modules retain their state but no debug activity is possible.

NOTE

When using cJTAG and entering LLS mode, the cJTAG controller must be reset on exit from LLS mode.

Going into a VLLSx mode causes all the debug controls and settings to be reset. To give time to the debugger to sync up with the HW, the MDM-AP Control register can be configured hold the system in reset on recovery so that the debugger can regain control and reconfigure debug logic prior to the system exiting reset and resuming operation.

9.14.1 Debug Module State in Low Power Modes

The following table shows the state of the debug modules in low power modes. These terms are used:

- FF = Full functionality. In VLPR and VLPW the system frequency is limited, but if a module does not have a limitation in its functionality, it is still listed as FF.
- static = Module register states and associated memories are retained.
- OFF = Modules are powered off; module is in reset state upon wakeup.

Table 9-7. Debug Module State in Low Power Modes

Module	STOP	VLPR	VLPW	VLPS	LLS	VLLSx
Debug Port	FF	FF	FF	OFF	static	OFF
AHB-AP	FF	FF	FF	OFF	static	OFF
ITM	FF	FF	FF	OFF	static	OFF
ETM	FF	FF	FF	OFF	static	OFF
ETB	FF	FF	FF	OFF	static	OFF
TPIU	FF	FF	FF	OFF	static	OFF
DWT	FF	FF	FF	OFF	static	OFF

9.15 Debug & Security

When security is enabled (FSEC[SEC] != 10), the debug port capabilities are limited in order to prevent exploitation of secure data. In the secure state the debugger still has access to the MDM-AP Status Register and can determine the current security state of the device. In the case of a secure device, the debugger also has the capability of performing a mass erase operation via writes to the MDM-AP Control Register. In the case of a secure device that has mass erase disabled (FSEC[MEEN] = 10), attempts to mass erase via the debug interface are blocked.

Chapter 10
Signal Multiplexing and Signal Descriptions

10.1 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 Port Control block controls which signal is present on the external pin. Reference that chapter to find which register controls the operation of a specific pin.

10.2 Signal Multiplexing Integration

This section summarizes how the module is integrated into the device. For a comprehensive description of the module itself, see the module’s dedicated chapter.

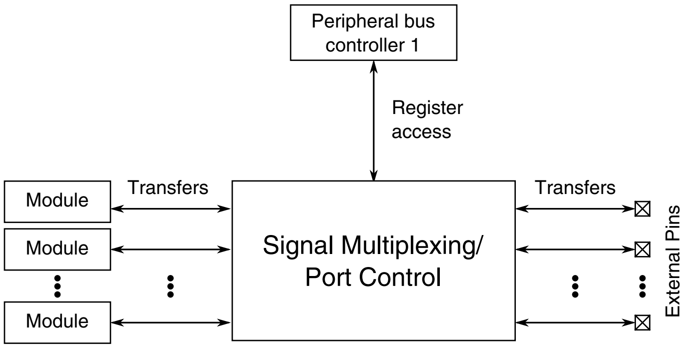


Figure 10-1. Signal multiplexing integration

Table 10-1. Reference links to related information

Topic	Related module	Reference
Full description	Port control	Port control
System memory map		System memory map

Table continues on the next page...

Table 10-1. Reference links to related information (continued)

Topic	Related module	Reference
Clocking		Clock Distribution
Register access	Peripheral bus controller	Peripheral bridge

10.2.1 Port control and interrupt module features

- Six 32-pin ports

NOTE

Not all pins are available on the device. See the following section for details.

- Each 32-pin port is assigned one interrupt.
- The digital filter option has two clock source options: bus clock and 1-kHz LPO. The 1-kHz LPO option gives users this feature in low power modes.
- The digital filter is configurable from 1 to 32 clock cycles when enabled.

10.2.2 Clock gating

The clock to the port control module can be gated on and off using the SCGC5[PORTx] bits in the SIM module. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing the corresponding module, set SCGC5[PORTx] in the SIM module to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution chapter.

10.2.3 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.

10.3 Pinout

10.3.1 K20 Signal Multiplexing and Pin Assignments

The following table shows the signals available on each pin and the locations of these pins on the devices supported by this document. The Port Control Module is responsible for selecting which ALT functionality is available on each pin.

144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
—	L5	RTC_WAKEUP_B	RTC_WAKEUP_B	RTC_WAKEUP_B								
—	M5	NC	NC	NC								
—	A10	NC	NC	NC								
—	B10	NC	NC	NC								
—	C10	NC	NC	NC								
1	D3	PTE0	ADC1_SE4a	ADC1_SE4a	PTE0	SPI1_PCS1	UART1_TX	SDHC0_D1		I2C1_SDA	RTC_CLKOUT	
2	D2	PTE1/LLWU_P0	ADC1_SE5a	ADC1_SE5a	PTE1/LLWU_P0	SPI1_SOUT	UART1_RX	SDHC0_D0		I2C1_SCL	SPI1_SIN	
3	D1	PTE2/LLWU_P1	ADC1_SE6a	ADC1_SE6a	PTE2/LLWU_P1	SPI1_SCK	UART1_CTS_b	SDHC0_DCLK				
4	E4	PTE3	ADC1_SE7a	ADC1_SE7a	PTE3	SPI1_SIN	UART1_RTS_b	SDHC0_CMD			SPI1_SOUT	
5	E5	VDD	VDD	VDD								
6	F6	VSS	VSS	VSS								
7	E3	PTE4/LLWU_P2	DISABLED		PTE4/LLWU_P2	SPI1_PCS0	UART3_TX	SDHC0_D3				
8	E2	PTE5	DISABLED		PTE5	SPI1_PCS2	UART3_RX	SDHC0_D2		FTM3_CH0		
9	E1	PTE6	DISABLED		PTE6	SPI1_PCS3	UART3_CTS_b	I2S0_MCLK		FTM3_CH1	USB_SOF_OUT	
10	F4	PTE7	DISABLED		PTE7		UART3_RTS_b	I2S0_RXD0		FTM3_CH2		
11	F3	PTE8	ADC2_SE16	ADC2_SE16	PTE8	I2S0_RXD1	UART5_TX	I2S0_RX_FS		FTM3_CH3		
12	F2	PTE9	ADC2_SE17	ADC2_SE17	PTE9	I2S0_TXD1	UART5_RX	I2S0_RX_BCLK		FTM3_CH4		
13	F1	PTE10	DISABLED		PTE10		UART5_CTS_b	I2S0_TXD0		FTM3_CH5		
14	G4	PTE11	ADC3_SE16	ADC3_SE16	PTE11		UART5_RTS_b	I2S0_TX_FS		FTM3_CH6		
15	G3	PTE12	ADC3_SE17	ADC3_SE17	PTE12			I2S0_TX_BCLK		FTM3_CH7		
16	E6	VDD	VDD	VDD								
17	F7	VSS	VSS	VSS								
18	H3	VSS	VSS	VSS								
19	H1	USB0_DP	USB0_DP	USB0_DP								

144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
20	H2	USB0_DM	USB0_DM	USB0_DM								
21	G1	VOUT33	VOUT33	VOUT33								
22	G2	VREGIN	VREGIN	VREGIN								
23	J1	PGA2_DP/ADC2_DP0/ADC3_DP3/ADC0_DP1	PGA2_DP/ADC2_DP0/ADC3_DP3/ADC0_DP1	PGA2_DP/ADC2_DP0/ADC3_DP3/ADC0_DP1								
24	J2	PGA2_DM/ADC2_DM0/ADC3_DM3/ADC0_DM1	PGA2_DM/ADC2_DM0/ADC3_DM3/ADC0_DM1	PGA2_DM/ADC2_DM0/ADC3_DM3/ADC0_DM1								
25	K1	PGA3_DP/ADC3_DP0/ADC2_DP3/ADC1_DP1	PGA3_DP/ADC3_DP0/ADC2_DP3/ADC1_DP1	PGA3_DP/ADC3_DP0/ADC2_DP3/ADC1_DP1								
26	K2	PGA3_DM/ADC3_DM0/ADC2_DM3/ADC1_DM1	PGA3_DM/ADC3_DM0/ADC2_DM3/ADC1_DM1	PGA3_DM/ADC3_DM0/ADC2_DM3/ADC1_DM1								
27	L1	PGA0_DP/ADC0_DP0/ADC1_DP3	PGA0_DP/ADC0_DP0/ADC1_DP3	PGA0_DP/ADC0_DP0/ADC1_DP3								
28	L2	PGA0_DM/ADC0_DM0/ADC1_DM3	PGA0_DM/ADC0_DM0/ADC1_DM3	PGA0_DM/ADC0_DM0/ADC1_DM3								
29	M1	PGA1_DP/ADC1_DP0/ADC0_DP3	PGA1_DP/ADC1_DP0/ADC0_DP3	PGA1_DP/ADC1_DP0/ADC0_DP3								
30	M2	PGA1_DM/ADC1_DM0/ADC0_DM3	PGA1_DM/ADC1_DM0/ADC0_DM3	PGA1_DM/ADC1_DM0/ADC0_DM3								
31	H5	VDDA	VDDA	VDDA								
32	G5	VREFH	VREFH	VREFH								
33	G6	VREFL	VREFL	VREFL								
34	H6	VSSA	VSSA	VSSA								
35	K3	ADC1_SE16/CMP2_IN2/ADC0_SE22	ADC1_SE16/CMP2_IN2/ADC0_SE22	ADC1_SE16/CMP2_IN2/ADC0_SE22								
36	J3	ADC0_SE16/CMP1_IN2/ADC0_SE21	ADC0_SE16/CMP1_IN2/ADC0_SE21	ADC0_SE16/CMP1_IN2/ADC0_SE21								
37	M3	VREF_OUT/CMP1_IN5/CMP0_IN5/ADC1_SE18	VREF_OUT/CMP1_IN5/CMP0_IN5/ADC1_SE18	VREF_OUT/CMP1_IN5/CMP0_IN5/ADC1_SE18								

144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
38	L3	DAC0_OUT/CMP1_IN3/ADC0_SE23	DAC0_OUT/CMP1_IN3/ADC0_SE23	DAC0_OUT/CMP1_IN3/ADC0_SE23								
39	L4	DAC1_OUT/CMP0_IN4/CMP2_IN3/ADC1_SE23	DAC1_OUT/CMP0_IN4/CMP2_IN3/ADC1_SE23	DAC1_OUT/CMP0_IN4/CMP2_IN3/ADC1_SE23								
40	M7	XTAL32	XTAL32	XTAL32								
41	M6	EXTAL32	EXTAL32	EXTAL32								
42	L6	VBAT	VBAT	VBAT								
43	—	VDD	VDD	VDD								
44	—	VSS	VSS	VSS								
45	M4	PTE24	ADC0_SE17/EXTAL1	ADC0_SE17/EXTAL1	PTE24	CAN1_TX	UART4_TX	I2S1_TX_FS		EWM_OUT_b	I2S1_RXD1	
46	K5	PTE25	ADC0_SE18/XTAL1	ADC0_SE18/XTAL1	PTE25	CAN1_RX	UART4_RX	I2S1_TX_BCLK		EWM_IN	I2S1_TXD1	
47	K4	PTE26	ADC3_SE5b	ADC3_SE5b	PTE26		UART4_CTS_b	I2S1_TXD0		RTC_CLKOUT	USB_CLKIN	
48	J4	PTE27	ADC3_SE4b	ADC3_SE4b	PTE27		UART4_RTS_b	I2S1_MCLK				
49	H4	PTE28	ADC3_SE7a	ADC3_SE7a	PTE28							
50	J5	PTA0	JTAG_TCLK/SWD_CLK/EZP_CLK	TSI0_CH1	PTA0	UART0_CTS_b/UART0_COL_b	FTM0_CH5				JTAG_TCLK/SWD_CLK	EZP_CLK
51	J6	PTA1	JTAG_TDI/EZP_DI	TSI0_CH2	PTA1	UART0_RX	FTM0_CH6				JTAG_TDI	EZP_DI
52	K6	PTA2	JTAG_TDO/TRACE_SWO/EZP_DO	TSI0_CH3	PTA2	UART0_TX	FTM0_CH7				JTAG_TDO/TRACE_SWO	EZP_DO
53	K7	PTA3	JTAG_TMS/SWD_DIO	TSI0_CH4	PTA3	UART0_RTS_b	FTM0_CH0				JTAG_TMS/SWD_DIO	
54	L7	PTA4/LLWU_P3	NMI_b/EZP_CS_b	TSI0_CH5	PTA4/LLWU_P3		FTM0_CH1				NMI_b	EZP_CS_b
55	M8	PTA5	DISABLED		PTA5	USB_CLKIN	FTM0_CH2		CMP2_OUT	I2S0_TX_BCLK	JTAG_TRST_b	
56	E7	VDD	VDD	VDD								
57	G7	VSS	VSS	VSS								
58	J7	PTA6	ADC3_SE6a	ADC3_SE6a	PTA6	ULPI_CLK	FTM0_CH3	I2S1_RXD0			TRACE_CLKOUT	
59	J8	PTA7	ADC0_SE10	ADC0_SE10	PTA7	ULPI_DIR	FTM0_CH4	I2S1_RX_BCLK			TRACE_D3	
60	K8	PTA8	ADC0_SE11	ADC0_SE11	PTA8	ULPI_NXT	FTM1_CH0	I2S1_RX_FS		FTM1_QD_PHA	TRACE_D2	

144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
61	L8	PTA9	ADC3_SE5a	ADC3_SE5a	PTA9	ULPI_STP	FTM1_CH1			FTM1_QD_PHB	TRACE_D1	
62	M9	PTA10	ADC3_SE4a	ADC3_SE4a	PTA10	ULPI_DATA0	FTM2_CH0			FTM2_QD_PHA	TRACE_D0	
63	L9	PTA11	ADC3_SE15	ADC3_SE15	PTA11	ULPI_DATA1	FTM2_CH1			FTM2_QD_PHB		
64	K9	PTA12	CMP2_IN0	CMP2_IN0	PTA12	CAN0_TX	FTM1_CH0			I2S0_TXD0	FTM1_QD_PHA	
65	J9	PTA13/LLWU_P4	CMP2_IN1	CMP2_IN1	PTA13/LLWU_P4	CAN0_RX	FTM1_CH1			I2S0_TX_FS	FTM1_QD_PHB	
66	L10	PTA14	CMP3_IN0	CMP3_IN0	PTA14	SPI0_PCS0	UART0_TX			I2S0_RX_BCLK	I2S0_TXD1	
67	L11	PTA15	CMP3_IN1	CMP3_IN1	PTA15	SPI0_SCK	UART0_RX			I2S0_RXD0		
68	K10	PTA16	CMP3_IN2	CMP3_IN2	PTA16	SPI0_SOUT	UART0_CTS_b/UART0_COL_b			I2S0_RX_FS	I2S0_RXD1	
69	K11	PTA17	ADC1_SE17	ADC1_SE17	PTA17	SPI0_SIN	UART0_RTS_b			I2S0_MCLK		
70	E8	VDD	VDD	VDD								
71	G8	VSS	VSS	VSS								
72	M12	PTA18	EXTAL0	EXTAL0	PTA18		FTM0_FLT2	FTM_CLKIN0				
73	M11	PTA19	XTAL0	XTAL0	PTA19		FTM1_FLT0	FTM_CLKIN1		LPTMR0_ALT1		
74	L12	RESET_b	RESET_b	RESET_b								
75	K12	PTA24	CMP3_IN4	CMP3_IN4	PTA24	ULPI_DATA2				FB_A29		
76	J12	PTA25	CMP3_IN5	CMP3_IN5	PTA25	ULPI_DATA3				FB_A28		
77	J11	PTA26	ADC2_SE15	ADC2_SE15	PTA26	ULPI_DATA4				FB_A27		
78	J10	PTA27	ADC2_SE14	ADC2_SE14	PTA27	ULPI_DATA5				FB_A26		
79	H12	PTA28	ADC2_SE13	ADC2_SE13	PTA28	ULPI_DATA6				FB_A25		
80	H11	PTA29	ADC2_SE12	ADC2_SE12	PTA29	ULPI_DATA7				FB_A24		
81	H10	PTB0/LLWU_P5	ADC0_SE8/ADC1_SE8/ADC2_SE8/ADC3_SE8/TSI0_CH0	ADC0_SE8/ADC1_SE8/ADC2_SE8/ADC3_SE8/TSI0_CH0	PTB0/LLWU_P5	I2C0_SCL	FTM1_CH0			FTM1_QD_PHA		
82	H9	PTB1	ADC0_SE9/ADC1_SE9/ADC2_SE9/ADC3_SE9/TSI0_CH6	ADC0_SE9/ADC1_SE9/ADC2_SE9/ADC3_SE9/TSI0_CH6	PTB1	I2C0_SDA	FTM1_CH1			FTM1_QD_PHB		

144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
83	G12	PTB2	ADC0_SE12/TSI0_CH7	ADC0_SE12/TSI0_CH7	PTB2	I2C0_SCL	UART0_RTS_b			FTM0_FLT3		
84	G11	PTB3	ADC0_SE13/TSI0_CH8	ADC0_SE13/TSI0_CH8	PTB3	I2C0_SDA	UART0_CTS_b/ UART0_COL_b			FTM0_FLT0		
85	G10	PTB4	ADC1_SE10	ADC1_SE10	PTB4					FTM1_FLT0		
86	G9	PTB5	ADC1_SE11	ADC1_SE11	PTB5					FTM2_FLT0		
87	F12	PTB6	ADC1_SE12	ADC1_SE12	PTB6				FB_AD23			
88	F11	PTB7	ADC1_SE13	ADC1_SE13	PTB7				FB_AD22			
89	F10	PTB8	DISABLED		PTB8		UART3_RTS_b		FB_AD21			
90	F9	PTB9	DISABLED		PTB9	SPI1_PCS1	UART3_CTS_b		FB_AD20			
91	E12	PTB10	ADC1_SE14	ADC1_SE14	PTB10	SPI1_PCS0	UART3_RX	I2S1_TX_BCLK	FB_AD19	FTM0_FLT1		
92	E11	PTB11	ADC1_SE15	ADC1_SE15	PTB11	SPI1_SCK	UART3_TX	I2S1_TX_FS	FB_AD18	FTM0_FLT2		
93	H7	VSS	VSS	VSS								
94	F5	VDD	VDD	VDD								
95	E10	PTB16	TSI0_CH9	TSI0_CH9	PTB16	SPI1_SOUT	UART0_RX	I2S1_TXD0	FB_AD17	EWM_IN		
96	E9	PTB17	TSI0_CH10	TSI0_CH10	PTB17	SPI1_SIN	UART0_TX	I2S1_TXD1	FB_AD16	EWM_OUT_b		
97	D12	PTB18	TSI0_CH11	TSI0_CH11	PTB18	CAN0_TX	FTM2_CH0	I2S0_TX_BCLK	FB_AD15	FTM2_QD_PHA		
98	D11	PTB19	TSI0_CH12	TSI0_CH12	PTB19	CAN0_RX	FTM2_CH1	I2S0_TX_FS	FB_OE_b	FTM2_QD_PHB		
99	D10	PTB20	ADC2_SE4a	ADC2_SE4a	PTB20	SPI2_PCS0			FB_AD31/ NFC_DATA15	CMP0_OUT		
100	D9	PTB21	ADC2_SE5a	ADC2_SE5a	PTB21	SPI2_SCK			FB_AD30/ NFC_DATA14	CMP1_OUT		
101	C12	PTB22	DISABLED		PTB22	SPI2_SOUT			FB_AD29/ NFC_DATA13	CMP2_OUT		
102	C11	PTB23	DISABLED		PTB23	SPI2_SIN	SPI0_PCS5		FB_AD28/ NFC_DATA12	CMP3_OUT		
103	B12	PTC0	ADC0_SE14/ TSI0_CH13	ADC0_SE14/ TSI0_CH13	PTC0	SPI0_PCS4	PDB0_EXTRG		FB_AD14/ NFC_DATA11	I2S0_TXD1		

144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
104	B11	PTC1/LLWU_P6	ADC0_SE15/TSI0_CH14	ADC0_SE15/TSI0_CH14	PTC1/LLWU_P6	SPI0_PCS3	UART1_RTS_b	FTM0_CH0	FB_AD13/NFC_DATA10	I2S0_TXD0		
105	A12	PTC2	ADC0_SE4b/CMP1_IN0/TSI0_CH15	ADC0_SE4b/CMP1_IN0/TSI0_CH15	PTC2	SPI0_PCS2	UART1_CTS_b	FTM0_CH1	FB_AD12/NFC_DATA9	I2S0_TX_FS		
106	A11	PTC3/LLWU_P7	CMP1_IN1	CMP1_IN1	PTC3/LLWU_P7	SPI0_PCS1	UART1_RX	FTM0_CH2		I2S0_TX_BCLK		
107	H8	VSS	VSS	VSS								
108	—	VDD	VDD	VDD								
109	A9	PTC4/LLWU_P8	DISABLED		PTC4/LLWU_P8	SPI0_PCS0	UART1_TX	FTM0_CH3	FB_AD11/NFC_DATA8	CMP1_OUT	I2S1_TX_BCLK	
110	D8	PTC5/LLWU_P9	DISABLED		PTC5/LLWU_P9	SPI0_SCK	LPTMR0_ALT2	I2S0_RXD0	FB_AD10/NFC_DATA7	CMP0_OUT	I2S1_TX_FS	
111	C8	PTC6/LLWU_P10	CMP0_IN0	CMP0_IN0	PTC6/LLWU_P10	SPI0_SOUT	PDB0_EXTRG	I2S0_RX_BCLK	FB_AD9/NFC_DATA6	I2S0_MCLK		
112	B8	PTC7	CMP0_IN1	CMP0_IN1	PTC7	SPI0_SIN	USB_SOF_OUT	I2S0_RX_FS	FB_AD8/NFC_DATA5			
113	A8	PTC8	ADC1_SE4b/CMP0_IN2	ADC1_SE4b/CMP0_IN2	PTC8		FTM3_CH4	I2S0_MCLK	FB_AD7/NFC_DATA4			
114	D7	PTC9	ADC1_SE5b/CMP0_IN3	ADC1_SE5b/CMP0_IN3	PTC9		FTM3_CH5	I2S0_RX_BCLK	FB_AD6/NFC_DATA3	FTM2_FLT0		
115	C7	PTC10	ADC1_SE6b	ADC1_SE6b	PTC10	I2C1_SCL	FTM3_CH6	I2S0_RX_FS	FB_AD5/NFC_DATA2	I2S1_MCLK		
116	B7	PTC11/LLWU_P11	ADC1_SE7b	ADC1_SE7b	PTC11/LLWU_P11	I2C1_SDA	FTM3_CH7	I2S0_RXD1	FB_RW_b/NFC_WE			
117	A7	PTC12	DISABLED		PTC12		UART4_RTS_b		FB_AD27	FTM3_FLT0		
118	D6	PTC13	DISABLED		PTC13		UART4_CTS_b		FB_AD26			
119	C6	PTC14	DISABLED		PTC14		UART4_RX		FB_AD25			
120	B6	PTC15	DISABLED		PTC15		UART4_TX		FB_AD24			
121	—	VSS	VSS	VSS								
122	—	VDD	VDD	VDD								
123	A6	PTC16	DISABLED		PTC16	CAN1_RX	UART3_RX		FB_CS5_b/FB_TSIZ1/FB_BE23_16_BLS15_8_b	NFC_RB		
124	D5	PTC17	DISABLED		PTC17	CAN1_TX	UART3_TX		FB_CS4_b/FB_TSIZ0/	NFC_CE0_b		

144 LQFP	144 MAP BGA	Pin Name	Default	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	EzPort
									FB_BE31_24_BLS7_0_b			
125	C5	PTC18	DISABLED		PTC18		UART3_RTS_b		FB_TBST_b/FB_CS2_b/FB_BE15_8_BLS23_16_b	NFC_CE1_b		
126	B5	PTC19	DISABLED		PTC19		UART3_CTS_b		FB_CS3_b/FB_BE7_0_BLS31_24_b	FB_TA_b		
127	A5	PTD0/LLWU_P12	DISABLED		PTD0/LLWU_P12	SPI0_PCS0	UART2_RTS_b	FTM3_CH0	FB_ALE/FB_CS1_b/FB_TS_b	I2S1_RXD1		
128	D4	PTD1	ADC0_SE5b	ADC0_SE5b	PTD1	SPI0_SCK	UART2_CTS_b	FTM3_CH1	FB_CS0_b	I2S1_RXD0		
129	C4	PTD2/LLWU_P13	DISABLED		PTD2/LLWU_P13	SPI0_SOUT	UART2_RX	FTM3_CH2	FB_AD4	I2S1_RX_FS		
130	B4	PTD3	DISABLED		PTD3	SPI0_SIN	UART2_TX	FTM3_CH3	FB_AD3	I2S1_RX_BCLK		
131	A4	PTD4/LLWU_P14	DISABLED		PTD4/LLWU_P14	SPI0_PCS1	UART0_RTS_b	FTM0_CH4	FB_AD2/NFC_DATA1	EWM_IN		
132	A3	PTD5	ADC0_SE6b	ADC0_SE6b	PTD5	SPI0_PCS2	UART0_CTS_b/UART0_COL_b	FTM0_CH5	FB_AD1/NFC_DATA0	EWM_OUT_b		
133	A2	PTD6/LLWU_P15	ADC0_SE7b	ADC0_SE7b	PTD6/LLWU_P15	SPI0_PCS3	UART0_RX	FTM0_CH6	FB_AD0	FTM0_FLT0		
134	M10	VSS	VSS	VSS								
135	F8	VDD	VDD	VDD								
136	A1	PTD7	DISABLED		PTD7	CMT_IRO	UART0_TX	FTM0_CH7		FTM0_FLT1		
137	C9	PTD8	DISABLED		PTD8	I2C0_SCL	UART5_RX			FB_A16/NFC_CLE		
138	B9	PTD9	DISABLED		PTD9	I2C0_SDA	UART5_TX			FB_A17/NFC_ALE		
139	B3	PTD10	DISABLED		PTD10		UART5_RTS_b			FB_A18/NFC_RE		
140	B2	PTD11	DISABLED		PTD11	SPI2_PCS0	UART5_CTS_b	SDHC0_CLKIN		FB_A19		
141	B1	PTD12	DISABLED		PTD12	SPI2_SCK	FTM3_FLT0	SDHC0_D4		FB_A20		
142	C3	PTD13	DISABLED		PTD13	SPI2_SOUT		SDHC0_D5		FB_A21		
143	C2	PTD14	DISABLED		PTD14	SPI2_SIN		SDHC0_D6		FB_A22		
144	C1	PTD15	DISABLED		PTD15	SPI2_PCS1		SDHC0_D7		FB_A23		

10.3.2 K20 Pinouts

The below figure shows the pinout diagram for the devices supported by this document. Many signals may be multiplexed onto a single pin. To determine what signals can be used on which pin, see the previous section.

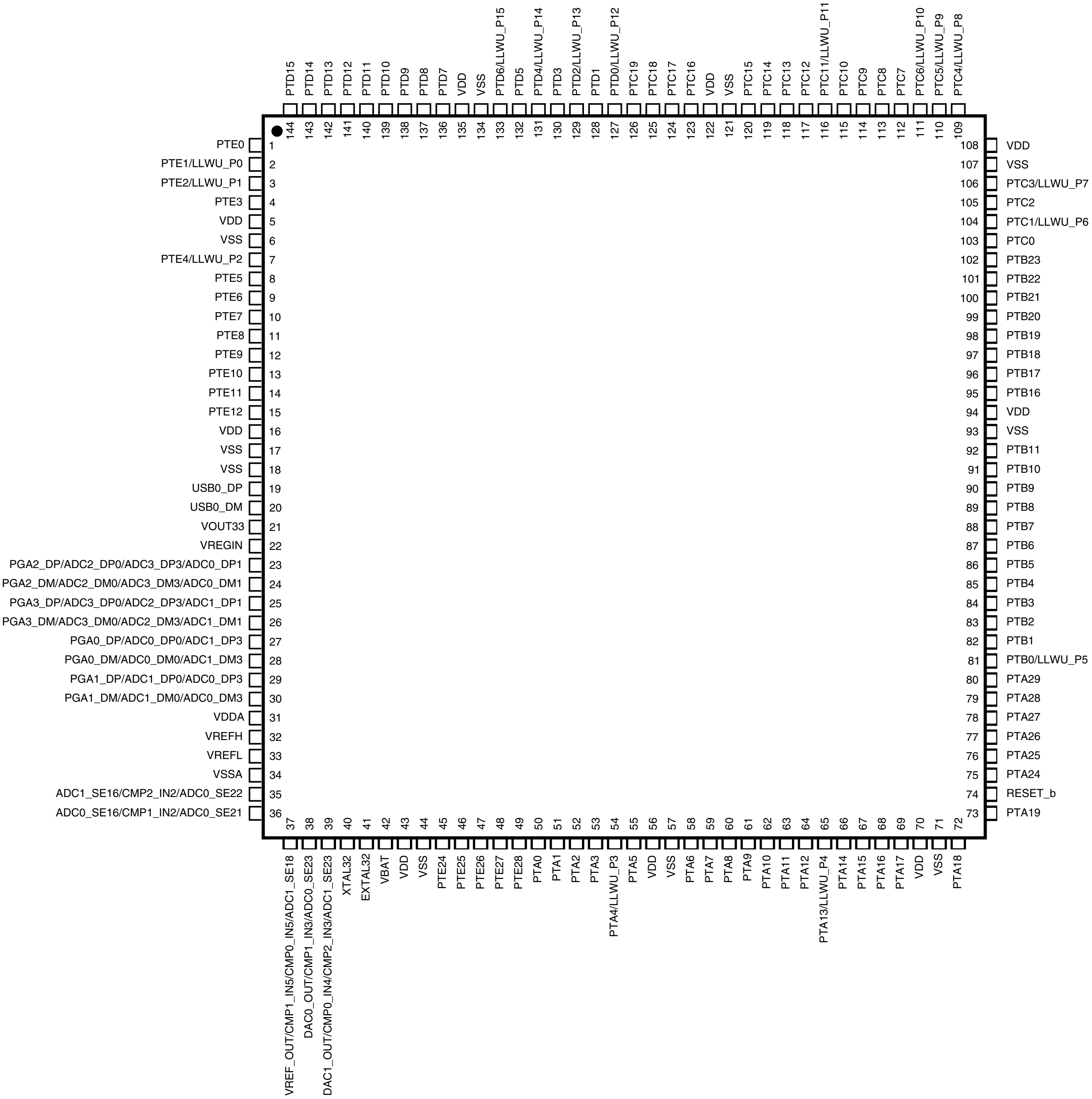


Figure 10-2. K20 144 LQFP Pinout Diagram

	1	2	3	4	5	6	7	8	9	10	11	12	
A	PTD7	PTD6/ LLWU_P15	PTD5	PTD4/ LLWU_P14	PTD0/ LLWU_P12	PTC16	PTC12	PTC8	PTC4/ LLWU_P8	NC	PTC3/ LLWU_P7	PTC2	A
B	PTD12	PTD11	PTD10	PTD3	PTC19	PTC15	PTC11/ LLWU_P11	PTC7	PTD9	NC	PTC1/ LLWU_P6	PTC0	B
C	PTD15	PTD14	PTD13	PTD2/ LLWU_P13	PTC18	PTC14	PTC10	PTC6/ LLWU_P10	PTD8	NC	PTB23	PTB22	C
D	PTE2/ LLWU_P1	PTE1/ LLWU_P0	PTE0	PTD1	PTC17	PTC13	PTC9	PTC5/ LLWU_P9	PTB21	PTB20	PTB19	PTB18	D
E	PTE6	PTE5	PTE4/ LLWU_P2	PTE3	VDD	VDD	VDD	VDD	PTB17	PTB16	PTB11	PTB10	E
F	PTE10	PTE9	PTE8	PTE7	VDD	VSS	VSS	VDD	PTB9	PTB8	PTB7	PTB6	F
G	VOUT33	VREGIN	PTE12	PTE11	VREFH	VREFL	VSS	VSS	PTB5	PTB4	PTB3	PTB2	G
H	USB0_DP	USB0_DM	VSS	PTE28	VDDA	VSSA	VSS	VSS	PTB1	PTB0/ LLWU_P5	PTA29	PTA28	H
J	PGA2_DP/ ADC2_DP0/ ADC3_DP3/ ADC0_DP1	PGA2_DM/ ADC2_DM0/ ADC3_DM3/ ADC0_DM1	ADC0_SE16/ CMP1_IN2/ ADC0_SE21	PTE27	PTA0	PTA1	PTA6	PTA7	PTA13/ LLWU_P4	PTA27	PTA26	PTA25	J
K	PGA3_DP/ ADC3_DP0/ ADC2_DP3/ ADC1_DP1	PGA3_DM/ ADC3_DM0/ ADC2_DM3/ ADC1_DM1	ADC1_SE16/ CMP2_IN2/ ADC0_SE22	PTE26	PTE25	PTA2	PTA3	PTA8	PTA12	PTA16	PTA17	PTA24	K
L	PGA0_DP/ ADC0_DP0/ ADC1_DP3	PGA0_DM/ ADC0_DM0/ ADC1_DM3	DAC0_OUT/ CMP1_IN3/ ADC0_SE23	DAC1_OUT/ CMP0_IN4/ CMP2_IN3/ ADC1_SE23	RTC_ WAKEUP_B	VBAT	PTA4/ LLWU_P3	PTA9	PTA11	PTA14	PTA15	RESET_b	L
M	PGA1_DP/ ADC1_DP0/ ADC0_DP3	PGA1_DM/ ADC1_DM0/ ADC0_DM3	VREF_OUT/ CMP1_IN5/ CMP0_IN5/ ADC1_SE18	PTE24	NC	EXTAL32	XTAL32	PTA5	PTA10	VSS	PTA19	PTA18	M
	1	2	3	4	5	6	7	8	9	10	11	12	

Figure 10-3. K20 144 MAPBGA Pinout Diagram

10.4 Module Signal Description Tables

The following sections correlate the chip-level signal name with the signal name used in the module's chapter. They also briefly describe the signal function and direction.

10.4.1 Core Modules

Table 10-2. JTAG Signal Descriptions

Chip signal name	Module signal name	Description	I/O
JTAG_TMS	JTAG_TMS/ SWD_DIO	JTAG Test Mode Selection	I/O
JTAG_TCLK	JTAG_TCLK/ SWD_CLK	JTAG Test Clock	I
JTAG_TDI	JTAG_TDI	JTAG Test Data Input	I
JTAG_TDO	JTAG_TDO/ TRACE_SWO	JTAG Test Data Output	O
JTAG_TRST	JTAG_TRST_b	JTAG Reset	I

Table 10-3. SWD Signal Descriptions

Chip signal name	Module signal name	Description	I/O
SWD_DIO	JTAG_TMS/ SWD_DIO	Serial Wire Data	I/O
SWD_CLK	JTAG_TCLK/ SWD_CLK	Serial Wire Clock	I

Table 10-4. TPIU Signal Descriptions

Chip signal name	Module signal name	Description	I/O
TRACE_CLKOUT	TRACECLK	Trace clock output from the ARM CoreSight debug block	O
TRACE_D[3:2]	TRACEDATA	Trace output data from the ARM CoreSight debug block used for 5-pin interface	O
TRACE_D[1:0]	TRACEDATA	Trace output data from the ARM CoreSight debug block used for both 5-pin and 3-pin interfaces	O
TRACE_SWO	JTAG_TDO/ TRACE_SWO	Trace output data from the ARM CoreSight debug block over a single pin	O

10.4.2 System Modules

Table 10-5. System Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{NMI}}$	—	Non-maskable interrupt **NOTE:** Driving the $\overline{\text{NMI}}$ signal low forces a non-maskable interrupt, if the $\overline{\text{NMI}}$ function is selected on the corresponding pin.	I

Table continues on the next page...

Table 10-5. System Signal Descriptions (continued)

Chip signal name	Module signal name	Description	I/O
$\overline{\text{RESET}}$	—	Reset input signal	I
VDD	—	MCU power	I
VSS	—	MCU ground	I

Table 10-6. EWM Signal Descriptions

Chip signal name	Module signal name	Description	I/O
EWM_IN	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
$\overline{\text{EWM\_OUT}}$	$\overline{\text{EWM\_out}}$	EWM reset out signal	O

10.4.3 Clock Modules

Table 10-7. OSC Signal Descriptions

Chip signal name	Module signal name	Description	I/O
EXTAL0	EXTAL	External clock/Oscillator input	I
XTAL0	XTAL	Oscillator output	O

Table 10-8. RTC OSC Signal Descriptions

Chip signal name	Module signal name	Description	I/O
EXTAL32	EXTAL32	32.768 kHz oscillator input	I
XTAL32	XTAL32	32.768 kHz oscillator output	O

10.4.4 Memories and Memory Interfaces

Table 10-9. EzPort Signal Descriptions

Chip signal name	Module signal name	Description	I/O
EZP_CLK	EZP_CK	EzPort Clock	Input
$\overline{\text{EZP\_CS}}$	$\overline{\text{EZP\_CS}}$	EzPort Chip Select	Input
EZP_DI	EZP_D	EzPort Serial Data In	Input
EZP_DO	EZP_Q	EzPort Serial Data Out	Output

Table 10-10. FlexBus Signal Descriptions

Chip signal name	Module signal name	Description	I/O
FB_CLKOUT	FB_CLK	FlexBus clock output	O
FB_A[29:16]	FB_A[29:16]	In a non-multiplexed configuration, this is the address bus.	O
FB_AD[31:0]	FB_D[31:0]/ FB_AD[31:0]	In a non-multiplexed configuration, this is the data bus. In a multiplexed configuration this bus is the address/data bus, FB_AD[31:0]. In non-multiplexed and multiplexed configurations, during the first cycle, this bus drives the upper address byte, addr[31:24].	I/O
$\overline{\mathrm{FB\_CS}}$[5:0]	$\overline{\mathrm{FB\_CS}}$[5:0]	General purpose chip-selects. The actual number of chip selects available depends upon the device and its pin configuration.	O
$\overline{\mathrm{FB\_BE31\_24\_BLS7\_0}}$, $\overline{\mathrm{FB\_BE23\_16\_BLS15\_8}}$, $\overline{\mathrm{FB\_BE15\_8\_BLS23\_16}}$, $\overline{\mathrm{FB\_BE7\_0\_BLS31\_24}}$	$\overline{\mathrm{FB\_BE\_31\_24}}$ $\overline{\mathrm{FB\_BE\_23\_16}}$ $\overline{\mathrm{FB\_BE\_15\_8}}$ $\overline{\mathrm{FB\_BE\_7\_0}}$	Byte enables	O
$\overline{\mathrm{FB\_OE}}$	$\overline{\mathrm{FB\_OE}}$	Output enable	O
FB_R $\overline{\mathrm{W}}$	FB_R/$\overline{\mathrm{W}}$	Read/write. 1 = Read, 0 = Write	O
$\overline{\mathrm{FB\_TS}}$/ FB_ALE	$\overline{\mathrm{FB\_TS}}$	Transfer start	O
FB_TSIZ[1:0]	FB_TSIZ[1:0]	Transfer size	O
$\overline{\mathrm{FB\_TA}}$	$\overline{\mathrm{FB\_TA}}$	Transfer acknowledge	I
$\overline{\mathrm{FB\_TBST}}$	$\overline{\mathrm{FB\_TBST}}$	Burst transfer indicator	O

10.4.5 Analog

Table 10-11. ADC 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
ADC0_DP3, PGA0_DP, ADC0_DP[1:0]	DADP[3:0]	Differential analog channel inputs	I
ADC0_DM3, PGA0_DM, ADC0_DM[1:0]	DADM[3:0]	Differential analog channel inputs	I
ADC0_SE[18:4]	AD[23:4]	Single-ended analog channel inputs	I
VREFH	V_{REFSH}	Voltage reference select high	I
VREFL	V_{REFSL}	Voltage reference select low	I
VDDA	V_{DDA}	Analog power supply	I

Table continues on the next page...

Table 10-11. ADC 0 Signal Descriptions (continued)

Chip signal name	Module signal name	Description	I/O
VSSA	V_{SSA}	Analog ground	I

Table 10-12. CMP 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
CMP0_IN[5:0]	IN[5:0]	Analog voltage inputs	I
CMP0_OUT	CMPO	Comparator output	O

Table 10-13. CMP 1 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
CMP1_IN[5:0]	IN[5:0]	Analog voltage inputs	I
CMP1_OUT	CMPO	Comparator output	O

Table 10-14. DAC 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
DAC0_OUT	—	DAC output	O

Table 10-15. DAC 1 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
DAC1_OUT	—	DAC output	O

Table 10-16. VREF Signal Descriptions

Chip signal name	Module signal name	Description	I/O
VREF_OUT	VREF_OUT	Internally-generated Voltage Reference output	O

10.4.6 Communication Interfaces

Table 10-17. USB HS OTG Signal Descriptions

Chip signal name	Module signal name	Description	I/O
ULPI_DATA[7:0]	ULPI_DATA[7:0]	Data bit *n*	I/O

Table continues on the next page...

Table 10-17. USB HS OTG Signal Descriptions (continued)

Chip signal name	Module signal name	Description	I/O
ULPI_CLK	ULPI_CLK	60 MHz clock input from the ULPI transceiver	I
ULPI_DIR	ULPI_DIR	Controls data bus direction	I
ULPI_NXT	ULPI_NXT	Next data	I
ULPI_STP	ULPI_STP	Indicates the end of a transfer on the bus	O

Table 10-18. USB FS OTG Signal Descriptions

Chip signal name	Module signal name	Description	I/O
USB0_DM	usb_dm	USB D- analog data signal on the USB bus.	I/O
USB0_DP	usb_dp	USB D+ analog data signal on the USB bus.	I/O
USB_CLKIN	—	Alternate USB clock input	I

Table 10-19. USB VREG Signal Descriptions

Chip signal name	Module signal name	Description	I/O
VOUT33	reg33_out	Regulator output voltage	O
VREGIN	reg33_in	Unregulated power supply	I

Table 10-20. CAN 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
CAN0_RX	CAN Rx	CAN Receive Pin	Input
CAN0_TX	CAN Tx	CAN Transmit Pin	Output

Table 10-21. CAN 1 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
CAN1_RX	CAN Rx	CAN Receive Pin	Input
CAN1_TX	CAN Tx	CAN Transmit Pin	Output

Table 10-22. SPI 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
SPI0_PCS0	PCS0/$\overline{\text{SS}}$	Master mode: Peripheral Chip Select 0 output Slave mode: Slave Select input	I/O

Table continues on the next page...

Table 10-22. SPI 0 Signal Descriptions (continued)

Chip signal name	Module signal name	Description	I/O
SPI0_PCS[3:1]	PCS[3:1]	Master mode: Peripheral Chip Select 1 - 3 Slave mode: Unused	O
SPI0_PCS4	PCS4	Master mode: Peripheral Chip Select 4 Slave mode: Unused	O
SPI0_PCS5	PCS5/ $\overline{\text{PCSS}}$	Master mode: Peripheral Chip Select 5 / Peripheral Chip Select Strobe Slave mode: Unused	O
SPI0_SIN	SIN	Serial Data In	I
SPI0_SOUT	SOUT	Serial Data Out	O
SPI0_SCK	SCK	Master mode: Serial Clock (output) Slave mode: Serial Clock (input)	I/O

Table 10-23. SPI 2 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
SPI2_PCS0	PCS0/$\overline{\text{SS}}$	Master mode: Peripheral Chip Select 0 output Slave mode: Slave Select input	I/O
SPI2_PCS1	PCS[3:1]	Master mode: Peripheral Chip Select 1 - 3 Slave mode: Unused	O
SPI2_SIN	SIN	Serial Data In	I
SPI2_SOUT	SOUT	Serial Data Out	O
SPI2_SCK	SCK	Master mode: Serial Clock (output) Slave mode: Serial Clock (input)	I/O

Table 10-24. I^{2}C 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
I2C0_SCL	SCL	Bidirectional serial clock line of the I^{2}C system.	I/O
I2C0_SDA	SDA	Bidirectional serial data line of the I^{2}C system.	I/O

Table 10-25. I^{2}C 1 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
I2C1_SCL	SCL	Bidirectional serial clock line of the I^{2}C system.	I/O
I2C1_SDA	SDA	Bidirectional serial data line of the I^{2}C system.	I/O

Table 10-26. UART 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{UART0\_CTS}}$	$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{UART0\_RTS}}$	$\overline{\text{RTS}}$	Request to send	O
UART0_TX	TXD	Transmit data	O
UART0_RX	RXD	Receive data	I
$\overline{\text{UART0\_COL}}$	$\overline{\text{Collision}}$	Collision detect	I

Table 10-27. UART 1 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{UART1\_CTS}}$	$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{UART1\_RTS}}$	$\overline{\text{RTS}}$	Request to send	O
UART1_TX	TXD	Transmit data	O
UART1_RX	RXD	Receive data	I

Table 10-28. UART 2 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{UART2\_CTS}}$	$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{UART2\_RTS}}$	$\overline{\text{RTS}}$	Request to send	O
UART2_TX	TXD	Transmit data	O
UART2_RX	RXD	Receive data	I

Table 10-29. UART 3 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{UART3\_CTS}}$	$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{UART3\_RTS}}$	$\overline{\text{RTS}}$	Request to send	O
UART3_TX	TXD	Transmit data	O
UART3_RX	RXD	Receive data	I

Table 10-30. UART 4 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{UART4\_CTS}}$	$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{UART4\_RTS}}$	$\overline{\text{RTS}}$	Request to send	O
UART4_TX	TXD	Transmit data	O

Table continues on the next page...

Table 10-30. UART 4 Signal Descriptions (continued)

Chip signal name	Module signal name	Description	I/O
UART4_RX	RXD	Receive data	I

Table 10-31. UART 5 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
$\overline{\text{UART5\_CTS}}$	$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{UART5\_RTS}}$	$\overline{\text{RTS}}$	Request to send	O
UART5_TX	TXD	Transmit data	O
UART5_RX	RXD	Receive data	I

Table 10-32. SDHC Signal Descriptions

Chip signal name	Module signal name	Description	I/O
SDHC0_CLKIN	—	SDHC clock input	I
SDHC0_DCLK	SDHC_DCLK	Generated clock used to drive the MMC, SD, SDIO or CE-ATA cards.	O
SDHC0_CMD	SDHC_CMD	Send commands to and receive responses from the card.	I/O
SDHC0_D0	SDHC_D0	DAT0 line or busy-state detect	I/O
SDHC0_D1	SDHC_D1	8-bit mode: DAT1 line 4-bit mode: DAT1 line or interrupt detect 1-bit mode: Interrupt detect	I/O
SDHC0_D2	SDHC_D2	4-/8-bit mode: DAT2 line or read wait 1-bit mode: Read wait	I/O
SDHC0_D3	SDHC_D3	4-/8-bit mode: DAT3 line or configured as card detection pin 1-bit mode: May be configured as card detection pin	I/O
SDHC0_D4	SDHC_D4	DAT4 line in 8-bit mode Not used in other modes	I/O
SDHC0_D5	SDHC_D5	DAT5 line in 8-bit mode Not used in other modes	I/O
SDHC0_D6	SDHC_D6	DAT6 line in 8-bit mode Not used in other modes	I/O
SDHC0_D7	SDHC_D7	DAT7 line in 8-bit mode Not used in other modes	I/O

Table 10-33. I^{2}S0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
I2S0_MCLK	SAI_MCLK	Audio Master Clock	I/O
I2S0_RX_BCLK	SAI_RX_BCLK	Receive Bit Clock	I/O
I2S0_RX_FS	SAI_RX_SYNC	Receive Frame Sync	I/O
I2S0_RXD	SAI_RX_DATA[1:0]	Receive Data	I
I2S0_TX_BCLK	SAI_TX_BCLK	Transmit Bit Clock	I/O
I2S0_TX_FS	SAI_TX_SYNC	Transmit Frame Sync	I/O
I2S0_TXD	SAI_TX_DATA[1:0]	Transmit Data	O

Table 10-34. I^{2}S1 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
I2S1_MCLK	SAI_MCLK	Audio Master Clock	I/O
I2S1_RX_BCLK	SAI_RX_BCLK	Receive Bit Clock	I/O
I2S1_RX_FS	SAI_RX_SYNC	Receive Frame Sync	I/O
I2S1_RXD	SAI_RX_DATA[1:0]	Receive Data	I
I2S1_TX_BCLK	SAI_TX_BCLK	Transmit Bit Clock	I/O
I2S1_TX_FS	SAI_TX_SYNC	Transmit Frame Sync	I/O
I2S1_TXD	SAI_TX_DATA[1:0]	Transmit Data	O

10.4.7 Human-Machine Interfaces (HMI)

Table 10-35. GPIO Signal Descriptions

Chip signal name	Module signal name	Description	I/O
PTA[31:0][1]	PORTA[31:0]	General purpose input/output	I/O
PTB[31:0][1]	PORTB[31:0]	General purpose input/output	I/O
PTC[31:0][1]	PORTC[31:0]	General purpose input/output	I/O
PTD[31:0][1]	PORTD[31:0]	General purpose input/output	I/O
PTE[31:0][1]	PORTE[31:0]	General purpose input/output	I/O
PTF[31:0][1]	PORTF[31:0]	General purpose input/output	I/O

[1] The available GPIO pins depends on the specific package. See the signal multiplexing section for which exact GPIO signals are available.

Table 10-36. TSI 0 Signal Descriptions

Chip signal name	Module signal name	Description	I/O
TSI0_CH[15:0]	TSI_IN[15:0]	TSI pins. Switchable driver that connects directly to the electrode pins TSI[15:0] can operate as GPIO pins	I/O

Chapter 11
Port control and interrupts (PORT)

11.1 Introduction

NOTE

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

11.1.1 Overview

The port control and interrupt (PORT) module provides support for external interrupt, digital filtering and port control 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.

11.1.2 Features

- Pin interrupt
  - Interrupt flag and enable registers for each pin
  - Supports edge sensitive (rising, falling, both) or level sensitive (low, high) configured per pin
  - Support for interrupt or DMA request configured per pin
  - Asynchronous wakeup in low-power modes
  - Pin interrupt is functional in all digital pin muxing modes
- Digital input filter
  - Digital input filter for each pin, usable by any digital peripheral muxed onto pin
  - Individual enable or bypass control bit per pin

  - Selectable clock source for digital input filter with 5-bit resolution on filter size
  - Digital filter is functional in all digital pin muxing modes
- Port control
  - Individual pull control registers with pullup, pulldown and pull-disable support
  - Individual drive strength register supporting high and low drive strength
  - Individual slew rate register supporting fast and slow slew rates
  - Individual input passive filter register supporting enabled and disabled
  - Individual open-drain register supporting enabled and disabled
  - Individual mux control register supporting analog (or pin disabled), GPIO plus up to six chip specific digital functions
  - Pad configuration registers are functional in all digital pin muxing modes

11.1.3 Modes of operation

11.1.3.1 Run mode

In run mode, the PORT operates normally.

11.1.3.2 Wait mode

In wait mode, the 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 wait mode, but do not cause an exit from the low power mode.

11.1.3.3 Stop mode

In stop mode, the digital input filters are bypassed unless they are configured to run from the 1 kHz LPO clock source. The PORT can be configured to exit the low power mode via an asynchronous wakeup signal if an enabled interrupt (but not DMA request) is detected.

11.1.3.4 Debug mode

In debug mode, the PORTx operates normally.

11.2 External signal description

Table 11-1. 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.

11.3 Detailed signal descriptions

Table 11-2. PORTx interface-detailed signal descriptions

Signal	I/O	Description	
PORTx[31:0]	I/O	External interrupt.	
		State meaning	Asserted-pin is logic one. Negated-pin is logic zero.
		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.

11.4 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.

PORT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_9000	Pin Control Register n (PORTA_PCR0)	32	R/W	0000_0000h	11.4.1/281
4004_9004	Pin Control Register n (PORTA_PCR1)	32	R/W	0000_0000h	11.4.1/281
4004_9008	Pin Control Register n (PORTA_PCR2)	32	R/W	0000_0000h	11.4.1/281

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_900C	Pin Control Register n (PORTA_PCR3)	32	R/W	0000_0000h	11.4.1/281
4004_9010	Pin Control Register n (PORTA_PCR4)	32	R/W	0000_0000h	11.4.1/281
4004_9014	Pin Control Register n (PORTA_PCR5)	32	R/W	0000_0000h	11.4.1/281
4004_9018	Pin Control Register n (PORTA_PCR6)	32	R/W	0000_0000h	11.4.1/281
4004_901C	Pin Control Register n (PORTA_PCR7)	32	R/W	0000_0000h	11.4.1/281
4004_9020	Pin Control Register n (PORTA_PCR8)	32	R/W	0000_0000h	11.4.1/281
4004_9024	Pin Control Register n (PORTA_PCR9)	32	R/W	0000_0000h	11.4.1/281
4004_9028	Pin Control Register n (PORTA_PCR10)	32	R/W	0000_0000h	11.4.1/281
4004_902C	Pin Control Register n (PORTA_PCR11)	32	R/W	0000_0000h	11.4.1/281
4004_9030	Pin Control Register n (PORTA_PCR12)	32	R/W	0000_0000h	11.4.1/281
4004_9034	Pin Control Register n (PORTA_PCR13)	32	R/W	0000_0000h	11.4.1/281
4004_9038	Pin Control Register n (PORTA_PCR14)	32	R/W	0000_0000h	11.4.1/281
4004_903C	Pin Control Register n (PORTA_PCR15)	32	R/W	0000_0000h	11.4.1/281
4004_9040	Pin Control Register n (PORTA_PCR16)	32	R/W	0000_0000h	11.4.1/281
4004_9044	Pin Control Register n (PORTA_PCR17)	32	R/W	0000_0000h	11.4.1/281
4004_9048	Pin Control Register n (PORTA_PCR18)	32	R/W	0000_0000h	11.4.1/281
4004_904C	Pin Control Register n (PORTA_PCR19)	32	R/W	0000_0000h	11.4.1/281
4004_9050	Pin Control Register n (PORTA_PCR20)	32	R/W	0000_0000h	11.4.1/281
4004_9054	Pin Control Register n (PORTA_PCR21)	32	R/W	0000_0000h	11.4.1/281
4004_9058	Pin Control Register n (PORTA_PCR22)	32	R/W	0000_0000h	11.4.1/281
4004_905C	Pin Control Register n (PORTA_PCR23)	32	R/W	0000_0000h	11.4.1/281
4004_9060	Pin Control Register n (PORTA_PCR24)	32	R/W	0000_0000h	11.4.1/281
4004_9064	Pin Control Register n (PORTA_PCR25)	32	R/W	0000_0000h	11.4.1/281
4004_9068	Pin Control Register n (PORTA_PCR26)	32	R/W	0000_0000h	11.4.1/281
4004_906C	Pin Control Register n (PORTA_PCR27)	32	R/W	0000_0000h	11.4.1/281
4004_9070	Pin Control Register n (PORTA_PCR28)	32	R/W	0000_0000h	11.4.1/281
4004_9074	Pin Control Register n (PORTA_PCR29)	32	R/W	0000_0000h	11.4.1/281
4004_9078	Pin Control Register n (PORTA_PCR30)	32	R/W	0000_0000h	11.4.1/281
4004_907C	Pin Control Register n (PORTA_PCR31)	32	R/W	0000_0000h	11.4.1/281
4004_9080	Global Pin Control Low Register (PORTA_GPCLR)	32	W (always reads zero)	0000_0000h	11.4.2/284
4004_9084	Global Pin Control High Register (PORTA_GPCHR)	32	W (always	0000_0000h	11.4.3/284

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
			reads zero)		
4004_90A0	Interrupt Status Flag Register (PORTA_ISFR)	32	w1c	0000_0000h	11.4.4/285
4004_90C0	Digital Filter Enable Register (PORTA_DFER)	32	R/W	0000_0000h	11.4.5/286
4004_90C4	Digital Filter Clock Register (PORTA_DFCR)	32	R/W	0000_0000h	11.4.6/286
4004_90C8	Digital Filter Width Register (PORTA_DFWR)	32	R/W	0000_0000h	11.4.7/287
4004_A000	Pin Control Register n (PORTB_PCR0)	32	R/W	0000_0000h	11.4.1/281
4004_A004	Pin Control Register n (PORTB_PCR1)	32	R/W	0000_0000h	11.4.1/281
4004_A008	Pin Control Register n (PORTB_PCR2)	32	R/W	0000_0000h	11.4.1/281
4004_A00C	Pin Control Register n (PORTB_PCR3)	32	R/W	0000_0000h	11.4.1/281
4004_A010	Pin Control Register n (PORTB_PCR4)	32	R/W	0000_0000h	11.4.1/281
4004_A014	Pin Control Register n (PORTB_PCR5)	32	R/W	0000_0000h	11.4.1/281
4004_A018	Pin Control Register n (PORTB_PCR6)	32	R/W	0000_0000h	11.4.1/281
4004_A01C	Pin Control Register n (PORTB_PCR7)	32	R/W	0000_0000h	11.4.1/281
4004_A020	Pin Control Register n (PORTB_PCR8)	32	R/W	0000_0000h	11.4.1/281
4004_A024	Pin Control Register n (PORTB_PCR9)	32	R/W	0000_0000h	11.4.1/281
4004_A028	Pin Control Register n (PORTB_PCR10)	32	R/W	0000_0000h	11.4.1/281
4004_A02C	Pin Control Register n (PORTB_PCR11)	32	R/W	0000_0000h	11.4.1/281
4004_A030	Pin Control Register n (PORTB_PCR12)	32	R/W	0000_0000h	11.4.1/281
4004_A034	Pin Control Register n (PORTB_PCR13)	32	R/W	0000_0000h	11.4.1/281
4004_A038	Pin Control Register n (PORTB_PCR14)	32	R/W	0000_0000h	11.4.1/281
4004_A03C	Pin Control Register n (PORTB_PCR15)	32	R/W	0000_0000h	11.4.1/281
4004_A040	Pin Control Register n (PORTB_PCR16)	32	R/W	0000_0000h	11.4.1/281
4004_A044	Pin Control Register n (PORTB_PCR17)	32	R/W	0000_0000h	11.4.1/281
4004_A048	Pin Control Register n (PORTB_PCR18)	32	R/W	0000_0000h	11.4.1/281
4004_A04C	Pin Control Register n (PORTB_PCR19)	32	R/W	0000_0000h	11.4.1/281
4004_A050	Pin Control Register n (PORTB_PCR20)	32	R/W	0000_0000h	11.4.1/281
4004_A054	Pin Control Register n (PORTB_PCR21)	32	R/W	0000_0000h	11.4.1/281
4004_A058	Pin Control Register n (PORTB_PCR22)	32	R/W	0000_0000h	11.4.1/281
4004_A05C	Pin Control Register n (PORTB_PCR23)	32	R/W	0000_0000h	11.4.1/281
4004_A060	Pin Control Register n (PORTB_PCR24)	32	R/W	0000_0000h	11.4.1/281
4004_A064	Pin Control Register n (PORTB_PCR25)	32	R/W	0000_0000h	11.4.1/281
4004_A068	Pin Control Register n (PORTB_PCR26)	32	R/W	0000_0000h	11.4.1/281
4004_A06C	Pin Control Register n (PORTB_PCR27)	32	R/W	0000_0000h	11.4.1/281

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_A070	Pin Control Register n (PORTB_PCR28)	32	R/W	0000_0000h	11.4.1/281
4004_A074	Pin Control Register n (PORTB_PCR29)	32	R/W	0000_0000h	11.4.1/281
4004_A078	Pin Control Register n (PORTB_PCR30)	32	R/W	0000_0000h	11.4.1/281
4004_A07C	Pin Control Register n (PORTB_PCR31)	32	R/W	0000_0000h	11.4.1/281
4004_A080	Global Pin Control Low Register (PORTB_GPCLR)	32	W (always reads zero)	0000_0000h	11.4.2/284
4004_A084	Global Pin Control High Register (PORTB_GPCHR)	32	W (always reads zero)	0000_0000h	11.4.3/284
4004_A0A0	Interrupt Status Flag Register (PORTB_ISFR)	32	w1c	0000_0000h	11.4.4/285
4004_A0C0	Digital Filter Enable Register (PORTB_DFER)	32	R/W	0000_0000h	11.4.5/286
4004_A0C4	Digital Filter Clock Register (PORTB_DFCR)	32	R/W	0000_0000h	11.4.6/286
4004_A0C8	Digital Filter Width Register (PORTB_DFWR)	32	R/W	0000_0000h	11.4.7/287
4004_B000	Pin Control Register n (PORTC_PCR0)	32	R/W	0000_0000h	11.4.1/281
4004_B004	Pin Control Register n (PORTC_PCR1)	32	R/W	0000_0000h	11.4.1/281
4004_B008	Pin Control Register n (PORTC_PCR2)	32	R/W	0000_0000h	11.4.1/281
4004_B00C	Pin Control Register n (PORTC_PCR3)	32	R/W	0000_0000h	11.4.1/281
4004_B010	Pin Control Register n (PORTC_PCR4)	32	R/W	0000_0000h	11.4.1/281
4004_B014	Pin Control Register n (PORTC_PCR5)	32	R/W	0000_0000h	11.4.1/281
4004_B018	Pin Control Register n (PORTC_PCR6)	32	R/W	0000_0000h	11.4.1/281
4004_B01C	Pin Control Register n (PORTC_PCR7)	32	R/W	0000_0000h	11.4.1/281
4004_B020	Pin Control Register n (PORTC_PCR8)	32	R/W	0000_0000h	11.4.1/281
4004_B024	Pin Control Register n (PORTC_PCR9)	32	R/W	0000_0000h	11.4.1/281
4004_B028	Pin Control Register n (PORTC_PCR10)	32	R/W	0000_0000h	11.4.1/281
4004_B02C	Pin Control Register n (PORTC_PCR11)	32	R/W	0000_0000h	11.4.1/281
4004_B030	Pin Control Register n (PORTC_PCR12)	32	R/W	0000_0000h	11.4.1/281
4004_B034	Pin Control Register n (PORTC_PCR13)	32	R/W	0000_0000h	11.4.1/281
4004_B038	Pin Control Register n (PORTC_PCR14)	32	R/W	0000_0000h	11.4.1/281
4004_B03C	Pin Control Register n (PORTC_PCR15)	32	R/W	0000_0000h	11.4.1/281
4004_B040	Pin Control Register n (PORTC_PCR16)	32	R/W	0000_0000h	11.4.1/281
4004_B044	Pin Control Register n (PORTC_PCR17)	32	R/W	0000_0000h	11.4.1/281
4004_B048	Pin Control Register n (PORTC_PCR18)	32	R/W	0000_0000h	11.4.1/281
4004_B04C	Pin Control Register n (PORTC_PCR19)	32	R/W	0000_0000h	11.4.1/281

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_B050	Pin Control Register n (PORTC_PCR20)	32	R/W	0000_0000h	11.4.1/281
4004_B054	Pin Control Register n (PORTC_PCR21)	32	R/W	0000_0000h	11.4.1/281
4004_B058	Pin Control Register n (PORTC_PCR22)	32	R/W	0000_0000h	11.4.1/281
4004_B05C	Pin Control Register n (PORTC_PCR23)	32	R/W	0000_0000h	11.4.1/281
4004_B060	Pin Control Register n (PORTC_PCR24)	32	R/W	0000_0000h	11.4.1/281
4004_B064	Pin Control Register n (PORTC_PCR25)	32	R/W	0000_0000h	11.4.1/281
4004_B068	Pin Control Register n (PORTC_PCR26)	32	R/W	0000_0000h	11.4.1/281
4004_B06C	Pin Control Register n (PORTC_PCR27)	32	R/W	0000_0000h	11.4.1/281
4004_B070	Pin Control Register n (PORTC_PCR28)	32	R/W	0000_0000h	11.4.1/281
4004_B074	Pin Control Register n (PORTC_PCR29)	32	R/W	0000_0000h	11.4.1/281
4004_B078	Pin Control Register n (PORTC_PCR30)	32	R/W	0000_0000h	11.4.1/281
4004_B07C	Pin Control Register n (PORTC_PCR31)	32	R/W	0000_0000h	11.4.1/281
4004_B080	Global Pin Control Low Register (PORTC_GPCLR)	32	W (always reads zero)	0000_0000h	11.4.2/284
4004_B084	Global Pin Control High Register (PORTC_GPCHR)	32	W (always reads zero)	0000_0000h	11.4.3/284
4004_B0A0	Interrupt Status Flag Register (PORTC_ISFR)	32	w1c	0000_0000h	11.4.4/285
4004_B0C0	Digital Filter Enable Register (PORTC_DFER)	32	R/W	0000_0000h	11.4.5/286
4004_B0C4	Digital Filter Clock Register (PORTC_DFCR)	32	R/W	0000_0000h	11.4.6/286
4004_B0C8	Digital Filter Width Register (PORTC_DFWR)	32	R/W	0000_0000h	11.4.7/287
4004_C000	Pin Control Register n (PORTD_PCR0)	32	R/W	0000_0000h	11.4.1/281
4004_C004	Pin Control Register n (PORTD_PCR1)	32	R/W	0000_0000h	11.4.1/281
4004_C008	Pin Control Register n (PORTD_PCR2)	32	R/W	0000_0000h	11.4.1/281
4004_C00C	Pin Control Register n (PORTD_PCR3)	32	R/W	0000_0000h	11.4.1/281
4004_C010	Pin Control Register n (PORTD_PCR4)	32	R/W	0000_0000h	11.4.1/281
4004_C014	Pin Control Register n (PORTD_PCR5)	32	R/W	0000_0000h	11.4.1/281
4004_C018	Pin Control Register n (PORTD_PCR6)	32	R/W	0000_0000h	11.4.1/281
4004_C01C	Pin Control Register n (PORTD_PCR7)	32	R/W	0000_0000h	11.4.1/281
4004_C020	Pin Control Register n (PORTD_PCR8)	32	R/W	0000_0000h	11.4.1/281
4004_C024	Pin Control Register n (PORTD_PCR9)	32	R/W	0000_0000h	11.4.1/281
4004_C028	Pin Control Register n (PORTD_PCR10)	32	R/W	0000_0000h	11.4.1/281
4004_C02C	Pin Control Register n (PORTD_PCR11)	32	R/W	0000_0000h	11.4.1/281

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_C030	Pin Control Register n (PORTD_PCR12)	32	R/W	0000_0000h	11.4.1/281
4004_C034	Pin Control Register n (PORTD_PCR13)	32	R/W	0000_0000h	11.4.1/281
4004_C038	Pin Control Register n (PORTD_PCR14)	32	R/W	0000_0000h	11.4.1/281
4004_C03C	Pin Control Register n (PORTD_PCR15)	32	R/W	0000_0000h	11.4.1/281
4004_C040	Pin Control Register n (PORTD_PCR16)	32	R/W	0000_0000h	11.4.1/281
4004_C044	Pin Control Register n (PORTD_PCR17)	32	R/W	0000_0000h	11.4.1/281
4004_C048	Pin Control Register n (PORTD_PCR18)	32	R/W	0000_0000h	11.4.1/281
4004_C04C	Pin Control Register n (PORTD_PCR19)	32	R/W	0000_0000h	11.4.1/281
4004_C050	Pin Control Register n (PORTD_PCR20)	32	R/W	0000_0000h	11.4.1/281
4004_C054	Pin Control Register n (PORTD_PCR21)	32	R/W	0000_0000h	11.4.1/281
4004_C058	Pin Control Register n (PORTD_PCR22)	32	R/W	0000_0000h	11.4.1/281
4004_C05C	Pin Control Register n (PORTD_PCR23)	32	R/W	0000_0000h	11.4.1/281
4004_C060	Pin Control Register n (PORTD_PCR24)	32	R/W	0000_0000h	11.4.1/281
4004_C064	Pin Control Register n (PORTD_PCR25)	32	R/W	0000_0000h	11.4.1/281
4004_C068	Pin Control Register n (PORTD_PCR26)	32	R/W	0000_0000h	11.4.1/281
4004_C06C	Pin Control Register n (PORTD_PCR27)	32	R/W	0000_0000h	11.4.1/281
4004_C070	Pin Control Register n (PORTD_PCR28)	32	R/W	0000_0000h	11.4.1/281
4004_C074	Pin Control Register n (PORTD_PCR29)	32	R/W	0000_0000h	11.4.1/281
4004_C078	Pin Control Register n (PORTD_PCR30)	32	R/W	0000_0000h	11.4.1/281
4004_C07C	Pin Control Register n (PORTD_PCR31)	32	R/W	0000_0000h	11.4.1/281
4004_C080	Global Pin Control Low Register (PORTD_GPCLR)	32	W (always reads zero)	0000_0000h	11.4.2/284
4004_C084	Global Pin Control High Register (PORTD_GPCHR)	32	W (always reads zero)	0000_0000h	11.4.3/284
4004_C0A0	Interrupt Status Flag Register (PORTD_ISFR)	32	w1c	0000_0000h	11.4.4/285
4004_C0C0	Digital Filter Enable Register (PORTD_DFER)	32	R/W	0000_0000h	11.4.5/286
4004_C0C4	Digital Filter Clock Register (PORTD_DFCR)	32	R/W	0000_0000h	11.4.6/286
4004_C0C8	Digital Filter Width Register (PORTD_DFWR)	32	R/W	0000_0000h	11.4.7/287
4004_D000	Pin Control Register n (PORTE_PCR0)	32	R/W	0000_0000h	11.4.1/281
4004_D004	Pin Control Register n (PORTE_PCR1)	32	R/W	0000_0000h	11.4.1/281
4004_D008	Pin Control Register n (PORTE_PCR2)	32	R/W	0000_0000h	11.4.1/281
4004_D00C	Pin Control Register n (PORTE_PCR3)	32	R/W	0000_0000h	11.4.1/281

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_D010	Pin Control Register n (PORTE_PCR4)	32	R/W	0000_0000h	11.4.1/281
4004_D014	Pin Control Register n (PORTE_PCR5)	32	R/W	0000_0000h	11.4.1/281
4004_D018	Pin Control Register n (PORTE_PCR6)	32	R/W	0000_0000h	11.4.1/281
4004_D01C	Pin Control Register n (PORTE_PCR7)	32	R/W	0000_0000h	11.4.1/281
4004_D020	Pin Control Register n (PORTE_PCR8)	32	R/W	0000_0000h	11.4.1/281
4004_D024	Pin Control Register n (PORTE_PCR9)	32	R/W	0000_0000h	11.4.1/281
4004_D028	Pin Control Register n (PORTE_PCR10)	32	R/W	0000_0000h	11.4.1/281
4004_D02C	Pin Control Register n (PORTE_PCR11)	32	R/W	0000_0000h	11.4.1/281
4004_D030	Pin Control Register n (PORTE_PCR12)	32	R/W	0000_0000h	11.4.1/281
4004_D034	Pin Control Register n (PORTE_PCR13)	32	R/W	0000_0000h	11.4.1/281
4004_D038	Pin Control Register n (PORTE_PCR14)	32	R/W	0000_0000h	11.4.1/281
4004_D03C	Pin Control Register n (PORTE_PCR15)	32	R/W	0000_0000h	11.4.1/281
4004_D040	Pin Control Register n (PORTE_PCR16)	32	R/W	0000_0000h	11.4.1/281
4004_D044	Pin Control Register n (PORTE_PCR17)	32	R/W	0000_0000h	11.4.1/281
4004_D048	Pin Control Register n (PORTE_PCR18)	32	R/W	0000_0000h	11.4.1/281
4004_D04C	Pin Control Register n (PORTE_PCR19)	32	R/W	0000_0000h	11.4.1/281
4004_D050	Pin Control Register n (PORTE_PCR20)	32	R/W	0000_0000h	11.4.1/281
4004_D054	Pin Control Register n (PORTE_PCR21)	32	R/W	0000_0000h	11.4.1/281
4004_D058	Pin Control Register n (PORTE_PCR22)	32	R/W	0000_0000h	11.4.1/281
4004_D05C	Pin Control Register n (PORTE_PCR23)	32	R/W	0000_0000h	11.4.1/281
4004_D060	Pin Control Register n (PORTE_PCR24)	32	R/W	0000_0000h	11.4.1/281
4004_D064	Pin Control Register n (PORTE_PCR25)	32	R/W	0000_0000h	11.4.1/281
4004_D068	Pin Control Register n (PORTE_PCR26)	32	R/W	0000_0000h	11.4.1/281
4004_D06C	Pin Control Register n (PORTE_PCR27)	32	R/W	0000_0000h	11.4.1/281
4004_D070	Pin Control Register n (PORTE_PCR28)	32	R/W	0000_0000h	11.4.1/281
4004_D074	Pin Control Register n (PORTE_PCR29)	32	R/W	0000_0000h	11.4.1/281
4004_D078	Pin Control Register n (PORTE_PCR30)	32	R/W	0000_0000h	11.4.1/281
4004_D07C	Pin Control Register n (PORTE_PCR31)	32	R/W	0000_0000h	11.4.1/281
4004_D080	Global Pin Control Low Register (PORTE_GPCLR)	32	W (always reads zero)	0000_0000h	11.4.2/284
4004_D084	Global Pin Control High Register (PORTE_GPCHR)	32	W (always reads zero)	0000_0000h	11.4.3/284

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_D0A0	Interrupt Status Flag Register (PORTE_ISFR)	32	w1c	0000_0000h	11.4.4/285
4004_D0C0	Digital Filter Enable Register (PORTE_DFER)	32	R/W	0000_0000h	11.4.5/286
4004_D0C4	Digital Filter Clock Register (PORTE_DFCR)	32	R/W	0000_0000h	11.4.6/286
4004_D0C8	Digital Filter Width Register (PORTE_DFWR)	32	R/W	0000_0000h	11.4.7/287
4004_E000	Pin Control Register n (PORTF_PCR0)	32	R/W	0000_0000h	11.4.1/281
4004_E004	Pin Control Register n (PORTF_PCR1)	32	R/W	0000_0000h	11.4.1/281
4004_E008	Pin Control Register n (PORTF_PCR2)	32	R/W	0000_0000h	11.4.1/281
4004_E00C	Pin Control Register n (PORTF_PCR3)	32	R/W	0000_0000h	11.4.1/281
4004_E010	Pin Control Register n (PORTF_PCR4)	32	R/W	0000_0000h	11.4.1/281
4004_E014	Pin Control Register n (PORTF_PCR5)	32	R/W	0000_0000h	11.4.1/281
4004_E018	Pin Control Register n (PORTF_PCR6)	32	R/W	0000_0000h	11.4.1/281
4004_E01C	Pin Control Register n (PORTF_PCR7)	32	R/W	0000_0000h	11.4.1/281
4004_E020	Pin Control Register n (PORTF_PCR8)	32	R/W	0000_0000h	11.4.1/281
4004_E024	Pin Control Register n (PORTF_PCR9)	32	R/W	0000_0000h	11.4.1/281
4004_E028	Pin Control Register n (PORTF_PCR10)	32	R/W	0000_0000h	11.4.1/281
4004_E02C	Pin Control Register n (PORTF_PCR11)	32	R/W	0000_0000h	11.4.1/281
4004_E030	Pin Control Register n (PORTF_PCR12)	32	R/W	0000_0000h	11.4.1/281
4004_E034	Pin Control Register n (PORTF_PCR13)	32	R/W	0000_0000h	11.4.1/281
4004_E038	Pin Control Register n (PORTF_PCR14)	32	R/W	0000_0000h	11.4.1/281
4004_E03C	Pin Control Register n (PORTF_PCR15)	32	R/W	0000_0000h	11.4.1/281
4004_E040	Pin Control Register n (PORTF_PCR16)	32	R/W	0000_0000h	11.4.1/281
4004_E044	Pin Control Register n (PORTF_PCR17)	32	R/W	0000_0000h	11.4.1/281
4004_E048	Pin Control Register n (PORTF_PCR18)	32	R/W	0000_0000h	11.4.1/281
4004_E04C	Pin Control Register n (PORTF_PCR19)	32	R/W	0000_0000h	11.4.1/281
4004_E050	Pin Control Register n (PORTF_PCR20)	32	R/W	0000_0000h	11.4.1/281
4004_E054	Pin Control Register n (PORTF_PCR21)	32	R/W	0000_0000h	11.4.1/281
4004_E058	Pin Control Register n (PORTF_PCR22)	32	R/W	0000_0000h	11.4.1/281
4004_E05C	Pin Control Register n (PORTF_PCR23)	32	R/W	0000_0000h	11.4.1/281
4004_E060	Pin Control Register n (PORTF_PCR24)	32	R/W	0000_0000h	11.4.1/281
4004_E064	Pin Control Register n (PORTF_PCR25)	32	R/W	0000_0000h	11.4.1/281
4004_E068	Pin Control Register n (PORTF_PCR26)	32	R/W	0000_0000h	11.4.1/281
4004_E06C	Pin Control Register n (PORTF_PCR27)	32	R/W	0000_0000h	11.4.1/281
4004_E070	Pin Control Register n (PORTF_PCR28)	32	R/W	0000_0000h	11.4.1/281

Table continues on the next page...

PORT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_E074	Pin Control Register n (PORTF_PCR29)	32	R/W	0000_0000h	11.4.1/281
4004_E078	Pin Control Register n (PORTF_PCR30)	32	R/W	0000_0000h	11.4.1/281
4004_E07C	Pin Control Register n (PORTF_PCR31)	32	R/W	0000_0000h	11.4.1/281
4004_E080	Global Pin Control Low Register (PORTF_GPCLR)	32	W (always reads zero)	0000_0000h	11.4.2/284
4004_E084	Global Pin Control High Register (PORTF_GPCHR)	32	W (always reads zero)	0000_0000h	11.4.3/284
4004_E0A0	Interrupt Status Flag Register (PORTF_ISFR)	32	w1c	0000_0000h	11.4.4/285
4004_E0C0	Digital Filter Enable Register (PORTF_DFER)	32	R/W	0000_0000h	11.4.5/286
4004_E0C4	Digital Filter Clock Register (PORTF_DFCR)	32	R/W	0000_0000h	11.4.6/286
4004_E0C8	Digital Filter Width Register (PORTF_DFWR)	32	R/W	0000_0000h	11.4.7/287

11.4.1 Pin Control Register n (PORTx_PCR*n*)

NOTE

For PCR1 to PCR5 of the port A, bit 0, 1, 6, 8, 9,10 reset to 1; for the PCR0 of the port A, bit 1, 6, 8, 9, 10 reset to 1; in other conditions, all bits reset to 0.

Addresses: 4004_9000h base + 0h offset + (4d × *n*), where *n* = 0d to 31d

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							ISF	0				IRQC				LK	0				MUX			0	DSE	ODE	PFE	0	SRE	PE	PS
W								w1c																								
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

PORTx_PCR*n* field descriptions

Field	Description
31–25 Reserved	This read-only field is reserved and always has the value zero.
24 ISF	Interrupt Status Flag The pin interrupt configuration is valid in all digital pin muxing modes.

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
	0 Configured interrupt has not been detected. 1 Configured interrupt has been detected. If 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 one is written to that flag. If configured for a level sensitive interrupt that remains asserted then flag will set again immediately.
23–20 Reserved	This read-only field is reserved and always has the value zero.
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 as follows: 0000 Interrupt/DMA Request disabled. 0001 DMA Request on rising edge. 0010 DMA Request on falling edge. 0011 DMA Request on either edge. 0100 Reserved. 1000 Interrupt when logic zero. 1001 Interrupt on rising edge. 1010 Interrupt on falling edge. 1011 Interrupt on either edge. 1100 Interrupt when logic one. Others Reserved.
15 LK	Lock Register 0 Pin Control Register bits [15:0] are not locked. 1 Pin Control Register bits [15:0] are locked and cannot be updated until the next System Reset.
14–11 Reserved	This read-only field is reserved and always has the value zero.
10–8 MUX	Pin Mux Control The corresponding pin is configured as follows: 000 Pin Disabled (Analog). 001 Alternative 1 (GPIO). 010 Alternative 2 (chip specific). 011 Alternative 3 (chip specific). 100 Alternative 4 (chip specific). 101 Alternative 5 (chip specific). 110 Alternative 6 (chip specific). 111 Alternative 7 (chip specific / JTAG / NMI).
7 Reserved	This read-only field is reserved and always has the value zero.
6 DSE	Drive Strength Enable Drive Strength configuration is valid in all digital pin muxing modes.

Table continues on the next page...

PORTx_PCR*n* field descriptions (continued)

Field	Description
	0 Low drive strength is configured on the corresponding pin, if pin is configured as a digital output. 1 High drive strength is configured on the corresponding pin, if pin is configured as a digital output.
5 ODE	Open Drain Enable Open Drain configuration is valid in all digital pin muxing modes. 0 Open Drain output is disabled on the corresponding pin. 1 Open Drain output is enabled on the corresponding pin, provided pin is configured as a digital output.
4 PFE	Passive Filter Enable Passive Filter configuration is valid in all digital pin muxing modes. 0 Passive Input Filter is disabled on the corresponding pin. 1 Passive Input Filter is enabled on the corresponding pin, provided pin is configured as a digital input. A low pass filter (10 MHz to 30 MHz bandwidth) is enabled on the digital input path. Disable the Passive Input Filter when supporting high speed interfaces (> 2 MHz) on the pin.
3 Reserved	This read-only field is reserved and always has the value zero.
2 SRE	Slew Rate Enable Slew Rate configuration is valid in all digital pin muxing modes. 0 Fast slew rate is configured on the corresponding pin, if pin is configured as a digital output. 1 Slow slew rate is configured on the corresponding pin, if pin is configured as a digital output.
1 PE	Pull Enable Pull configuration is valid in all digital pin muxing modes. 0 Internal pull-up or pull-down resistor is not enabled on the corresponding pin. 1 Internal pull-up or pull-down resistor is enabled on the corresponding pin, provided pin is configured as a digital input.
0 PS	Pull Select Pull configuration is valid in all digital pin muxing modes. 0 Internal pull-down resistor is enabled on the corresponding pin, if the corresponding Port Pull Enable Register bit is set. 1 Internal pull-up resistor is enabled on the corresponding pin, if the corresponding Port Pull Enable Register bit is set.

11.4.2 Global Pin Control Low Register (PORTx_GPCLR)

Addresses: PORTA_GPCLR is 4004_9000h base + 80h offset = 4004_9080h
PORTB_GPCLR is 4004_A000h base + 80h offset = 4004_A080h
PORTC_GPCLR is 4004_B000h base + 80h offset = 4004_B080h
PORTD_GPCLR is 4004_C000h base + 80h offset = 4004_C080h
PORTE_GPCLR is 4004_D000h base + 80h offset = 4004_D080h
PORTF_GPCLR is 4004_E000h base + 80h offset = 4004_E080h

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																0															
W	GPWE																GPWD															
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

PORTx_GPCLR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable When set, causes bits [15:0] of the corresponding Pin Control Register (15 through 0) to update with the value in the Global Pin Write Data field.
15–0 GPWD	Global Pin Write Data Value to be written to bits [15:0] of all Pin Control Registers that are enabled by the Global Pin Write Enable field, provided the corresponding register has not been locked.

11.4.3 Global Pin Control High Register (PORTx_GPCHR)

Addresses: PORTA_GPCHR is 4004_9000h base + 84h offset = 4004_9084h
PORTB_GPCHR is 4004_A000h base + 84h offset = 4004_A084h
PORTC_GPCHR is 4004_B000h base + 84h offset = 4004_B084h
PORTD_GPCHR is 4004_C000h base + 84h offset = 4004_C084h
PORTE_GPCHR is 4004_D000h base + 84h offset = 4004_D084h
PORTF_GPCHR is 4004_E000h base + 84h offset = 4004_E084h

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																0															
W	GPWE																GPWD															
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

PORTx_GPCHR field descriptions

Field	Description
31–16 GPWE	Global Pin Write Enable

Table continues on the next page...

PORTx_GPCHR field descriptions (continued)

Field	Description
	When set, causes bits [15:0] of the corresponding Pin Control Register (31 through 16) to update with the value in the Global Pin Write Data field.
15–0 GPWD	Global Pin Write Data Value to be written to bits [15:0] of all Pin Control Registers that are enabled by the Global Pin Write Enable field, provided the corresponding register has not been locked.

11.4.4 Interrupt Status Flag Register (PORTx_ISFR)

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.

Addresses: PORTA_ISFR is 4004_9000h base + A0h offset = 4004_90A0h
PORTB_ISFR is 4004_A000h base + A0h offset = 4004_A0A0h
PORTC_ISFR is 4004_B000h base + A0h offset = 4004_B0A0h
PORTD_ISFR is 4004_C000h base + A0h offset = 4004_C0A0h
PORTE_ISFR is 4004_D000h base + A0h offset = 4004_D0A0h
PORTF_ISFR is 4004_E000h base + A0h offset = 4004_E0A0h

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	ISF																															
W	w1c																															
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

PORTx_ISFR field descriptions

Field	Description
31–0 ISF	Interrupt Status Flag Each bit in the field indicates the detection of the configured interrupt of the same number as the bit. 0 Configured interrupt has not been detected. 1 Configured interrupt has been detected. If 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 one is written to the flag. If configured for a level sensitive interrupt and the pin remains asserted then the flag will set again immediately after it is cleared.

11.4.5 Digital Filter Enable Register (PORTx_DFER)

Addresses: PORTA_DFER is 4004_9000h base + C0h offset = 4004_90C0h
PORTB_DFER is 4004_A000h base + C0h offset = 4004_A0C0h
PORTC_DFER is 4004_B000h base + C0h offset = 4004_B0C0h
PORTD_DFER is 4004_C000h base + C0h offset = 4004_C0C0h
PORTE_DFER is 4004_D000h base + C0h offset = 4004_D0C0h
PORTF_DFER is 4004_E000h base + C0h offset = 4004_E0C0h

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	DFE																															
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

PORTx_DFER field descriptions

Field	Description
31–0 DFE	Digital Filter Enable The digital filter configuration is valid in all digital pin muxing modes. The output of each digital filter is reset to zero at system reset and whenever the digital filter is disabled. 0 Digital Filter is disabled on the corresponding pin and output of the digital filter is reset to zero.Each bit in the field enables the digital filter of the same number as the bit. 1 Digital Filter is enabled on the corresponding pin, provided pin is configured as a digital input.

11.4.6 Digital Filter Clock Register (PORTx_DFCR)

Addresses: PORTA_DFCR is 4004_9000h base + C4h offset = 4004_90C4h
PORTB_DFCR is 4004_A000h base + C4h offset = 4004_A0C4h
PORTC_DFCR is 4004_B000h base + C4h offset = 4004_B0C4h
PORTD_DFCR is 4004_C000h base + C4h offset = 4004_C0C4h
PORTE_DFCR is 4004_D000h base + C4h offset = 4004_D0C4h
PORTF_DFCR is 4004_E000h base + C4h offset = 4004_E0C4h

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																															CS
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

PORTx_DFCR field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value zero.
0 CS	Clock Source The digital filter configuration is valid in all digital pin muxing modes. Configures the clock source for the digital input filters. Changing the filter clock source should only be done after disabling all enabled digital filters. 0 Digital Filters are clocked by the bus clock. 1 Digital Filters are clocked by the 1 kHz LPO clock.

11.4.7 Digital Filter Width Register (PORTx_DFWR)

The digital filter configuration is valid in all digital pin muxing modes.

Addresses: PORTA_DFWR is 4004_9000h base + C8h offset = 4004_90C8h
PORTB_DFWR is 4004_A000h base + C8h offset = 4004_A0C8h
PORTC_DFWR is 4004_B000h base + C8h offset = 4004_B0C8h
PORTD_DFWR is 4004_C000h base + C8h offset = 4004_C0C8h
PORTE_DFWR is 4004_D000h base + C8h offset = 4004_D0C8h
PORTF_DFWR is 4004_E000h base + C8h offset = 4004_E0C8h

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																											FILT				
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

PORTx_DFWR field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value zero.
4–0 FILT	Filter Length The digital filter configuration is valid in all digital pin muxing modes. Configures the maximum size of the glitches (in clock cycles) the digital filter absorbs for enabled digital filters. Glitches that are longer than this register setting (in clock cycles) will pass through the digital filter, while glitches that are equal to or less than this register setting (in clock cycles) will be filtered. Changing the filter length should only be done after disabling all enabled filters.

11.5 Functional description

11.5.1 Pin control

Each port pin has a corresponding Pin Control Register PORT_PCRn associated with it.

The upper half of the Pin Control Register configures the pin's capability to either interrupt the CPU or request a DMA transfer, on a rising/falling edge or both edges as well as a logic level occurring on the port pin. It also includes a flag to indicate that an interrupt has occurred.

The lower half of the pin control register configures the following functions for each pin within the 32-bit port.

- Pullup or pulldown enable
- Drive strength and slew rate configuration
- Open drain enable
- Passive input filter enable
- Pin muxing mode

These functions apply across all digital pin muxing modes and individual peripherals do not override the configuration in this register (for example, if an I^{2}C function is enabled on a pin then that does not override the pullup or open drain configuration for that pin).

When the pin muxing mode is configured for disabled (analog), then the all digital functions on that pin are disabled. This includes the pullup and pulldown enables, digital output buffer enable, digital input buffer enable and passive filter enable.

A lock bit also exists that allows the configuration for each pin to be locked until the next system reset. Once locked, writes to the lower half of that pin control register are ignored, although a bus error is not generated on an attempted write to a locked register.

The configuration of each pin control register is retained when the PORT module is disabled.

11.5.2 Global pin control

The two global pin control registers allow a single register write to update the lower half of the pin control register on up to sixteen pins, all with the same value. Registers that are locked cannot be written using the global pin control registers.

The global pin control registers are designed to enable software to quickly configure multiple pins within the one port for the same peripheral function. Note however that interrupt functions are unable to be configured using the global pin control registers.

The global pin control registers are write only registers, that always read as zero.

11.5.3 External interrupts

The external interrupt capability of the PORT module are 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)
- 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 output of the digital filter (if enabled) or pin (if digital filter is bypassed). 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 interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that port. The interrupt negates once the interrupt status flags for all enabled interrupts have been cleared by writing a logic 0 to the ISF flag in the PORT_PCRn register.

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 once the DMA transfer has been completed, since that clears the interrupt status flags for all enabled DMA requests.

During stop mode, the interrupt status flag for any enabled interrupt (but not DMA request) will asynchronously set if the required level or edge is detected. This also generates an asynchronous wakeup signal to exit the low power mode.

11.5.4 Digital filter

The digital filter capabilities of the PORT module are available in all digital pin muxing modes provided the PORT module is enabled.

The clock used for all digital filters within the one port can be configured between the bus clock or the 1 kHz LPO clock. This selection should be changed only when all digital filters for that port are disabled. If the digital filters for a port are configured to use the bus clock, then the digital filters are bypassed (and do not update) during stop mode.

The filter width in clock size is the same for all enabled digital filters within the one port and should be changed only when all digital filters for that port are disabled.

The output of each digital filter is logic zero after system reset and whenever a digital filter is disabled. Once a digital filter is enabled, the input is synchronized to the filter clock (either the bus clock or the 1 kHz LPO clock). If the synchronized input and the output of the digital filter remain different for a number of filter clock cycles equal to the filter width register configuration, then the output of the digital filter updates to equal the synchronized filter input.

The minimum latency through a digital filter equals two or three filter clock cycles plus the filter width configuration register.

Chapter 12
System integration module (SIM)

12.1 Introduction

NOTE

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

The system integration module (SIM) provides system control and chip configuration registers.

12.1.1 Features

- Configuration for system clocking
  - Clock source selection for ESDHC, I^{2}S, NFC, USB, and PLL/FLL source
  - System clock divide values
  - NFC, I^{2}S, and USB clock divide values
- Architectural clock gating control
- Flash configuration
- USB regulator configuration
- RAM repair control
- Flextimer external clock and fault source selection
- UART0 and UART1 receive/transmit source selection/configuration

12.1.2 Modes of operation

- Run mode
- Sleep mode
- Deep sleep mode
- VLLS mode

12.2 Memory Map and Registers

NOTE

The SIM_SOPT1 and SIM_SOPT1CFG register sare located at a different base address than the other SIM registers.

SIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_7000	System Options Register 1 (SIM_SOPT1)	32	R/W	See section	12.2.1/293
4004_7004	SOPT1 Configuration Register (SIM_SOPT1CFG)	32	R/W	0000_0000h	12.2.2/295
4004_8004	System Options Register 2 (SIM_SOPT2)	32	R/W	4400_1004h	12.2.3/296
4004_800C	System Options Register 4 (SIM_SOPT4)	32	R/W	0000_0000h	12.2.4/299
4004_8010	System Options Register 5 (SIM_SOPT5)	32	R/W	See section	12.2.5/302
4004_8014	System Options Register 6 (SIM_SOPT6)	32	R/W	0000_0000h	12.2.6/303
4004_8018	System Options Register 7 (SIM_SOPT7)	32	R/W	0000_0000h	12.2.7/304
4004_8024	System Device Identification Register (SIM_SDID)	32	R	See section	12.2.8/307
4004_8028	System Clock Gating Control Register 1 (SIM_SCGC1)	32	R/W	0000_0000h	12.2.9/308
4004_802C	System Clock Gating Control Register 2 (SIM_SCGC2)	32	R/W	0000_0000h	12.2.10/ 309
4004_8030	System Clock Gating Control Register 3 (SIM_SCGC3)	32	R/W	0000_0000h	12.2.11/ 310
4004_8034	System Clock Gating Control Register 4 (SIM_SCGC4)	32	R/W	F010_0030h	12.2.12/ 312
4004_8038	System Clock Gating Control Register 5 (SIM_SCGC5)	32	R/W	0004_0180h	12.2.13/ 314
4004_803C	System Clock Gating Control Register 6 (SIM_SCGC6)	32	R/W	4000_0001h	12.2.14/ 316
4004_8040	System Clock Gating Control Register 7 (SIM_SCGC7)	32	R/W	0000_0007h	12.2.15/ 318
4004_8044	System Clock Divider Register 1 (SIM_CLKDIV1)	32	R/W	See section	12.2.16/ 319
4004_8048	System Clock Divider Register 2 (SIM_CLKDIV2)	32	R/W	0000_0000h	12.2.17/ 321
4004_804C	Flash Configuration Register 1 (SIM_FCFG1)	32	R/W	See section	12.2.18/ 322
4004_8050	Flash Configuration Register 2 (SIM_FCFG2)	32	R	See section	12.2.19/ 324

Table continues on the next page...

SIM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_8054	Unique Identification Register High (SIM_UIDH)	32	R	Undefined	12.2.20/ 325
4004_8058	Unique Identification Register Mid-High (SIM_UIDMH)	32	R	Undefined	12.2.21/ 325
4004_805C	Unique Identification Register Mid Low (SIM_UIDML)	32	R	Undefined	12.2.22/ 326
4004_8060	Unique Identification Register Low (SIM_UIDL)	32	R	Undefined	12.2.23/ 326
4004_8068	System Clock Divider Register 4 (SIM_CLKDIV4)	32	R/W	0000_0002h	12.2.24/ 327
4004_806C	Misc Control Register (SIM_MCR)	32	R/W	0000_0000h	12.2.25/ 328

12.2.1 System Options Register 1 (SIM_SOPT1)

NOTE

The SOPT1 register is only reset on POR or LVD.

Address: SIM_SOPT1 is 4004_7000h base + 0h offset = 4004_7000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	USBREGEN	USBSSTBY	USBVSTBY	0									OSC32KSEL	0		
W																
Reset	1	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	RAMSIZE				0		0		0		0					
W																
Reset	1	0	0	1	0	0	0	0	0	0	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

SIM_SOPT1 field descriptions

Field	Description
31 USBREGEN	USB voltage regulator enable

Table continues on the next page...

SIM_SOPT1 field descriptions (continued)

Field	Description
	Controls whether the USB voltage regulator is enabled. This bit can only be written when the SOPT1CFG[URWE] bit is set. 0 USB voltage regulator is disabled. 1 USB voltage regulator is enabled
30 USBSSTBY	USB voltage regulator in standby mode during Stop, VLPS, LLS or VLLS Controls whether the USB voltage regulator is placed in standby mode during Stop, VLPS, LLS and VLLS modes. This bit can only be written when the SOPT1CFG[USSWE] bit is set. 0 USB voltage regulator not in standby during Stop, VLPS, LLS and VLLS modes. 1 USB voltage regulator in standby during Stop, VLPS, LLS and VLLS modes.
29 USBVSTBY	USB voltage regulator in standby mode during VLPR or VLPW Controls whether the USB voltage regulator is placed in standby mode during VLPR and VLPW modes. This bit can only be written when the SOPT1CFG[UVSWE] bit is set. 0 USB voltage regulator not in standby during VLPR and VLPW modes. 1 USB voltage regulator in standby during VLPR and VLPW modes.
28–20 Reserved	This read-only field is reserved and always has the value zero.
19 OSC32KSEL	32 kHz oscillator clock select Selects either the system oscillator or RTC oscillator as the 32 kHz clock source (ERCLK32K). This bit is reset only for POR/LVD. 0 System oscillator (OSC32KCLK) 1 RTC oscillator
18–16 Reserved	This read-only field is reserved and always has the value zero.
15–12 RAMSIZE	RAM size This field specifies the amount of system RAM available on the device. 0000 Undefined 0001 Undefined 0010 Undefined 0011 Undefined 0100 Undefined 0101 Undefined 0110 Undefined 0111 Undefined 1000 Undefined 1001 128 KB 1010 Undefined 1011 Undefined 1100 Undefined 1101 Undefined

Table continues on the next page...

SIM_SOPT1 field descriptions (continued)

Field	Description
	1110 Undefined 1111 Undefined
11–10 Reserved	This read-only field is reserved and always has the value zero.
9–8 Reserved	This read-only field is reserved and always has the value zero.
7–6 Reserved	This read-only field is reserved and always has the value zero.
5–0 Reserved	This read-only field is reserved and always has the value zero.

12.2.2 SOPT1 Configuration Register (SIM_SOPT1CFG)

Address: SIM_SOPT1CFG is 4004_7000h base + 4h offset = 4004_7004h

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					USSWE	UVSWE	URWE	0																							
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

SIM_SOPT1CFG field descriptions

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value zero.
26 USSWE	USB voltage regulator stop standby write enable Writing one to this bit allows the SOPT1[USBSSTBY] bit to be written. This register bit clears after a write to SOPT1[USBSSTBY]. 0 SOPT1[USBSSTBY] cannot be written. 1 SOPT1[USBSSTBY] can be written.
25 UVSWE	USB voltage regulator VLP standby write enable Writing one to this bit allows the SOPT1[USBVSTBY] bit to be written. This register bit clears after a write to SOPT1[USBVSTBY]. 0 SOPT1[USBVSTBY] cannot be written. 1 SOPT1[USBVSTBY] can be written.
24 URWE	USB voltage regulator enable write enable Writing one to this bit allows the SOPT1[USBREGEN] bit to be written. This register bit clears after a write to SOPT1[USBREGEN].

Table continues on the next page...

SIM_SOPT1CFG field descriptions (continued)

Field	Description
	0 SOPT1[USBREGEN] cannot be written. 1 SOPT1[USBREGEN] can be written.
23–0 Reserved	This read-only field is reserved and always has the value zero.

12.2.3 System Options Register 2 (SIM_SOPT2)

Address: SIM_SOPT2 is 4004_7000h base + 1004h offset = 4004_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NFCSRC		ESDHCSRC		0		0		USBFSRC		0		0	USBF_CLKSEL	PLLFLLSEL	
W																
Reset	0	1	0	0	0	1	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	NFC_CLKSEL	0	0	TRACECLKSEL	CMTUARTPAD	0	FBSL		CLKOUTSEL			RTCCLKOUTSEL	USBHSRC		0	
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0

SIM_SOPT2 field descriptions

Field	Description
31–30 NFCSRC	NFC Flash clock source select Selects the the clock divider source for NFC flash clock . 00 Bus clock 01 MCGPLL0CLK 10 MCGPLL1CLK 11 OSC0ERCLK
29–28 ESDHCSRC	ESDHC perclk source select Selects the clock source for the ESDHC clock . 00 Core/system clock 01 MCGPLLCLK/MCGFLLCLK selected by PLLFLLSEL[1:0] 10 OSC0ERCLK 11 External bypass clock (PTD11)

Table continues on the next page...

SIM_SOPT2 field descriptions (continued)

Field	Description
27–26 Reserved	This read-only field is reserved and always has the value zero.
25–24 Reserved	This read-only field is reserved and always has the value zero.
23–22 USBFSRC	USB FS clock source select Selects the clock source for the USB 48 MHz clock. 00 MCGPLLCLK/MCGFLLCLK selected by PLLFLLSEL[1:0] 01 MCGPLL0CLK 10 MCGPLL1CLK 11 OSC0ERCLK
21–20 Reserved	This read-only field is reserved and always has the value zero.
19 Reserved	This read-only field is reserved and always has the value zero.
18 USBF_CLKSEL	USB FS clock select Selects clock diver output or bypass clock. 0 External bypass clock (PTE26) 1 Clock divider USB FS clock
17–16 PLLFLLSEL	PLL/FLL clock select Selects the MCGPLLCLK or MCGFLLCLK clock for various peripheral clocking options. 00 MCGFLLCLK 01 MCGPLL0CLK 10 MCGPLL1CLK 11 System Platform clock
15 NFC_CLKSEL	NFC Flash clock select 0 Clock divider NFC clock 1 EXTAL1 clock.
14 Reserved	This read-only field is reserved and always has the value zero.
13 Reserved	This read-only field is reserved and always has the value zero.
12 TRACECLKSEL	Debug trace clock select Selects either the core/system clock or PLL clock as the trace clock source. 0 MCGCLKOUT 1 Core/system clock
11 CMTUARTPAD	CMT/UART pad drive strength Controls the output drive strength of the CMT IRO signal or UART0_TXD signal on PTD7 pin by selecting either one or two pads to drive it.

Table continues on the next page...

SIM_SOPT2 field descriptions (continued)

Field	Description
	0 Single-pad drive strength for CMT IRO or UART0_TXD. 1 Dual-pad drive strength for CMT IRO or UART0_TXD.
10 Reserved	This read-only field is reserved and always has the value zero.
9–8 FBSL	Flexbus security level If security is enabled, then this field affects what CPU operations can access off-chip via the FlexBus interface. This field has no effect if security is not enabled. 00 All off-chip accesses (op code and data) via the FlexBus are disallowed. 10 Off-chip op code accesses are disallowed. Data accesses are allowed. 11 Off-chip op code accesses and data accesses are allowed.
7–5 CLKOUTSEL	Clock out select Selects what internal clock to output on CLKOUT pin. 000 FlexBus clock (reset value) 001 Reserved 010 Flash ungated clock 011 LPO clock (1 kHz) 100 MCGIRCLK 101 RTC 32 kHz clock 110 OSC0ERCLK 111 OSC1ERCLK
4 RTCCLKOUTSEL	RTC clock out select Selects either the RTC 32 kHz clock or the RTC 1 Hz clock for the clock to the RTC CLKOUT pin (PTE26). PTE26 must be configured for the RTC CLKOUT function. 0 RTC 1 Hz clock drives RTC CLKOUT. 1 RTC 32 kHz oscillator drives RTC CLKOUT.
3–2 USBHSRC	USB HS clock source select Selects the clock source before clock divider for the USB 60 MHz clock. 00 Bus clock 01 MCGPLL0CLK 10 MCGPLL1CLK 11 OSC0ERCLK
1–0 Reserved	This read-only field is reserved and always has the value zero.

12.2.4 System Options Register 4 (SIM_SOPT4)

Address: SIM_SOPT4 is 4004_7000h base + 100Ch offset = 4004_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FTM3TRG1SRC	FTM3TRG0SRC	FTM0TRG1SRC	FTM0TRG0SRC	FTM3CLKSEL	FTM2CLKSEL	FTM1CLKSEL	FTM0CLKSEL	0		FTM2CH0SRC		FTM1CH0SRC		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		FTM3FLT0		0		FTM2FLT0		0		FTM1FLT0	FTM0FLT3	FTM0FLT2	FTM0FLT1	FTM0FLT0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_SOPT4 field descriptions

Field	Description
31 FTM3TRG1SRC	FlexTimer 3 hardware trigger 1 source select Selects the source of FTM3 hardware trigger 1. 0 PDB output trigger 3 drives FTM3 hardware trigger 1. 1 FTM2 channel match trigger drives FTM3 hardware trigger 1.
30 FTM3TRG0SRC	FlexTimer 3 hardware trigger 0 source select Selects the source of FTM3 hardware trigger 0. 0 CMP3 OUT drives FTM3 hardware trigger 0. 1 FTM1 channel match trigger drives FTM3 hardware trigger 0.
29 FTM0TRG1SRC	FlexTimer 0 hardware trigger 1 source select Selects the source of FTM0 hardware trigger 1. 0 PDB output trigger 1 drives FTM0 hardware trigger 1. 1 FTM2 channel match trigger drives FTM0 hardware trigger 1.
28 FTM0TRG0SRC	FlexTimer 0 hardware trigger 0 source select Selects the source of FTM0 hardware trigger 0. 0 CMP0 OUT drives FTM0 hardware trigger 0. 1 FTM1 channel match trigger drives FTM0 hardware trigger 0.
27 FTM3CLKSEL	FlexTimer 3 external clock pin select

Table continues on the next page...

SIM_SOPT4 field descriptions (continued)

Field	Description
	Selects the external pin used to drive the external clock to the FTM 3 module. Note that the selected pin must also be configured for the FTM external clock function through the appropriate PCTL pin control register. 0 FTM3 external clock driven by FTM CLKIN0 pin. 1 FTM3 external clock driven by FTM CLKIN1 pin .
26 FTM2CLKSEL	FlexTimer 2 external clock pin select Selects the external pin used to drive the external clock to the FTM2 module. Note that the selected pin must also be configured for the FTM external clock function through the appropriate PCTL pin control register. 0 FTM2 external clock driven by FTM CLKIN0 pin. 1 FTM2 external clock driven by FTM CLKIN1 pin.
25 FTM1CLKSEL	FlexTimer 1 external clock pin select Selects the external pin used to drive the external clock to the FTM1 module. Note that the selected pin must also be configured for the FTM external clock function through the appropriate PCTL pin control register. 0 FTM1 external clock driven by FTM CLKIN0 pin. 1 FTM1 external clock driven by FTM CLKIN1 pin.
24 FTM0CLKSEL	FlexTimer 0 external clock pin select Selects the external pin used to drive the external clock to the FTM0 module. Note that the selected pin must also be configured for the FTM external clock function through the appropriate PCTL pin control register. 0 FTM0 external clock driven by FTM CLKIN0 pin 1 FTM0 external clock driven by FTM CLKIN1 pin.
23–22 Reserved	This read-only field is reserved and always has the value zero.
21–20 FTM2CH0SRC	FlexTimer 2 channel 0 input capture source select Selects the source for FTM 2 channel 0 input capture. 00 FTM2_CH0 pin 01 CMP0 output 10 CMP1 output 11 Reserved
19–18 FTM1CH0SRC	FlexTimer 1 channel 0 input capture source select Selects the source for FTM 1 channel 0 input capture. 00 FTM1_CH0 pin 01 CMP0 output 10 CMP1 output 11 USB SOF trigger
17–13 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

SIM_SOPT4 field descriptions (continued)

Field	Description
12 FTM3FLT0	FlexTimer 3 Fault 0 Select. Selects the source of FTM 3 fault 0. Note that the pin source for fault 0 must be configured for the FTM fault function through the appropriate PCTL pin control register. 0 FTM3_FLT0 drives FTM 2 fault 0. 1 CMP0 OUT drives FTM 2 fault 0.
11–9 Reserved	This read-only field is reserved and always has the value zero.
8 FTM2FLT0	FlexTimer 2 Fault 0 Select Selects the source of FTM 2 fault 0. Note that the pin source for fault 0 must be configured for the FTM fault function through the appropriate PCTL pin control register. 0 FTM2_FLT0 drives FTM 2 fault 0. 1 CMP0 OUT drives FTM 2 fault 0.
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 FTM1FLT0	FlexTimer 1 Fault 0 Select Selects the source of FTM 1 fault 0. Note that the pin source for fault 0 must be configured for the FTM fault function through the appropriate PCTL pin control register. 0 FTM1_FLT0 drives FTM 1 fault 0. 1 CMP0 OUT drives FTM 1 fault 0.
3 FTM0FLT3	FlexTimer 0 Fault 3 Select. Selects the source of FTM 0 fault 3. Note that the pin source for fault 3 must be configured for the FTM fault function through the appropriate PCTL pin control register. 0 FTM0_FLT3 drives FTM 0 fault 3. 1 CMP0 OUT drives FTM 0 fault 3.
2 FTM0FLT2	FlexTimer 0 Fault 2 Select Selects the source of FTM 0 fault 2. Note that the pin source for fault 2 must be configured for the FTM fault function through the appropriate PCTL pin control register. 0 FTM0_FLT2 drives FTM 0 fault 2. 1 CMP2 OUT drives FTM 0 fault 2.
1 FTM0FLT1	FlexTimer 0 Fault 1 Select Selects the source of FTM 0 fault 1. Note that the pin source for fault 1 must be configured for the FTM fault function through the appropriate PCTL pin control register. 0 FTM0_FLT1 drives FTM 0 fault 1. 1 CMP1 OUT drives FTM 0 fault 1.
0 FTM0FLT0	FlexTimer 0 Fault 0 Select Selects the source of FTM 0 fault 0. Note that the pin source for fault 0 must be configured for the FTM fault function through the appropriate PCTL pin control register.

Table continues on the next page...

SIM_SOPT4 field descriptions (continued)

Field	Description
	0 FTM0_FLT0 drives FTM 0 fault 0. 1 CMP0 OUT drives FTM 0 fault 0.

12.2.5 System Options Register 5 (SIM_SOPT5)

Address: SIM_SOPT5 is 4004_7000h base + 1010h offset = 4004_8010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	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								UART1RXSRC		UART1TXSRC		UART0RXSRC		UART0TXSRC	
W																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	0	0	0	0	0	0	0	0

* Notes:

- x = Undefined at reset.

SIM_SOPT5 field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–6 UART1RXSRC	UART1 receive data source select Selects the source for the UART1 receive data. 00 UART1_RX pin 01 CMP0 10 CMP1 11 Reserved
5–4 UART1TXSRC	UART1 transmit data source select Selects the source for the UART1 transmit data. 00 UART1_TX pin 01 UART1_TX pin modulated with FTM1 channel 0 Output 10 UART1_TX pin modulated with FTM2 channel 0 Output 11 Reserved
3–2 UART0RXSRC	UART0 receive data source select Selects the source for the UART0 receive data.

Table continues on the next page...

SIM_SOPT5 field descriptions (continued)

Field	Description
	00 UART0_RX pin 01 CMP0 10 CMP1 11 Reserved
1–0 UART0TXSRC	UART0 transmit data source select Selects the source for the UART0 transmit data. 00 UART0_TX pin 01 UART0_TX pin modulated with FTM1 channel 0 output 10 UART0_TX pin modulated with FTM2 channel 0 output 11 Reserved

12.2.6 System Options Register 6 (SIM_SOPT6)

Address: SIM_SOPT6 is 4004_7000h base + 1014h offset = 4004_8014h

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												PCR				MCC															
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

SIM_SOPT6 field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–16 PCR	PCR FlexBus hold cycles before FlexBus can release bus to NFC or to IDLE.
15–0 MCC	MCC NFC hold cycle in case FlexBus request while NFC is granted.

12.2.7 System Options Register 7 (SIM_SOPT7)

Address: SIM_SOPT7 is 4004_7000h base + 1018h offset = 4004_8018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADC3ALTTRGEN	0		ADC3PRETRGSEL	ADC3TRGSEL				ADC2ALTTRGEN	0		ADC2PRETRGSEL	ADC2TRGSEL			
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	ADC1ALTTRGEN	0		ADC1PRETRGSEL	ADC1TRGSEL				ADC0ALTTRGEN	0		ADC0PRETRGSEL	ADC0TRGSEL			
W		▨								▨						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_SOPT7 field descriptions

Field	Description
31 ADC3ALTTRGEN	ADC3 alternate trigger enable Enable alternative conversion triggers for ADC3. 0 PDB trigger selected for ADC3. 1 Alternate trigger selected for ADC3.
30–29 Reserved	This read-only field is reserved and always has the value zero.
28 ADC3PRETRGSEL	ADC3 pre-trigger select Selects the ADC3 pre-trigger source when alternative triggers are enabled through ADC3ALTTRGEN. 0 Pre-trigger A selected for ADC3. 1 Pre-trigger B selected for ADC3.
27–24 ADC3TRGSEL	ADC3 trigger select Selects the ADC3 trigger source when alternative triggers are enabled through ADC3ALTTRGEN. 0000 External trigger 0001 High speed comparator 0 asynchronous interrupt 0010 High speed comparator 1 asynchronous interrupt 0011 High speed comparator 2 asynchronous interrupt

Table continues on the next page...

SIM_SOPT7 field descriptions (continued)

Field	Description
	0100 PIT trigger 0 0101 PIT trigger 1 0110 PIT trigger 2 0111 PIT trigger 3 1000 FTM0 trigger 1001 FTM1 trigger 1010 FTM2 trigger 1011 FTM3 trigger 1100 RTC alarm 1101 RTC seconds 1110 Low-power timer trigger 1111 High speed comparator 3 asynchronous interrupt
23 ADC2ALTTRGEN	ADC2 alternate trigger enable Enable alternative conversion triggers for ADC2. 0 PDB trigger selected for ADC2. 1 Alternate trigger selected for ADC2.
22–21 Reserved	This read-only field is reserved and always has the value zero.
20 ADC2PRETRGSEL	ADC2 pre-trigger select Selects the ADC2 pre-trigger source when alternative triggers are enabled through ADC2ALTTRGEN. 0 Pre-trigger A selected for ADC2. 1 Pre-trigger B selected for ADC2.
19–16 ADC2TRGSEL	ADC2 trigger select Selects the ADC2 trigger source when alternative triggers are enabled through ADC2ALTTRGEN. 0000 External trigger 0001 High speed comparator 0 asynchronous interrupt 0010 High speed comparator 1 asynchronous interrupt 0011 High speed comparator 2 asynchronous interrupt 0100 PIT trigger 0 0101 PIT trigger 1 0110 PIT trigger 2 0111 PIT trigger 3 1000 FTM0 trigger 1001 FTM1 trigger 1010 FTM2 trigger 1011 FTM3 trigger 1100 RTC alarm 1101 RTC seconds 1110 Low-power timer trigger 1111 High speed comparator 3 asynchronous interrupt
15 ADC1ALTTRGEN	ADC1 alternate trigger enable

Table continues on the next page...

SIM_SOPT7 field descriptions (continued)

Field	Description
	Enable alternative conversion triggers for ADC1. 0 PDB trigger selected for ADC1. 1 Alternate trigger selected for ADC1.
14–13 Reserved	This read-only field is reserved and always has the value zero.
12 ADC1PRETRGSEL	ADC1 pre-trigger select Selects the ADC1 pre-trigger source when alternative triggers are enabled through ADC1ALTTRGEN. 0 Pre-trigger A selected for ADC1. 1 Pre-trigger B selected for ADC1.
11–8 ADC1TRGSEL	ADC1 trigger select Selects the ADC1 trigger source when alternative triggers are enabled through ADC1ALTTRGEN. 0000 External trigger 0001 High speed comparator 0 asynchronous interrupt 0010 High speed comparator 1 asynchronous interrupt 0011 High speed comparator 2 asynchronous interrupt 0100 PIT trigger 0 0101 PIT trigger 1 0110 PIT trigger 2 0111 PIT trigger 3 1000 FTM0 trigger 1001 FTM1 trigger 1010 FTM2 trigger 1011 FTM3 trigger 1100 RTC alarm 1101 RTC seconds 1110 Low-power timer trigger 1111 High speed comparator 3 asynchronous interrupt
7 ADC0ALTTRGEN	ADC0 alternate trigger enable Enable alternative conversion triggers for ADC0. 0 PDB trigger selected for ADC0. 1 Alternate trigger selected for ADC0.
6–5 Reserved	This read-only field is reserved and always has the value zero.
4 ADC0PRETRGSEL	ADC0 pre-trigger select Selects the ADC0 pre-trigger source when alternative triggers are enabled through ADC0ALTTRGEN. 0 Pre-trigger A selected for ADC0. 1 Pre-trigger B selected for ADC0.
3–0 ADC0TRGSEL	ADC0 trigger select Selects the ADC0 trigger source when alternative triggers are enabled through ADC0ALTTRGEN.

Table continues on the next page...

SIM_SOPT7 field descriptions (continued)

Field	Description
	0000 External trigger 0001 High speed comparator 0 asynchronous interrupt 0010 High speed comparator 1 asynchronous interrupt 0011 High speed comparator 2 asynchronous interrupt 0100 PIT trigger 0 0101 PIT trigger 1 0110 PIT trigger 2 0111 PIT trigger 3 1000 FTM0 trigger 1001 FTM1 trigger 1010 FTM2 trigger 1011 FTM3 trigger 1100 RTC alarm 1101 RTC seconds 1110 Low-power timer trigger 1111 High speed comparator 3 asynchronous interrupt

12.2.8 System Device Identification Register (SIM_SDID)

Address: SIM_SDID is 4004_7000h base + 1024h offset = 4004_8024h

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																REVID				0		0	1	1	FAMID			PINID			
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	1	1	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

SIM_SDID field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–12 REVID	Device revision number Specifies the silicon implementation number for the device.
11–10 Reserved	This read-only field is reserved and always has the value zero.
9 Reserved	This read-only field is reserved and always has the value zero.
8 Reserved	This read-only field is reserved and always has the value one.

Table continues on the next page...

SIM_SDID field descriptions (continued)

Field	Description
7 Reserved	This read-only field is reserved and always has the value one.
6–4 FAMID	Kinetis family identification Specifies the Kinetis family of the device. 000 K10 001 K20 010 K61 011 Reserved 100 K60 101 K70 110 Reserved 111 Reserved
3–0 PINID	Pincount identification Specifies the pincount of the device. 0000 Reserved 0001 Reserved 0010 Reserved 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 Reserved 1000 Reserved 1001 Reserved 1010 144-pin 1011 Reserved 1100 196-pin 1101 Reserved 1110 256-pin 1111 Reserved

12.2.9 System Clock Gating Control Register 1 (SIM_SCGC1)

Address: SIM_SCGC1 is 4004_7000h base + 1028h offset = 4004_8028h

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																				UART5	UART4	0				OSC1	0				
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

SIM_SCGC1 field descriptions

Field	Description
31–12 Reserved	This read-only field is reserved and always has the value zero.
11 UART5	UART5 clock gate control 0 Clock is disabled. 1 Clock is enabled.
10 UART4	UART4 clock gate control 0 Clock is disabled. 1 Clock is enabled.
9–6 Reserved	This read-only field is reserved and always has the value zero.
5 OSC1	OSC1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
4–0 Reserved	This read-only field is reserved and always has the value zero.

12.2.10 System Clock Gating Control Register 2 (SIM_SCGC2)

Address: SIM_SCGC2 is 4004_7000h base + 102Ch offset = 4004_802Ch

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																		DAC1	DAC0	0											ENET
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

SIM_SCGC2 field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value zero.
13 DAC1	12BDAC1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
12 DAC0	12BDAC0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
11–1 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

SIM_SCGC2 field descriptions (continued)

Field	Description
0 ENET	ENET clock gate control 0 Clock is disabled. 1 Clock is enabled.

12.2.11 System Clock Gating Control Register 3 (SIM_SCGC3)

Address: SIM_SCGC3 is 4004_7000h base + 1030h offset = 4004_8030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			ADC3	ADC1	0	FTM3	FTM2	0	0	0				ESDHC	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	SAI1	0	0	DSPI2	0			NFC	0			FLEXCAN1	0			RNGA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_SCGC3 field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
28 ADC3	ADC3 clock gate control 0 Clock is disabled. 1 Clock is enabled.
27 ADC1	ADC1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
26 Reserved	This read-only field is reserved and always has the value zero.
25 FTM3	FTM3 clock gate control 0 Clock is disabled. 1 Clock is enabled.
24 FTM2	FTM2 clock gate control

Table continues on the next page...

SIM_SCGC3 field descriptions (continued)

Field	Description
	0 Clock is disabled. 1 Clock is enabled.
23 Reserved	This read-only field is reserved and always has the value zero.
22 Reserved	This read-only field is reserved and always has the value zero.
21–18 Reserved	This read-only field is reserved and always has the value zero.
17 ESDHC	ESDHC clock gate control 0 Clock is disabled. 1 Clock is enabled.
16 Reserved	This read-only field is reserved and always has the value zero.
15 SAI1	SAI1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
14 Reserved	This read-only field is reserved and always has the value zero.
13 Reserved	This read-only field is reserved and always has the value zero.
12 DSPI2	DSPI2 clock gate control 0 Clock is disabled. 1 Clock is enabled.
11–9 Reserved	This read-only field is reserved and always has the value zero.
8 NFC	NFC clock gate control 0 Clock is disabled. 1 Clock is enabled.
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 FLEXCAN1	FlexCAN1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
3–1 Reserved	This read-only field is reserved and always has the value zero.
0 RNGA	RNGA clock gate control 0 Clock is disabled. 1 Clock is enabled.

12.2.12 System Clock Gating Control Register 4 (SIM_SCGC4)

Address: SIM_SCGC4 is 4004_7000h base + 1034h offset = 4004_8034h

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	1	1	1	LLWU	0							VREF	CMP	USBFS	0				UART3	UART2	UART1	UART0	0		IIC1	IIC0	1	1	0	CMT	EWM	0
W																																
Reset	1	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

SIM_SCGC4 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value one.
30 Reserved	This read-only field is reserved and always has the value one.
29 Reserved	This read-only field is reserved and always has the value one.
28 LLWU	LLWU Clock Gate Control This bit controls the clock gate to the LLWU module. 0 Clock is disabled. 1 Clock is enabled.
27–21 Reserved	This read-only field is reserved and always has the value zero.
20 VREF	VREF clock gate control 0 Clock is disabled. 1 Clock is enabled.
19 CMP	Comparator clock gate control 0 Clock is disabled. 1 Clock is enabled.
18 USBFS	USB FS clock gate control 0 Clock is disabled. 1 Clock is enabled.
17–14 Reserved	This read-only field is reserved and always has the value zero.
13 UART3	UART3 clock gate control 0 Clock is disabled. 1 Clock is enabled.
12 UART2	UART2 clock gate control

Table continues on the next page...

SIM_SCGC4 field descriptions (continued)

Field	Description
	0 Clock is disabled. 1 Clock is enabled.
11 UART1	UART1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
10 UART0	UART0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
9–8 Reserved	This read-only field is reserved and always has the value zero.
7 IIC1	IIC1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
6 IIC0	IIC0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
5 Reserved	This read-only field is reserved and always has the value one.
4 Reserved	This read-only field is reserved and always has the value one.
3 Reserved	This read-only field is reserved and always has the value zero.
2 CMT	CMT clock gate control 0 Clock is disabled. 1 Clock is enabled.
1 EWM	EWM clock gate control 0 Clock is disabled. 1 Clock is enabled.
0 Reserved	This read-only field is reserved and always has the value zero.

12.2.13 System Clock Gating Control Register 5 (SIM_SCGC5)

Address: SIM_SCGC5 is 4004_7000h base + 1038h offset = 4004_8038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													1	0	
W																
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	0	PORTF	PORTE	PORTD	PORTC	PORTB	PORTA	1	1	0	TSI	0	0	0	REGFILE	LPTIMER
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

SIM_SCGC5 field descriptions

Field	Description
31–19 Reserved	This read-only field is reserved and always has the value zero.
18 Reserved	This read-only field is reserved and always has the value one.
17–15 Reserved	This read-only field is reserved and always has the value zero.
14 PORTF	PORTF clock gate control 0 Clock is disabled. 1 Clock is enabled.
13 PORTE	PORTE clock gate control 0 Clock is disabled. 1 Clock is enabled.
12 PORTD	PORTD clock gate control 0 Clock is disabled. 1 Clock is enabled.
11 PORTC	PORTC clock gate control 0 Clock is disabled. 1 Clock is enabled.
10 PORTB	PORTB clock gate control 0 Clock is disabled. 1 Clock is enabled.

Table continues on the next page...

SIM_SCGC5 field descriptions (continued)

Field	Description
9 PORTA	PORTA clock gate control 0 Clock is disabled. 1 Clock is enabled.
8 Reserved	This read-only field is reserved and always has the value one.
7 Reserved	This read-only field is reserved and always has the value one.
6 Reserved	This read-only field is reserved and always has the value zero.
5 TSI	TSI clock gate control 0 Clock is disabled. 1 Clock is enabled.
4 Reserved	This read-only field is reserved and always has the value zero.
3 Reserved	This read-only field is reserved and always has the value zero.
2 Reserved	This read-only field is reserved and always has the value zero.
1 REGFILE	Register File Clock Gate Control This bit controls the clock gate to the Register File module. 0 Clock is disabled. 1 Clock is enabled.
0 LPTIMER	LPTMR clock gate control 0 Clock is disabled. 1 Clock is enabled.

12.2.14 System Clock Gating Control Register 6 (SIM_SCGC6)

Address: SIM_SCGC6 is 4004_7000h base + 103Ch offset = 4004_803Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	1	RTC	ADC2	ADC0	0	FTM1	FTM0	PIT	PDB	USBDCD	USBHS	0	CRC	0	
W																
Reset	0	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	SAI0	0	DSPI1	DSPI0	0							FLEXCAN0	0	DMAMUX1	DMAMUX0	1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

SIM_SCGC6 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30 Reserved	This read-only field is reserved and always has the value one.
29 RTC	RTC clock gate control 0 Clock is disabled. 1 Clock is enabled.
28 ADC2	ADC2 clock gate control 0 Clock is disabled. 1 Clock is enabled.
27 ADC0	ADC0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
26 Reserved	This read-only field is reserved and always has the value zero.
25 FTM1	FTM1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
24 FTM0	FTM0 clock gate control 0 Clock is disabled. 1 Clock is enabled.

Table continues on the next page...

SIM_SCGC6 field descriptions (continued)

Field	Description
23 PIT	PIT clock gate control 0 Clock is disabled. 1 Clock is enabled.
22 PDB	PDB clock gate control 0 Clock is disabled. 1 Clock is enabled.
21 USBDCD	USB DCD clock gate control 0 Clock is disabled. 1 Clock is enabled.
20 USBHS	USBHS clock gate control 0 Clock is disabled. 1 Clock is enabled.
19 Reserved	This read-only field is reserved and always has the value zero.
18 CRC	CRC clock gate control 0 Clock is disabled. 1 Clock is enabled.
17–16 Reserved	This read-only field is reserved and always has the value zero.
15 SAI0	SAI0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
14 Reserved	This read-only field is reserved and always has the value zero.
13 DSPI1	DSPI1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
12 DSPI0	DSPI0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
11–5 Reserved	This read-only field is reserved and always has the value zero.
4 FLEXCAN0	FlexCAN0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
3 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

SIM_SCGC6 field descriptions (continued)

Field	Description
2 DMAMUX1	DMAMUX1 clock gate control 0 Clock is disabled. 1 Clock is enabled.
1 DMAMUX0	DMAMUX0 clock gate control 0 Clock is disabled. 1 Clock is enabled.
0 Reserved	This read-only field is reserved and always has the value one.

12.2.15 System Clock Gating Control Register 7 (SIM_SCGC7)

Address: SIM_SCGC7 is 4004_7000h base + 1040h offset = 4004_8040h

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												0	MPU	DMA	FLEXBUS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

SIM_SCGC7 field descriptions

Field	Description
31–4 Reserved	This read-only field is reserved and always has the value zero.
3 Reserved	This read-only field is reserved and always has the value zero.
2 MPU	MPU clock gate control 0 Clock is disabled. 1 Clock is enabled.
1 DMA	DMA controller clock gate control 0 Clock is disabled. 1 Clock is enabled.

Table continues on the next page...

SIM_SCGC7 field descriptions (continued)

Field	Description
0 FLEXBUS	FlexBus controller clock gate control 0 Clock is disabled. 1 Clock is enabled.

12.2.16 System Clock Divider Register 1 (SIM_CLKDIV1)

NOTE

The CLKDIV1 register cannot be written to when the device is in VLPR mode. The OUTDIV[1:4] bitfields reset only for power-on reset and are unaffected by other reset types.

Address: SIM_CLKDIV1 is 4004_7000h base + 1044h offset = 4004_8044h

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	OUTDIV1				OUTDIV2				OUTDIV3				OUTDIV4				0															
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

* Notes:
- x = Undefined at reset.

SIM_CLKDIV1 field descriptions

Field	Description
31–28 OUTDIV1	Clock 1 output divider value This field sets the divide value for the core/system clock. At the end of reset, it is loaded with either 0000 or 0111 depending on FTFE_FOPT[LPBOOT]. 0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14.

Table continues on the next page...

SIM_CLKDIV1 field descriptions (continued)

Field	Description
	1110 Divide-by-15. 1111 Divide-by-16.
27–24 OUTDIV2	Clock 2 output divider value This field sets the divide value for the peripheral clock. At the end of reset, it is loaded with either 0000 or 0111 depending on FTFE_FOPT[LPBOOT]. 0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14. 1110 Divide-by-15. 1111 Divide-by-16.
23–20 OUTDIV3	Clock 3 output divider value This field sets the divide value for the FlexBus clock driven to the external pin (FB_CLK). At the end of reset, it is loaded with either 0001 or 1111 depending on FTFE_FOPT[LPBOOT]. 0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14. 1110 Divide-by-15. 1111 Divide-by-16.
19–16 OUTDIV4	Clock 4 output divider value This field sets the divide value for the flash clock. At the end of reset, it is loaded with either 0001 or 1111 depending on FTFE_FOPT[LPBOOT].

Table continues on the next page...

SIM_CLKDIV1 field descriptions (continued)

Field	Description
	0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14. 1110 Divide-by-15. 1111 Divide-by-16.
15–0 Reserved	This read-only field is reserved and always has the value zero.

12.2.17 System Clock Divider Register 2 (SIM_CLKDIV2)

Address: SIM_CLKDIV2 is 4004_7000h base + 1048h offset = 4004_8048h

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				USBHSDIV			USBHSFRAC	0				USBFSDIV			USBFSFRAC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_CLKDIV2 field descriptions

Field	Description
31–12 Reserved	This read-only field is reserved and always has the value zero.
11–9 USBHSDIV	USB HS clock divider divisor

Table continues on the next page...

SIM_CLKDIV2 field descriptions (continued)

Field	Description
	This field sets the divide value for the fractional clock divider used as a USB clock source (SOPT2[USBHSRC] = 1). Divider output clock = Divider input clock × [(USBHSFRAC+1) / (USBHSDIV+1)]
8 USBHSFRAC	USB HS clock divider fraction This field sets the fraction multiply value for the fractional clock divider used as a USB clock source (SOPT2[USBHSRC] = 1). Divider output clock = Divider input clock × [(USBHSFRAC+1) / (USBHSDIV+1)]
7–4 Reserved	This read-only field is reserved and always has the value zero.
3–1 USBFSDIV	USB FS clock divider divisor This field sets the divide value for the fractional clock divider used as a USB clock source (SOPT2[USBFSRC] = 1). Divider output clock = Divider input clock × [(USBFSFRAC+1) / (USBFSDIV+1)]
0 USBFSFRAC	USB FS clock divider fraction This field sets the fraction multiply value for the fractional clock divider used as a USB clock source (SOPT2[USBFSRC] = 1). Divider output clock = Divider input clock × [(USBFSFRAC+1) / (USBFSDIV+1)]

12.2.18 Flash Configuration Register 1 (SIM_FCFG1)

Address: SIM_FCFG1 is 4004_7000h base + 104Ch offset = 4004_804Ch

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	NVMSIZE				PFSIZE				0				EESIZE				0				DEPART				0							FTFDIS
W																																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	0	0	0	0	x*	x*	x*	x*	0	0	0	0	x*	x*	x*	x*	0	0	0	0	0	0	0	0

* Notes:
- x = Undefined at reset.

SIM_FCFG1 field descriptions

Field	Description
31–28 NVMSIZE	FlexNVM size This field specifies the amount of FlexNVM memory available on the device . 0000 0 KB 0001 Reserved 0010 Reserved 0011 Reserved 0100 Reserved 0101 Reserved

Table continues on the next page...

SIM_FCFG1 field descriptions (continued)

Field	Description
	0110 Reserved 0111 Reserved 1000 Reserved 1001 Reserved 1010 Reserved 1011 512 KB, 16 KB protection region 1100 Reserved 1101 Reserved 1110 Reserved 1111 512 KB, 16 KB protection region
27–24 PFSIZE	Program flash size This field specifies the amount of program flash memory available on the device . 0000 Reserved 0001 Reserved 0010 Reserved 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 Reserved 1000 Reserved 1001 Reserved 1010 Reserved 1011 512 KB, 16 KB protection size 1100 Reserved 1101 1024 KB, 32 KB protection size 1110 Reserved 1111 1024 KB, 32 KB protection size
23–20 Reserved	This read-only field is reserved and always has the value zero.
19–16 EESIZE	EEPROM size EEPROM data size . 0000 16 KB 0001 8 KB 0010 4 KB 0011 2 KB 0100 1 KB 0101 512 Bytes 0110 256 Bytes 0111 128 Bytes 1000 64 Bytes 1001 32 Bytes

Table continues on the next page...

SIM_FCFG1 field descriptions (continued)

Field	Description
	1010-1110 Reserved 1111 0 Bytes
15–12 Reserved	This read-only field is reserved and always has the value zero.
11–8 DEPART	FlexNVM partition For devices with FlexNVM: Data flash / EEPROM backup split . See DEPART bit description in FTFE chapter. For devices without FlexNVM: Reserved
7–1 Reserved	This read-only field is reserved and always has the value zero.
0 FTFDIS	Disable FTFE

12.2.19 Flash Configuration Register 2 (SIM_FCFG2)

Address: SIM_FCFG2 is 4004_7000h base + 1050h offset = 4004_8050h

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	0	MAXADDR01						0	0	MAXADDR23						0															
W																																
Reset	0	0	x*	x*	x*	x*	x*	x*	0	0	x*	x*	x*	x*	x*	x*	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

* Notes:

- x = Undefined at reset.

SIM_FCFG2 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30 Reserved	This read-only field is reserved and always has the value zero.
29–24 MAXADDR01	Max address block 0 or 1 This field concatenated with 13 zeros indicates the number of 8 KB regions in logically interleaved program plash blocks 0 or 1.
23 Reserved	This read-only field is reserved and always has the value zero.
22 Reserved	This read-only field is reserved and always has the value zero.
21–16 MAXADDR23	Max address block 2 or 3 For devices with FlexNVM: This field concatenated with 13 zeros plus the FlexNVM base address indicates the number of 8 KB regions in logically interleaved FlexNVM blocks 2 or 3.

Table continues on the next page...

SIM_FCFG2 field descriptions (continued)

Field	Description
	For example, relative FlexNVM byte addresses below {MAXADDR23,13'b0} are valid for block 2 while byte addresses at or above {MAXADDR23,13'b0} and below {2*MAXADDR23,13'b0} are valid block 3. For devices with program flash only: This field concatenated with 13 zeros plus the value of the MAXADDR01 field indicates the number of 8 KB regions in logically interleaved program flash blocks 2 or 3. For example, relative P-Flash byte addresses at or above {2*MAXADDR01,13'b0} and below {2*MAXADDR01+MAXADDR23,13'b0} are valid for block 2 while byte addresses at or above {2*MAXADDR01+MAXADDR23,13'b0} and below {2*MAXADDR01+2*MAXADDR23,13'b0} are valid for block 3.
15–0 Reserved	This read-only field is reserved and always has the value zero.

12.2.20 Unique Identification Register High (SIM_UIDH)

Address: SIM_UIDH is 4004_7000h base + 1054h offset = 4004_8054h

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	UID																															
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.

SIM_UIDH field descriptions

Field	Description
31–0 UID	Unique Identification Unique identification for the device.

12.2.21 Unique Identification Register Mid-High (SIM_UIDMH)

Address: SIM_UIDMH is 4004_7000h base + 1058h offset = 4004_8058h

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	UID																															
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.

SIM_UIDMH field descriptions

Field	Description
31–0 UID	Unique Identification Unique identification for the device.

12.2.22 Unique Identification Register Mid Low (SIM_UIDML)

Address: SIM_UIDML is 4004_7000h base + 105Ch offset = 4004_805Ch

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	UID																															
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.

SIM_UIDML field descriptions

Field	Description
31–0 UID	Unique Identification Unique identification for the device.

12.2.23 Unique Identification Register Low (SIM_UIDL)

Address: SIM_UIDL is 4004_7000h base + 1060h offset = 4004_8060h

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	UID																															
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.

SIM_UIDL field descriptions

Field	Description
31–0 UID	Unique Identification Unique identification for the device.

12.2.24 System Clock Divider Register 4 (SIM_CLKDIV4)

Address: SIM_CLKDIV4 is 4004_7000h base + 1068h offset = 4004_8068h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NFCDIV					NFCFRAC			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												TRACEDIV			TRACEFRAC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

SIM_CLKDIV4 field descriptions

Field	Description
31–27 NFCDIV	NFC clock divider divisor This field sets the divide value for the fractional clock divider used as a source for NFC flash clock. The source clock for the fractional clock divider is set by the SOPT2[NFCSRC] bitfield. Divider output clock = Divider input clock x ((NFCFRAC+1)/(NFCDIV+1)) **NOTE:** The reciprocal of ((NFCFRAC+1)/(NFCDIV+1)) must be a multiple of 0.5. For example, NFCFRAC = 1 and NFCDIV = 2 is a valid setting, since the reciprocal is 1.5. However, NFCFRAC = 2 and NFCDIV=7 is not a valid setting, since the reciprocal is 2.6667.
26–24 NFCFRAC	NFC clock divider fraction This field sets the fraction multiply value for the fractional clock divider used as a source for NFC flash clock. The source clock for the fractional clock divider is set by the SOPT2[NFCSRC] bitfield. Divider output clock = Divider input clock x ((NFCFRAC+1)/(NFCDIV+1)) **NOTE:** The reciprocal of ((NFCFRAC+1)/(NFCDIV+1)) must be a multiple of 0.5. For example, NFCFRAC = 1 and NFCDIV = 2 is a valid setting, since the reciprocal is 1.5. However, NFCFRAC = 2 and NFCDIV=7 is not a valid setting, since the reciprocal is 2.6667.
23–4 Reserved	This read-only field is reserved and always has the value zero.
3–1 TRACEDIV	Trace clock divider divisor This field sets the divide value for the fractional clock divider used as a source for trace clock. The source clock for the fractional clock divider is set by the SOPT2 TRACECLKSEL register bit. Divider output clock = Divider input clock * ((TRACEFRAC+1)/(TRACEDIV+1))
0 TRACEFRAC	Trace clock divider fraction This field sets the divide value for the fractional clock divider used as a source for trace clock. The source clock for the fractional clock divider is set by the SOPT2 TRACECLKSEL register bit. Divider output clock = Divider input clock*((TRACEFRAC+1)/(TRACEDIV+1))

12.2.25 Misc Control Register (SIM_MCR)

Address: SIM_MCR is 4004_7000h base + 106Ch offset = 4004_806Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRACECLKDIS	ULPICLKOBE	PDBLOOP	0												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						0		Reserved				0			
W												0				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SIM_MCR field descriptions

Field	Description
31 TRACECLKDIS	Trace clock disable. 0 Enables trace clock. 1 Disable trace clock.
30 ULPICLKOBE	60 MHz ULPI clock (ULPI_CLK) output enable 0 Internal generated 60MHz ULPI clock is not output to the ULPI_CLK pin. 1 Interanl generated 60MHz ULPI clock provide clock for external ULPI phy.
29 PDBLOOP	PDB Loop Mode 0 Provides two seperated minor loop, loop for ADC0/1 and loop for ADC2/3D 1 Provides a loop to involve ADC0, ADC1, ADC2 and ADC3.
28–17 Reserved	This read-only field is reserved and always has the value zero.
16 Reserved	This read-only field is reserved and always has the value zero.
15–10 Reserved	This read-only field is reserved and always has the value zero.
9–8 Reserved	This read-only field is reserved and always has the value zero.
7–5 Reserved	This field is reserved.

Table continues on the next page...

SIM_MCR field descriptions (continued)

Field	Description
4 Reserved	This field is reserved. This reserved bit must remain cleared (set to 0).
3–0 Reserved	This read-only field is reserved and always has the value zero.

Chapter 13
Reset Control Module (RCM)

13.1 Introduction

This chapter describes the registers of the Reset Control Module (RCM). The RCM implements many of the reset functions for the chip. See the chip's reset chapter for more information.

13.2 Reset memory map and register descriptions

The reset control module (RCM) registers provide reset status information and reset filter control.

RCM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_F000	System Reset Status Register 0 (RCM_SRS0)	8	R	82h	13.2.1/331
4007_F001	System Reset Status Register 1 (RCM_SRS1)	8	R	00h	13.2.2/333
4007_F004	Reset Pin Filter Control Register (RCM_RPFC)	8	R/W	00h	13.2.3/334
4007_F005	Reset Pin Filter Width Register (RCM_RPFW)	8	R/W	00h	13.2.4/335
4007_F007	Mode Register (RCM_MR)	8	R	00h	13.2.5/337

13.2.1 System Reset Status Register 0 (RCM_SRS0)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x82

- LVD (without POR) — 0x02
- VLLS mode wakeup due to $\overline{\text{RESET}}$ pin assertion — 0x41
- VLLS mode wakeup due to other wakeup sources — 0x01
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: RCM_SRS0 is 4007_F000h base + 0h offset = 4007_F000h

Bit	7	6	5	4	3	2	1	0
Read	POR	PIN	WDOG	0		LOC	LVD	WAKEUP
Write								
Reset	1	0	0	0	0	0	1	0

RCM_SRS0 field descriptions

Field	Description
7 POR	Power-on reset Indicates a reset was caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold. 0 Reset not caused by POR 1 Reset caused by POR
6 PIN	External reset pin Indicates a reset was caused by an active-low level on the external $\overline{\text{RESET}}$pin. 0 Reset not caused by external reset pin 1 Reset caused by external reset pin
5 WDOG	Watchdog Indicates a reset was caused by the watchdog timer timing out. This reset source can be blocked by disabling the watchdog. 0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout
4–3 Reserved	This read-only field is reserved and always has the value zero.
2 LOC	Loss-of-clock reset Indicates a reset was caused by a loss of external clock. The MCG clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed MCG description for information on enabling the clock monitor. 0 Reset not caused by a loss of external clock. 1 Reset caused by a loss of external clock.
1 LVD	Low-voltage detect reset If the LVDRE bit is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This bit is also set by POR.

Table continues on the next page...

RCM_SRS0 field descriptions (continued)

Field	Description
	0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR
0 WAKEUP	Low leakage wakeup reset Indicates a reset was caused by an enabled LLWU module wakeup source while the chip was in a low leakage mode. In LLS mode, the $\overline{\text{RESET}}$ pin is the only wakeup source that can cause this reset. Any enabled wakeup source in a VLLSx mode causes a reset. This bit is cleared by any reset except WAKEUP. 0 Reset not caused by LLWU module wakeup source 1 Reset caused by LLWU module wakeup source

13.2.2 System Reset Status Register 1 (RCM_SRS1)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

NOTE

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x00
- LVD (without POR) — 0x00
- VLLS mode wakeup — 0x00
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: RCM_SRS1 is 4007_F000h base + 1h offset = 4007_F001h

Bit	7	6	5	4	3	2	1	0
Read	0	0	SACKERR	EZPT	MDM_AP	SW	LOCKUP	JTAG
Write								
Reset	0	0	0	0	0	0	0	0

RCM_SRS1 field descriptions

Field	Description
7 Reserved	This read-only field is reserved and always has the value zero.
6 Reserved	This read-only field is reserved and always has the value zero.
5 SACKERR	Stop Mode Acknowledge Error Reset Indicates a reset was caused, after an attempt to enter stop mode, by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode.

Table continues on the next page...

RCM_SRS1 field descriptions (continued)

Field	Description
	0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode
4 EZPT	EzPort Reset Indicates a reset was caused by EzPort receiving the RESET command while the device is in EzPort mode. 0 Reset not caused by EzPort receiving the RESET command while the device is in EzPort mode 1 Reset caused by EzPort receiving the RESET command while the device is in EzPort mode
3 MDM_AP	MDM-AP system reset request Indicates a reset was caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset not caused by host debugger system setting of the System Reset Request bit 1 Reset caused by host debugger system setting of the System Reset Request bit
2 SW	Software Indicates a reset was caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit
1 LOCKUP	Core Lockup Indicates a reset was caused by the ARM core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event
0 JTAG	JTAG generated reset Indicates a reset was caused by JTAG selection of certain IR codes (EZPORT, EXTEST, HIGHZ, and CLAMP). 0 Reset not caused by JTAG 1 Reset caused by JTAG

13.2.3 Reset Pin Filter Control Register (RCM_RPFC)

NOTE

The reset values of bits 2-0 are for Chip POR only. They are unaffected by other reset types.

NOTE

The bus clock filter is reset when disabled or when entering stop mode. The LPO filter is reset when disabled or when entering any low leakage stop mode.

Address: RCM_RPFC is 4007_F000h base + 4h offset = 4007_F004h

Bit	7	6	5	4	3	2	1	0
Read	0					RSTFLTSS	RSTFLTSRW	
Write								
Reset	0	0	0	0	0	0	0	0

RCM_RPFC field descriptions

Field	Description
7–3 Reserved	This read-only field is reserved and always has the value zero.
2 RSTFLTSS	Reset pin filter select in stop mode Selects how the reset pin filter is enabled in STOP and VLPS modes. 0 All filtering disabled 1 LPO clock filter enabled
1–0 RSTFLTSRW	Reset pin filter select in run and wait modes Selects how the reset pin filter is enabled in run and wait modes. 00 All filtering disabled 01 Bus clock filter enabled for normal operation 10 LPO clock filter enabled for normal operation 11 Reserved (all filtering disabled)

13.2.4 Reset Pin Filter Width Register (RCM_RPFW)

NOTE

The reset values of the bits in the RSTFLTSEL field are for Chip POR only. They are unaffected by other reset types.

Address: RCM_RPFW is 4007_F000h base + 5h offset = 4007_F005h

Bit	7	6	5	4	3	2	1	0
Read	0			RSTFLTSEL				
Write								
Reset	0	0	0	0	0	0	0	0

RCM_RPFW field descriptions

Field	Description
7–5 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

RCM_RPFW field descriptions (continued)

Field	Description
4–0 RSTFLTSEL	Reset pin filter bus clock select Selects the reset pin bus clock filter width. 00000 Bus clock filter count is 1 00001 Bus clock filter count is 2 00010 Bus clock filter count is 3 00011 Bus clock filter count is 4 00100 Bus clock filter count is 5 00101 Bus clock filter count is 6 00110 Bus clock filter count is 7 00111 Bus clock filter count is 8 01000 Bus clock filter count is 9 01001 Bus clock filter count is 10 01010 Bus clock filter count is 11 01011 Bus clock filter count is 12 01100 Bus clock filter count is 13 01101 Bus clock filter count is 14 01110 Bus clock filter count is 15 01111 Bus clock filter count is 16 10000 Bus clock filter count is 17 10001 Bus clock filter count is 18 10010 Bus clock filter count is 19 10011 Bus clock filter count is 20 10100 Bus clock filter count is 21 10101 Bus clock filter count is 22 10110 Bus clock filter count is 23 10111 Bus clock filter count is 24 11000 Bus clock filter count is 25 11001 Bus clock filter count is 26 11010 Bus clock filter count is 27 11011 Bus clock filter count is 28 11100 Bus clock filter count is 29 11101 Bus clock filter count is 30 11110 Bus clock filter count is 31 11111 Bus clock filter count is 32

13.2.5 Mode Register (RCM_MR)

This register includes read-only status flags to indicate the state of the mode pins during the last Chip Reset.

Address: RCM_MR is 4007_F000h base + 7h offset = 4007_F007h

Bit	7	6	5	4	3	2	1	0
Read	0						EZP_MS	0
Write								
Reset	0	0	0	0	0	0	0	0

RCM_MR field descriptions

Field	Description
7–2 Reserved	This read-only field is reserved and always has the value zero.
1 EZP_MS	EZP_MS_B pin state Reflects the state of the $\overline{\text{EZP\_MS}}$ pin during the last Chip Reset 0 Pin negated (logic 1) 1 Pin asserted (logic 0)
0 Reserved	This read-only field is reserved and always has the value zero.

Chapter 14
System Mode Controller

14.1 Introduction

NOTE

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

The System Mode Controller (SMC) is responsible for sequencing the system into and out of all low power stop and run modes. Specifically, it 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.

This chapter describes all the available low power modes, the sequence followed to enter/ exit each mode and the functionality available while in each of the modes.

The SMC is able to function during even the deepest low power modes.

14.2 Modes of Operation

The ARM CPU has three primary modes of operation: run, sleep, and deep sleep. The WFI or WFE instruction is used to invoke sleep and deep sleep modes. For Freescale microcontrollers, run, wait and stop are the common terminology used for the primary operating modes. The following table shows the translation between the ARM CPU modes and the Freescale MCU power modes.

ARM CPU mode	MCU mode
Sleep	Wait
Deep sleep	Stop

Accordingly, the ARM CPU documentation refers to sleep and deep sleep, while the Freescale MCU documentation normally uses wait and stop.

In addition, Freescale MCUs also augment stop, wait, and run modes in a number of ways. The power management controller (PMC) contains a run and a stop mode regulator. Run regulation is used in normal run, wait and stop modes. Stop mode regulation is used during all very low power and low leakage modes. During stop mode regulation the bus frequencies are limited for the very low power modes.

The SMC provides the user with multiple power options. The very low power run (VLPR) mode can drastically reduce run time power when maximum bus frequency is not required to handle the application needs. From normal run mode, the run mode (RUNM) bit field can be modified to change the the MCU into VLPR mode when limited frequency is sufficient for the application. From VLPR mode, a corresponding wait (VLPW) and stop (VLPS) mode can be entered.

Depending on the needs of the user application, a variety of stop modes are available that allow the state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. Several registers are used to configure the various modes of operation for the device.

The following table describes the power modes available for the device.

Table 14-1. Power modes

Mode	Description
RUN	MCU can be run at full speed and the internal supply is fully regulated (run regulation mode). This mode is also referred to as normal run mode.
WAIT	The Core Clock to the ARM Cortex-M4 core is shut off. The System Clock continues to operate; Bus Clocks, if enabled, continue to operate; and run regulation is maintained.
STOP	The Core Clock and System Clock to the ARM Cortex-M4 core are shut off. System Clock to other masters and Bus Clocks are stopped after all stop acknowledge signals from supporting peripherals are valid.
VLPR	The Core, System, Bus, and Flash Clock maximum frequencies are restricted in this mode. See the Power Management chapter for details about the maximum allowable frequencies.
VLPW	The Core Clock to the ARM Cortex-M4 core is shut off. The System, Bus, and Flash Clocks continue to operate, although their maximum frequency is restricted. See the Power Management chapter for details on what the maximum allowable frequencies are.
VLPS	The Core Clock and System Clock to the ARM Cortex-M4 core is shut off. System clock to other masters and Bus Clocks are stopped after all stop acknowledge signals from supporting peripherals are valid.
LLS	The Core Clock and System Clock to the ARM Cortex-M4 core is shut off. System clock and Bus Clocks are stopped after all stop acknowledge signals from supporting peripherals are valid. MCU is placed in a low leakage mode by reducing the voltage to internal logic. Internal logic states are retained.
VLLS3	The Core Clock and System Clock to the ARM Cortex-M4 core is shut off. System clock to other masters and Bus Clocks are stopped after all stop acknowledge signals from supporting peripherals are valid. MCU is placed in a low leakage mode by powering down the internal logic. All system RAM contents are retained and I/O states held. FlexRAM contents are not retained. Internal logic states are not retained.

Table continues on the next page...

Table 14-1. Power modes (continued)

Mode	Description
VLLS2	The Core Clock and System Clock to the ARM Cortex-M4 core is shut off. System clock to other masters and Bus Clocks are stopped after all stop acknowledge signals from supporting peripherals are valid.MCU is placed in a low leakage mode by powering down the internal logic and the system RAM2 partition. The system RAM1 partition contents are retained in this mode. FlexRAM contents are not retained. Internal logic states are not retained. [1]
VLLS1	In ARM architectures, Core Clock and System Clock to the ARM Cortex-M4 core is shut off. System clock to other masters and Bus Clocks are stopped after all stop acknowledge signals from supporting peripherals are valid. MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. A 32-byte register file (available in all modes) contents are retained and I/O states held. FlexRAM contents are not retained. Internal logic states are not retained.

[1] See the devices Chip Configuration details for the size and location of the system RAM partitions.

14.3 Memory Map and Register Descriptions

Details follow about the registers related to the System Mode Controller.

Different SMC 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.

SMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_E000	Power Mode Protection Register (SMC_PMPROT)	8	R/W	00h	14.3.1/341
4007_E001	Power Mode Control Register (SMC_PMCTRL)	8	R/W	00h	14.3.2/343
4007_E002	VLLS Control Register (SMC_VLLSCTRL)	8	R/W	03h	14.3.3/344
4007_E003	Power Mode Status Register (SMC_PMSTAT)	8	R	01h	14.3.4/345

14.3.1 Power Mode Protection Register (SMC_PMPROT)

This register provides protection for entry into any low power run or stop mode. The actual enabling of the low power 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 the MCU is configured for a disallowed or reserved power mode, the MCU remains in its current power mode. For example, if the MCU is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and the RUNM bits remain 00b, indicating the MCU is still in normal run mode.

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. See the Reset section details for more information.

Address: SMC_PMPROT is 4007_E000h base + 0h offset = 4007_E000h

Bit	7	6	5	4	3	2	1	0
Read	0		AVLP	0	ALLS	0	AVLLS	0
Write	▨	▨		▨		▨		▨
Reset	0	0	0	0	0	0	0	0

SMC_PMPROT field descriptions

Field	Description
7–6 Reserved	This read-only field is reserved and always has the value zero.
5 AVLP	Allow very low power modes Provided the appropriate control bits are set up in PMCTRL, this write-once bit allows the MCU to enter any very low power modes: VLPR, VLPW, and VLPS. 0 VLPR, VLPW and VLPS are not allowed 1 VLPR, VLPW and VLPS are allowed
4 Reserved	This read-only field is reserved and always has the value zero.
3 ALLS	Allow low leakage stop mode This write once bit allows the MCU to enter any low leakage stop mode (LLS) provided the appropriate control bits are set up in PMCTRL. 0 LLS is not allowed 1 LLS is allowed
2 Reserved	This read-only field is reserved and always has the value zero.
1 AVLLS	Allow very low leakage stop mode Provided the appropriate control bits are set up in PMCTRL, this write once bit allows the MCU to enter any very low leakage stop mode (VLLSx). 0 Any VLLSx mode is not allowed 1 Any VLLSx mode is allowed
0 Reserved	This read-only field is reserved and always has the value zero.

14.3.2 Power Mode Control Register (SMC_PMCTRL)

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 not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details. for more information.

Address: SMC_PMCTRL is 4007_E000h base + 1h offset = 4007_E001h

Bit	7	6	5	4	3	2	1	0
Read	LPWUI	RUNM		0	STOPA	STOPM		
Write								
Reset	0	0	0	0	0	0	0	0

SMC_PMCTRL field descriptions

Field	Description
7 LPWUI	Low Power Wake Up on Interrupt Causes the SMC to exit to normal RUN mode when any active MCU interrupt occurs while in a VLP mode (VLPR, VLPW or VLPS). **NOTE:** If VLPS mode was entered directly from RUN mode, the SMC will always exit back to normal RUN mode regardless of the LPWUI setting. **NOTE:** LPWUI should only be modified while the system is in RUN mode i.e. when PMSTAT=RUN. 0 The system remains in a VLP mode on an interrupt 1 The system exits to normal RUN mode on an interrupt
6–5 RUNM	Run Mode Control When written, this field 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. This field is cleared by hardware on any exit to normal RUN mode. **NOTE:** RUNM should only be set to VLPR when PMSTAT=RUN. Once written to VLPR, RUNM should not be written back to RUN until PMSTAT=VLPR. **NOTE:** RUNM should only be set to RUN when PMSTAT=VLPR. Once written to RUN, RUNM should not be written back to VLPR until PMSTAT=RUN. 00 Normal run mode (RUN) 01 Reserved 10 Very low power run mode (VLPR) 11 Reserved

Table continues on the next page...

SMC_PMCTRL field descriptions (continued)

Field	Description
4 Reserved	This read-only field is reserved and always has the value zero.
3 STOPA	Stop Aborted When set, this read-only status bit indicates an interrupt or reset occured during the previous stop mode entry sequence, preventing the system from entering that mode. This bit is cleared by hardware at the beginning of any stop mode entry sequence and is set if the sequence was aborted. 0 The previous stop mode entry was successsful. 1 The previous stop mode entry was aborted.
2–0 STOPM	Stop Mode Control When written, this field controls entry into the selected stop mode when sleep-now or sleep-on-exit mode is entered with SLEEPDEEP=1 . When this field is set to VLLSx, the VLLSCTRL register is used to further select the particular VLLS sub-mode which will be entered. 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. 000 Normal stop (STOP) 001 Reserved 010 Very low power stop (VLPS) 011 Low leakage stop (LLS) 100 Very low leakage stop (VLLSx) 101 Reserved 110 Reseved 111 Reserved

14.3.3 VLLS Control Register (SMC_VLLSCTRL)

The VLLSCTRL register selects which VLLSx mode is entered if STOPM=VLLS and controls power to FlexRAM during VLLS2.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details. for more information.

Address: SMC_VLLSCTRL is 4007_E000h base + 2h offset = 4007_E002h

Bit	7	6	5	4	3	2	1	0
Read	0		0	0	0	VLLSM		
Write								
Reset	0	0	0	0	0	0	1	1

SMC_VLLSCTRL field descriptions

Field	Description
7–6 Reserved	This read-only field is reserved and always has the value zero.
5 Reserved	This read-only field is reserved and always has the value zero.
4 Reserved	This read-only field is reserved and always has the value zero.
3 Reserved	This read-only field is reserved and always has the value zero.
2–0 VLLSM	VLLS Mode Control This field controls which VLLS sub-mode to enter if STOPM=VLLS. 000 Reserved 001 VLLS1 010 VLLS2 011 VLLS3 100 Reserved 101 Reserved 110 Reserved 111 Reserved

14.3.4 Power Mode Status Register (SMC_PMSTAT)

PMSTAT is a read-only, one-hot register which indicates the current power mode of the system.

NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details. for more information.

Address: SMC_PMSTAT is 4007_E000h base + 3h offset = 4007_E003h

Bit	7	6	5	4	3	2	1	0
Read	0	PMSTAT						
Write								
Reset	0	0	0	0	0	0	0	1

SMC_PMSTAT field descriptions

Field	Description
7 Reserved	This read-only field is reserved and always has the value zero.
6–0 PMSTAT	**NOTE:** When debug is enabled, the PMSTAT will not update to STOP or VLPS 000_0001 Current power mode is RUN 000_0010 Current power mode is STOP 000_0100 Current power mode is VLPR 000_1000 Current power mode is VLPW 001_0000 Current power mode is VLPS 010_0000 Current power mode is LLS 100_0000 Current power mode is VLLS

14.4 Functional Description

14.4.1 Power Mode Transitions

The following figure shows the power mode state transitions available on the chip. Any reset always brings the MCU back to the normal run state.

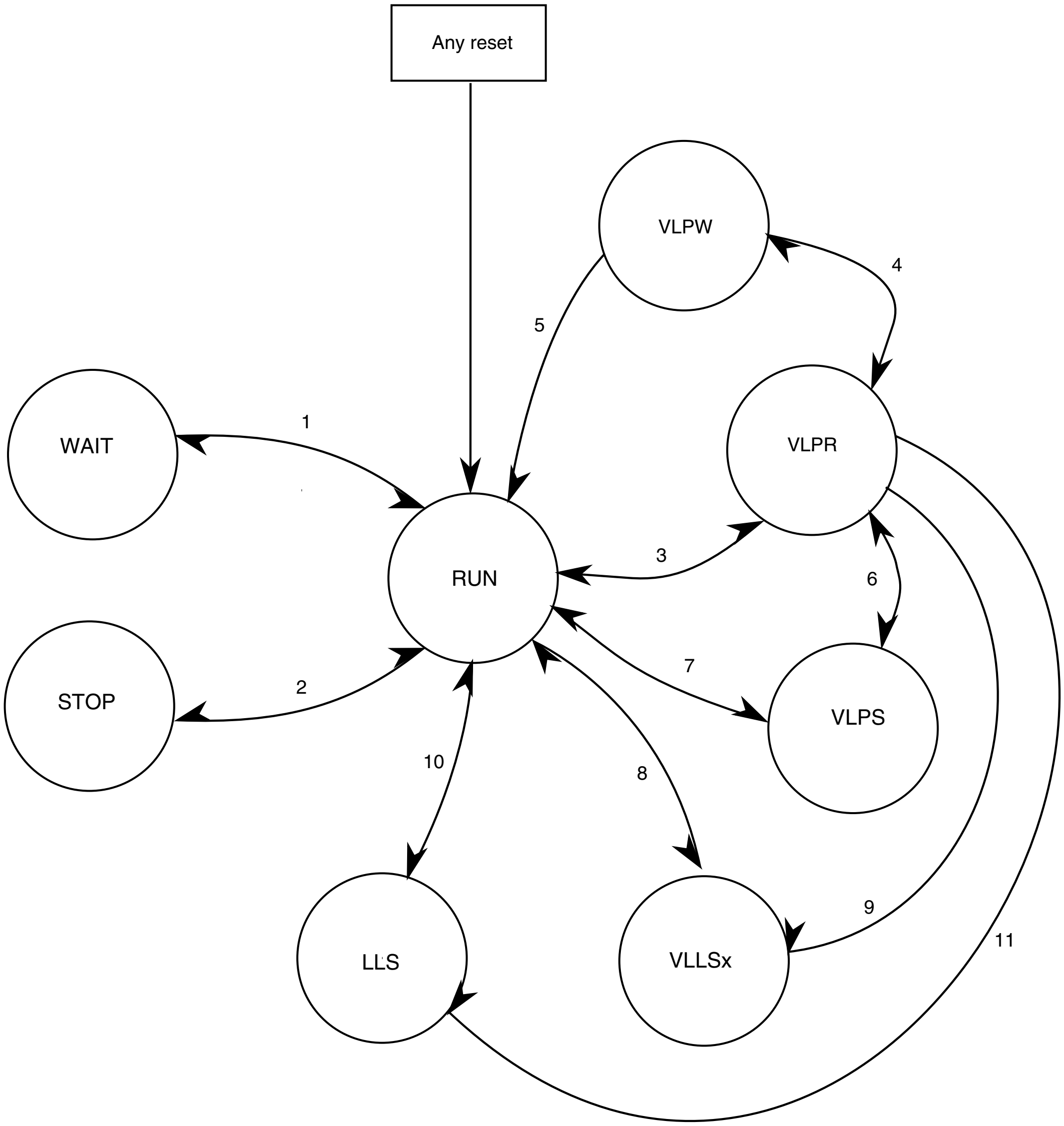


Figure 14-5. Power Mode State Diagram

The following table defines triggers for the various state transitions shown in the previous figure.

Table 14-7. Power mode transition triggers

Transition #	From	To	Trigger Conditions
1	RUN	WAIT	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, controlled in System Control Register in ARM core See note.[1]
	WAIT	RUN	Interrupt or Reset

Table continues on the next page...

Table 14-7. Power mode transition triggers (continued)

Transition #	From	To	Trigger Conditions
2	RUN	STOP	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, controlled in System Control Register in ARM core See note.[1]
	STOP	RUN	Interrupt or Reset
3	RUN	VLPR	Reduce system, bus and core frequency to 2 MHz or less, Flash access limited to 1MHz. Set PMPROT[AVLP]=1, PMCTRL[RUNM]=10.
	VLPR	RUN	Set PMCTRL[RUNM]=00 or Interrupt with PMCTRL[LPWUI] =1 or Reset.
4	VLPR	VLPW	Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, controlled in System Control Register in ARM core See note.[1]
	VLPW	VLPR	Interrupt with PMCTRL[LPWUI]=0
5	VLPW	RUN	Interrupt with PMCTRL[LPWUI]=1 or Reset
6	VLPR	VLPS	PMCTRL[STOPM]=000 or 010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, controlled in System Control Register in ARM core See note.[1]
	VLPS	VLPR	Interrupt with PMCTRL[LPWUI]=0 **NOTE:** If VLPS was entered directly from RUN, hardware will not allow this transition and will force exit back to RUN
7	RUN	VLPS	PMPROT[AVLP]=1, PMCTRL[STOPM]=010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, controlled in System Control Register in ARM core See note.[1]
	VLPS	RUN	Interrupt with PMCTRL[LPWUI]=1 or Interrupt with PMCTRL[LPWUI]=0 and VLPS mode was entered directly from RUN or Reset
8	RUN	VLLSx	PMPROT[AVLLS]=1, PMCTRL[STOPM]=100, VLLSCTRL[VLLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, controlled in System Control Register in ARM core
	VLLSx	RUN	Wakeup from enabled LLWU input source or RESET pin
9	VLPR	VLLSx	PMPROT[AVLLS]=1, PMCTRL[STOPM]=100, VLLSCTRL[VLLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, controlled in System Control Register in ARM core

Table continues on the next page...

Table 14-7. Power mode transition triggers (continued)

Transition #	From	To	Trigger Conditions
10	RUN	LLS	PMPROT[ALLS]=1, PMCTRL[STOPM]=011, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, controlled in System Control Register in ARM core
	LLS	RUN	Wakeup from enabled LLWU input source or RESET pin
11	VLPR	LLS	PMPROT[ALLS]=1, PMCTRL[STOPM]=011, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, controlled in System Control Register in ARM core

1. If debug is enabled, the core clock remains to support debug.

14.4.2 Power Mode Entry/Exit Sequencing

When entering or exiting low power modes, the system must conform to an orderly sequence to manage transitions safely. The SMC manages the system's entry to and exit from all power modes. The following diagram illustrates the connections of the SMC with other system components in the chip that are necessary to sequence the system through all power modes.

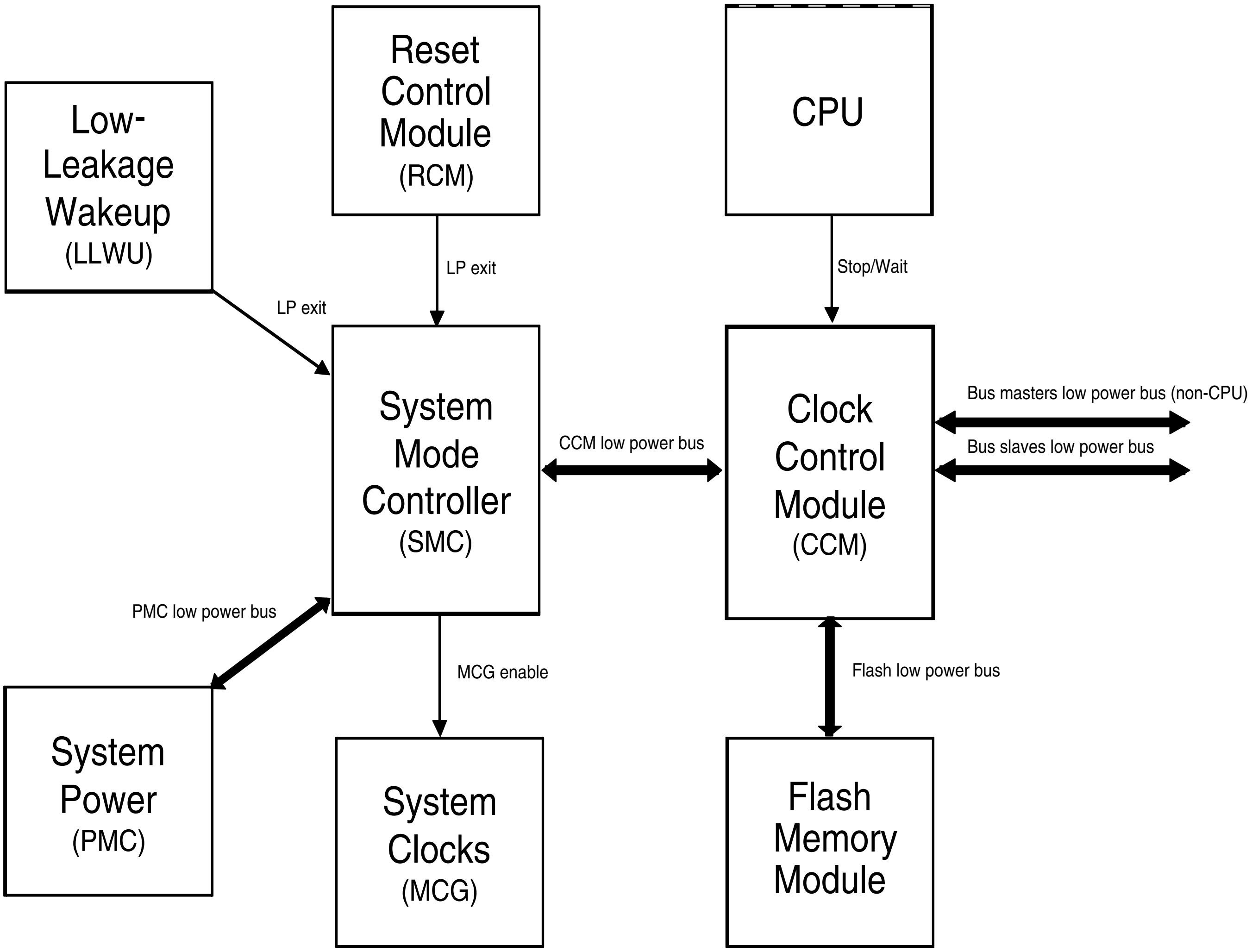


Figure 14-6. Low power system components and connections

14.4.2.1 Stop Mode Entry Sequence

Entry to a low power stop mode (STOP, VLPS, LLS, VLLSx) is initiated by CPU execution of the WFI instruction. After the instruction is executed, the following sequence occurs:

1. The CPU clock is gated off immediately.
2. Requests are made to all non-CPU bus masters to enter stop mode.
3. After all masters have acknowledged they are ready to enter stop mode, requests are made to all bus slaves to enter stop mode.
4. After all slaves have acknowledged they are ready to enter stop mode, all system and bus clocks are gated off.
5. Clock generators are disabled in the MCG.
6. The on-chip regulator in the PMC and internal power switches are configured to meet the power consumption goals for the targeted low power mode.

14.4.2.2 Stop Mode Exit Sequence

Exit from a low power stop mode is initiated by either a reset or an interrupt event. The following sequence then executes to restore the system to a run mode (RUN or VLPR):

1. The on-chip regulator in the PMC and internal power switches are restored.
2. Clock generators are enabled in the MCG.
3. System and bus clocks are enabled to all masters and slaves.
4. The CPU clock is enabled and the CPU begins servicing the reset or interrupt that initiated the exit from the low power stop mode.

14.4.2.3 Aborted Stop Mode Entry

If an interrupt or a reset occurs during a stop entry sequence, the SMC can abort the transition early and return to RUN mode without completely entering the stop mode. An aborted entry is only possible if the reset or interrupt occurs before the PMC begins the transition to stop mode regulation. After this point, the interrupt or reset is ignored until the PMC has completed its transition to stop mode regulation. When an aborted stop mode entry sequence occurs, the SMC's PMCTRL[STOPA] bit is set to 1.

Restriction

> Aborted entry to a stop mode is not supported when an interrupt occurs during a transition from VLPR mode to any stop mode.

14.4.2.4 Transition to Wait Modes

For wait modes (WAIT and VLPW), the CPU clock is gated off with all other clocking continuing, as in RUN and VLPR mode operation. Some modules that support stop-in-wait functionality have their clocks disabled in these configurations.

14.4.2.5 Transition from Stop Modes to Debug Mode

The debugger module supports a transition from STOP, WAIT, VLPS, and VLPW back to a Halted state when the debugger has been enabled (ENBDM is 1). As part of this transition, system clocking is re-established and is equivalent to the normal RUN and VLPR mode clocking configuration.

14.4.3 Run Modes

The device contains two different run modes:

- Run
- Very low power run (VLPR)

14.4.3.1 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, disable unused modules by clearing the peripherals corresponding clock gating control bit in the SIM's registers.

14.4.3.2 Very Low Power Run (VLPR) Mode

In VLPR mode, the on-chip voltage regulator is put into a stop mode regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules using the peripherals' corresponding clock gating control bits in the SIM's registers.

Before entering this mode, the following conditions must be met:

- The MCG must be configured in a mode which is supported during VLPR. See the Power Management details for information about these MCG modes.
- 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 (PMPROT[AVLP] is 1).
- PMCTRL[RUNM] is set to 10b to enter VLPR.
- Flash programming/erasing is not allowed.

While in VLPR mode, the regulator is slow responding and cannot manage fast load transitions. Therefore, do not change the clock frequency. In addition, do not modify the clock source or BDIV in the MCG module, the module clock enables in the SIM, or any clock divider registers.

To re-enter normal run mode, simply clear RUNM. The PMSTAT register 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 is in run regulation mode and the MCU can run at full speed in any clock mode. If a higher execution frequency is desired, poll the PMSTAT register until it is set to RUN when returning from VLPR mode.

VLPR mode also provides the option to return to run regulation if any interrupt occurs. Implement this option by setting the low power wakeup on interrupt (LPWUI) bit in the PMCTRL register. Any reset always causes an exit from VLPR and returns the device to RUN mode after the MCU exits its reset flow. The RUNM bits are cleared by hardware on any interrupt when LPWUI is set or on any reset.

14.4.4 Wait Modes

This device contains two different wait modes:

- Wait
- Very low power wait (VLPW)

14.4.4.1 WAIT Mode

WAIT mode is entered when the ARM core enters the "sleep-now" or "sleep-on-exit" modes while SLEEDEEP is cleared. The ARM CPU enters a low-power state in which it is not clocked, but peripherals continue to be clocked provided they are enabled and clock gating to the peripheral is enabled via the SIM.

When an interrupt request occurs, the CPU exits WAIT mode and resumes processing in RUN mode, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from WAIT mode, returning the device to normal RUN mode.

14.4.4.2 Very Low Power Wait (VLPW) Mode

VLPW is entered by the entering the “sleep-now" or "sleep-on-exit” mode while SLEEPDEEP is cleared and the MCU is in VLPR mode.

In VLPW, the on-chip voltage regulator remains in its stop regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules by clearing the peripherals' corresponding clock gating control bits in the SIM.

VLPR mode restrictions also apply to VLPW.

VLPW mode provides the option to return to full-regulated normal RUN mode if any enabled interrupt occurs. This is done by setting the low power wake up on interrupt (LPWUI) bit in the PMCTRL register. Wait for the PMSTAT register to set to RUN before increasing the frequency.

If the LPWUI bit is clear, when an interrupt from VLPW occurs, the device returns to VLPR mode to execute the interrupt service routine.

A system reset will cause an exit from WAIT mode, returning the device to normal RUN mode.

14.4.5 Stop Modes

This device contains a variety of stop modes to meet your application needs. The stop modes range from:

- a stopped CPU, with all I/O, logic and memory states retained, certain asynchronous mode peripherals operating

to:

- a powered down CPU, with only I/O and a small register file retained, very few asynchronous mode peripherals operating, while the remainder of the MCU is powered down

The choice of stop mode depends upon the user's application, where power usage and state retention versus functional needs are traded off.

The various stop modes are selected by setting the appropriate bits in the power mode protection (PMPROT) and power mode control (PMCTRL) registers. The selected stop mode mode is entered during the sleep-now or sleep-on-exit entry with with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The available stop modes are:

- Normal stop (STOP)
- Very low power stop (VLPS)
- Low leakage stop (LLS)
- Very low leakage stop (VLLSx)

14.4.5.1 STOP Mode

STOP mode is entered via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The MCG module can be configured to leave the reference clocks running.

A module capable of providing an asynchronous interrupt to the device takes the device out of STOP mode and returns the device to normal RUN mode. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in STOP mode. When an interrupt request occurs, the CPU exits STOP mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from STOP mode, returning the device to normal RUN mode via a MCU reset.

14.4.5.2 Very Low Power Stop (VLPS) Mode

VLPS mode can be entered in one of two ways:

- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in VLPR mode and STOPM=010 or 000 in the PMCTRL register.
- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in normal RUN mode and STOPM=010 in the PMCTRL register. Note, when VLPS is entered directly from RUN mode, exit to VLPR is disabled by hardware and the system will always exit back to RUN.

In VLPS, the on-chip voltage regulator remains in its stop regulation state as in VLPR.

A module capable of providing an asynchronous interrupt to the device takes the device out of VLPS and returns the device to VLPR mode provided the LPWUI bit is clear.

If LPWUI is set, the device returns to normal RUN mode upon an interrupt request. The PMSTAT register must be set to RUN before allowing the system to return to a frequency higher than that allowed in VLPR mode.

A system reset will also cause a VLPS exit, returning the device to normal RUN mode.

14.4.5.3 Low-Leakage Stop (LLS) Mode

Low leakage stop (LLS) mode can be entered from normal RUN or VLPR modes.

The MCU enters LLS mode if:

- In sleep-now or sleep-on-exit mode, the SLEEPDEEP bit is set in the System Control Register in the ARM core, and
- The device is configured as per Table 14-7.

In LLS, the on-chip voltage regulator is in stop regulation. Most of the peripherals are put in a state-retention mode that does not allow them to operate while in LLS.

Before entering LLS mode, user should configure the low leakage wake up (LLWU) module to enable the desired wakeup sources. The available wakeup sources in LLS are detailed in the Chip Configuration details for this device.

After wakeup from LLS, the device returns to normal RUN mode with a pending LLWU module interrupt. In the LLWU interrupt service routine (ISR), user can poll the LLWU module wakeup flags to determine the source of the wakeup.

NOTE

> The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery.

An asserted $\overline{\text{RESET}}$ pin will cause an exit from LLS mode returning the device to normal RUN mode. When LLS is exiting via the $\overline{\text{RESET}}$ pin, the PIN and WAKEUP bits are set in the SRS0 register of the reset control module (RCM).

14.4.5.4 Very Low-Leakage Stop (VLLSx) Modes

This device contains three very low leakage modes: VLLS3, VLLS2, and VLLS1. VLLSx is often used in this document to refer to all three VLLS3, VLLS2 and VLLS1 modes.

All VLLSx modes can be entered from normal RUN or VLPR modes.

The MCU enters the configured VLLS mode if:

- In sleep-now or sleep-on-exit mode, the SLEEPDEEP bit is set in the System Control Register in the ARM core, and
- The device is configured as per Table 14-7.

In VLLS, the on-chip voltage regulator is in its stop-regulation state while most digital logic is powered off.

In VLLS, configure the LLWU module to enable the desired wakeup sources. The available wakeup sources in VLLS are detailed LLWU's Chip Configuration details for this device.

When entering VLLS, each I/O pin is latched as configured before executing VLLS. Since all digital logic in the MCU is powered off, all port and peripheral data is lost during VLLS. This information must be restored before ACKISO bit in the PMC is set.

An asserted $\overline{\text{RESET}}$ pin will cause an exit from any VLLS mode returning the device to normal RUN mode. When exiting VLLS via the $\overline{\text{RESET}}$ pin, the PIN and WAKEUP bits are set in the SRS0 register of the reset control module (RCM).

14.4.6 Debug in Low Power Modes

When the MCU is secure the device disables/limits debugger operation. When the MCU is unsecure, the ARM debugger can assert two power-up request signals:

- System power up (SYSPWR bit in the Debug Port Control/Stat register)
- Debug power up (CDBGPWRUPREQ bit in the Debug Port Control/Stat register)

When asserted while in RUN, WAIT, VLPR, or VLPW, the Mode Controller drives a corresponding acknowledge for each signal (CDBGPWRUPACK, CSYSPWRUPACK). When both requests are asserted, the Mode Controller handles attempts to enter STOP and VLPS by entering an emulated stop state. In this emulated stop state:

- The regulator is in run regulation,
- The MCG-generated clock source is enabled,
- All system clocks, except core clock, are disabled,
- The debug module has access to core registers, and
- Access to the on-chip peripherals is blocked.

No debug is available while the MCU is in LLS or VLLS modes. LLS is a state-retention mode and all debug operation can continue after waking from LLS, even in cases where system wakeup is due to a system reset event.

Entering into a VLLS mode causes all the debug controls and settings to be powered off. To give time to the debugger to sync with the MCU, the MDM AP Control Register includes a Very Low Leakage Debug Request (VLLDBGREQ) bit that is set to configure the Reset Controller logic to hold the system in reset after the next recovery from a VLLS mode. This bit allows the debugger time to re-initialize the debug module before the debug session continues.

The MDM AP Control Register also includes a Very Low Leakage Debug Acknowledge (VLLDBGACK) bit that is set to release the ARM core being held in reset following a VLLS recovery. The debugger re-initializes all debug IP and then asserts the VLLDBGACK control bit to allow the RCM to release the ARM core from reset and allow CPU operation to begin.

The VLLDBGACK bit is cleared by the debugger (or can be left set as is) or clears automatically due to the reset generated as part of the next VLLS recovery.

Chapter 15
Power Management Controller

15.1 Introduction

NOTE

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

The PMC contains the internal voltage regulator, power on reset (POR), and low voltage detect system.

15.2 Features

- Internal voltage regulator
- Active POR providing brown-out detect
- Low-voltage detect supporting two low-voltage trip points with four warning levels per trip point

15.3 Low-Voltage Detect (LVD) System

This device includes a system to protect against low-voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system is comprised of a power-on reset (POR) circuit and a LVD circuit with a user-selectable trip voltage: high (V_{LVDH}) or low (V_{LVDL}). The trip voltage is selected by the LVDSC1[LVDV] bits. The LVD is disabled upon entering VLPx, LLS, and VLLSx modes.

Two flags are available to indicate the status of the low-voltage detect system:

- The low voltage detect flag (LVDF) operates in a level sensitive manner. The LVDF bit is set when the supply voltage falls below the selected trip point (VLVD). The

LVDF bit is cleared by writing one to the LVDACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVDF bit remains set.

- The low voltage warning flag (LVWF) operates in a level sensitive manner. The LVWF bit is set when the supply voltage falls below the selected monitor trip point (VLVW). The LVWF bit is cleared by writing one to the LVWACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVWF bit remains set.

15.3.1 LVD Reset Operation

By setting the LVDRE bit, the LVD generates a reset upon detection of a low voltage condition. The low voltage detection threshold is determined by the LVDV bits. After an LVD reset occurs, the LVD system holds the MCU in reset until the supply voltage rises above this threshold. The LVD bit in the SRS register is set following an LVD or power-on reset.

15.3.2 LVD Interrupt Operation

By configuring the LVD circuit for interrupt operation (LVDIE set and LVDRE clear), LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. The LVDF bit is cleared by writing one to the LVDSC1[LVDACK] bit.

15.3.3 Low-Voltage Warning (LVW) Interrupt Operation

The LVD system contains a low voltage warning flag (LVWF) to indicate that the supply voltage is approaching, but is above, the LVD voltage. The LVW also has an interrupt, which is enabled by setting the LVDSC2[LVWIE] bit. If enabled, an LVW interrupt request occurs when the LVWF is set. LVWF is cleared by writing one to the LVDSC2[LVWACK] bit.

The LVDSC2[LVWV] bits select one of four trip voltages:

- Highest (V_{LVW4})
- Two mid-levels (V_{LVW3} and V_{LVW2})
- Lowest (V_{LVW1})

15.4 I/O Retention

When in LLS mode, the I/O pins are held in their input or output state. Upon wakeup, the power management control (PMC) is re-enabled, goes through a power up sequence to full regulation, and releases the logic from state retention mode. The I/O are released immediately after a wakeup or reset event. In the case of LLS exit via a RESET pin, the I/O default to their reset state.

When in VLLS modes, the I/O states are held on a wakeup event (with the exception of wakeup by reset event) until the wakeup has been acknowledged via a write to the ACKISO bit. In the case of VLLS exit via a RESET pin, the I/O are released and default to their reset state. In this case, no write to the ACKISO is needed.

15.5 Memory Map and Register Descriptions

PMC register details follow.

NOTE

Different portions of PMC registers are reset only by particular 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.

PMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_D000	Low Voltage Detect Status and Control 1 Register (PMC_LVDSC1)	8	R/W	10h	15.5.1/362
4007_D001	Low Voltage Detect Status and Control 2 Register (PMC_LVDSC2)	8	R/W	00h	15.5.2/363
4007_D002	Regulator Status and Control Register (PMC_REGSC)	8	R/W	04h	15.5.3/364

15.5.1 Low Voltage Detect Status and Control 1 Register (PMC_LVDSC1)

This register contains status and control bits to support the low voltage detect function. This register should be written during the reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC1 settings. To protect systems that must have LVD always on, configure the SMC's power mode protection register (PMPROT) to disallow any very low power or low leakage modes from being enabled.

See the device's data sheet for the exact LVD trip voltages.

NOTE

> The LVDV bits are reset solely on a POR Only event. The register's other bits are reset on Chip Reset not VLLS. For more information about these reset types, refer to the Reset section details.

Address: PMC_LVDSC1 is 4007_D000h base + 0h offset = 4007_D000h

Bit	7	6	5	4	3	2	1	0
Read	LVDF	0	LVDIE	LVDRE	0		LVDV	
Write		LVDACK						
Reset	0	0	0	1	0	0	0	0

PMC_LVDSC1 field descriptions

Field	Description
7 LVDF	Low-Voltage Detect Flag This read-only status bit indicates a low-voltage detect event. 0 Low-voltage event not detected 1 Low-voltage event detected
6 LVDACK	Low-Voltage Detect Acknowledge This write-only bit is used to acknowledge low voltage detection errors (write 1 to clear LVDF). Reads always return 0.
5 LVDIE	Low-Voltage Detect Interrupt Enable Enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVDF = 1.

Table continues on the next page...

PMC_LVDSC1 field descriptions (continued)

Field	Description
4 LVDRE	Low-Voltage Detect Reset Enable This write-once bit enables LVDF events to generate a hardware reset. Additional writes are ignored. 0 LVDF does not generate hardware resets 1 Force an MCU reset when LVDF = 1
3–2 Reserved	This read-only field is reserved and always has the value zero.
1–0 LVDV	Low-Voltage Detect Voltage Select Selects the LVD trip point voltage (V_{LVD}). 00 Low trip point selected ($V_{LVD} = V_{LVDL}$) 01 High trip point selected ($V_{LVD} = V_{LVDH}$) 10 Reserved 11 Reserved

15.5.2 Low Voltage Detect Status and Control 2 Register (PMC_LVDSC2)

This register contains status and control bits to support the low voltage warning function.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC2 settings.

See the device's data sheet for the exact LVD trip voltages.

NOTE

The LVW trip voltages depend on LVWV and LVDV bits.

NOTE

The LVWV bits are reset solely on a POR Only event. The register's other bits are reset on Chip Reset not VLLS. For more information about these reset types, refer to the Reset section details.

Address: PMC_LVDSC2 is 4007_D000h base + 1h offset = 4007_D001h

Bit	7	6	5	4	3	2	1	0
Read	LVWF	0	LVWIE	0			LVWV	
Write		LVWACK						
Reset	0	0	0	0	0	0	0	0

PMC_LVDSC2 field descriptions

Field	Description
7 LVWF	Low-Voltage Warning Flag This read-only status bit indicates a low-voltage warning event. LVWF is set when V_{Supply} transitions below the trip point or after reset and V_{Supply} is already below V_{LVW} . 0 Low-voltage warning event not detected 1 Low-voltage warning event detected
6 LVWACK	Low-Voltage Warning Acknowledge This write-only bit is used to acknowledge low voltage warning errors (write 1 to clear LVWF). Reads always return 0.
5 LVWIE	Low-Voltage Warning Interrupt Enable Enables hardware interrupt requests for LVWF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVWF = 1.
4–2 Reserved	This read-only field is reserved and always has the value zero.
1–0 LVWV	Low-Voltage Warning Voltage Select Selects the LVW trip point voltage (V_{LVW}). The actual voltage for the warning depends on LVDSC1[LVDV]. 00 Low trip point selected ($V_{LVW} = V_{LVW1}$) 01 Mid 1 trip point selected ($V_{LVW} = V_{LVW2}$) 10 Mid 2 trip point selected ($V_{LVW} = V_{LVW3}$) 11 High trip point selected ($V_{LVW} = V_{LVW4}$)

15.5.3 Regulator Status and Control Register (PMC_REGSC)

The power management controller contains an internal voltage regulator. The voltage regulator design uses a bandgap reference that is also available through a buffer as input to certain internal peripherals, such as the CMP and ADC. The internal regulator provides a status bit (REGONS) indicating the regulator is in run regulation.

NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. See the Reset section for more information.

Address: PMC_REGSC is 4007_D000h base + 2h offset = 4007_D002h

Bit	7	6	5	4	3	2	1	0
Read	0				ACKISO	REGONS	Reserved	BGBE
Write					w1c			
Reset	0	0	0	0	0	1	0	0

PMC_REGSC field descriptions

Field	Description
7–4 Reserved	This read-only field is reserved and always has the value zero.
3 ACKISO	Acknowledge Isolation Reading this bit indicates whether certain peripherals and the I/O pads are in a latched state as a result of having been in a VLLS mode. Writing one to this bit when it is set releases the I/O pads and certain peripherals to their normal run mode state. **NOTE:** After recovering from a VLLS mode, user should restore chip configuration before clearing ACKISO. In particular, pin configuration for enabled LLWU wakeup pins should be restored to avoid any LLWU flag from being falsely set when ACKISO is cleared. 0 Peripherals and I/O pads are in normal run state 1 Certain peripherals and I/O pads are in an isolated and latched state
2 REGONS	Regulator in Run Regulation Status This read-only bit provides the current status of the internal voltage regulator. 0 Regulator is in stop regulation or in transition to/from it 1 Regulator is in run regulation
1 Reserved	This field is reserved. **NOTE:** This reserved bit must remain cleared (set to 0).
0 BGBE	Bandgap Buffer Enable Enables the bandgap buffer. 0 Bandgap buffer not enabled 1 Bandgap buffer enabled

Chapter 16
Low-Leakage Wake-up Unit (LLWU)

16.1 Introduction

NOTE

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

The LLWU module allows the user to select up to 16 external pin sources and up to 8 internal modules as a wakeup source from low-leakage power modes. The input sources are described in the device's Chip Configuration details. Each of the available wakeup sources can be individually enabled.

The $\overline{\text{RESET}}$ 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. The LLWU_RST[LLRSTE] bit must be set to allow an exit from low-leakage modes via the $\overline{\text{RESET}}$ pin. On a device where the $\overline{\text{RESET}}$ pin is shared with other functions, the explicit port mux control register must be set for the $\overline{\text{RESET}}$ pin before the $\overline{\text{RESET}}$ pin can be used as a low-leakage reset source.

The LLWU module also includes three optional digital pin filters: two for the external wakeup pins and one for the $\overline{\text{RESET}}$ pin.

16.1.1 Features

The LLWU module features include:

- Supports up to 16 external input pins and up to 8 internal modules with individual enable bits
- Input sources may be external pins or from internal peripherals capable of running in LLS or VLLS. See the Chip Configuration information for wakeup input sources for this device.

- Each external pin wakeup input is programmable as falling edge, rising edge, or any change
- Wakeup inputs are activated if enabled once MCU enters a low-leakage power mode
- Optional digital filters provided to qualify an external pin detect and $\overline{\text{RESET}}$ pin detect.

16.1.2 Modes of operation

The LLWU module becomes functional on entry into a low-leakage power mode. After recovery from LLS, the LLWU is then immediately disabled. After recovery from VLLS, the LLWU continues to detect wakeup events until the user has acknowledged the wakeup via a write to the PMC_REGSC[ACKISO] bit.

16.1.2.1 LLS mode

The LLWU module provides up to 16 external wakeup inputs and up to 8 internal module wakeup inputs. In addition, an LLS reset event can be initiated via assertion of the $\overline{\text{RESET}}$ pin.

Wakeup events due to external wakeup inputs and internal module wakeup inputs result in an interrupt flow when exiting LLS. A reset event due to $\overline{\text{RESET}}$ pin assertion results in a reset flow when exiting LLS.

NOTE

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery.

16.1.2.2 VLLS modes

The LLWU module provides up to 16 external wakeup inputs and up to 8 internal module wakeup inputs. In addition, a VLLS reset event can be initiated via assertion of the $\overline{\text{RESET}}$ pin. All wakeup and reset events result in VLLS exit via a reset flow.

16.1.2.3 Non-low leakage modes

The LLWU is not active in all non-low leakage modes where detection and control logic are in a static state. The LLWU registers are accessible in non-low leakage modes and are available for configuring and reading status when bus transactions are possible.

When the$\overline{\text{RESET}}$ pin filter or wakeup pin filters are enabled, filter operation begins immediately. If a low leakage mode is entered within 5 LPO clock cycles of an active edge, the edge event will be detected by the LLWU. For $\overline{\text{RESET}}$ pin filtering, this means there is no restart to the minimum LPO cycle duration as the filtering transitions from a non-low leakage filter (implemented in the RCM) to the LLWU filter.

16.1.2.4 Debug mode

When the chip is in debug mode and then enters LLS or a VLLSx mode, no debug logic works in the fully functional low leakage mode. Upon an exit from the LLS or VLLSx mode, the LLWU becomes inactive.

16.1.3 Block diagram

The following figure is the block diagram for the LLWU module.

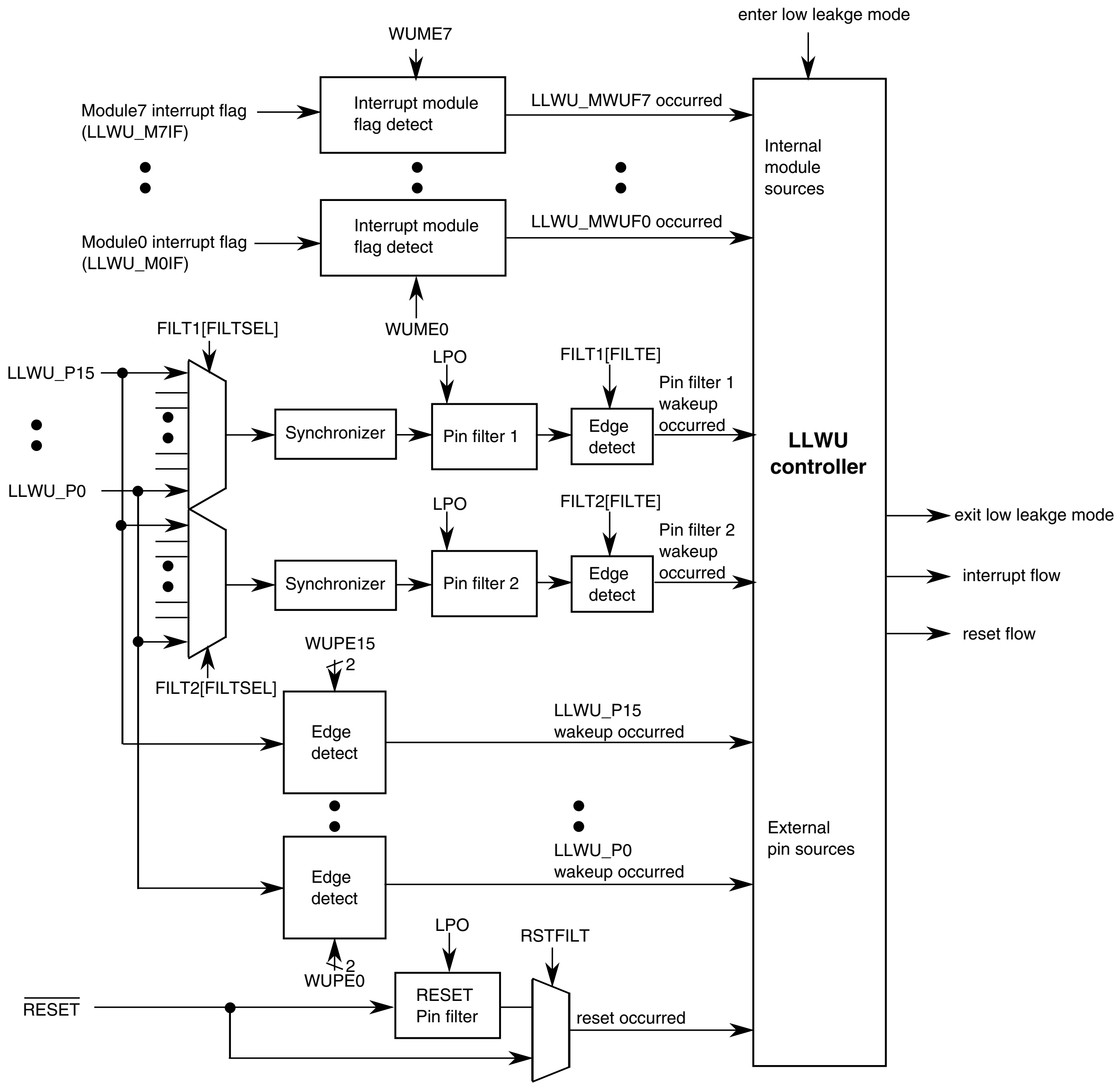


Figure 16-1. LLWU block diagram

16.2 LLWU Signal Descriptions

The signal properties of LLWU are shown in the following table. The external wakeup input pins can be enabled to detect either rising edge, falling edge, or on any change.

Table 16-1. LLWU Signal Descriptions

Signal	Description	I/O
LLWU_Pn	Wakeup inputs (n = 0-15)	I

16.3 Memory map/register definition

The LLWU includes the following registers:

- Five 8-bit wakeup source enable registers
  - Enable external pin input sources
  - Enable internal peripheral sources
- Three 8-bit wakeup flag registers
  - Indication of wakeup source that caused exit from a low-leakage power mode includes external pin or internal module interrupt
- Two 8-bit wakeup pin filter enable registers
- One 8-bit $\overline{\text{RESET}}$ pin filter enable register

NOTE

All LLWU registers are reset by Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. Each register's displayed reset value represents this subset of reset types. LLWU registers are unaffected by reset types that do not trigger Chip Reset not VLLS. For more information about the types of reset on this chip, refer to the Reset details.

LLWU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_C000	LLWU Pin Enable 1 Register (LLWU_PE1)	8	R/W	00h	16.3.1/372
4007_C001	LLWU Pin Enable 2 Register (LLWU_PE2)	8	R/W	00h	16.3.2/373
4007_C002	LLWU Pin Enable 3 Register (LLWU_PE3)	8	R/W	00h	16.3.3/374
4007_C003	LLWU Pin Enable 4 Register (LLWU_PE4)	8	R/W	00h	16.3.4/375
4007_C004	LLWU Module Enable Register (LLWU_ME)	8	R/W	00h	16.3.5/376
4007_C005	LLWU Flag 1 Register (LLWU_F1)	8	R/W	00h	16.3.6/378
4007_C006	LLWU Flag 2 Register (LLWU_F2)	8	R/W	00h	16.3.7/380
4007_C007	LLWU Flag 3 Register (LLWU_F3)	8	R/W	00h	16.3.8/381
4007_C008	LLWU Pin Filter 1 Register (LLWU_FILT1)	8	R/W	00h	16.3.9/383
4007_C009	LLWU Pin Filter 2 Register (LLWU_FILT2)	8	R/W	00h	16.3.10/ 384
4007_C00A	LLWU Reset Enable Register (LLWU_RST)	8	R/W	02h	16.3.11/ 385

16.3.1 LLWU Pin Enable 1 Register (LLWU_PE1)

LLWU_PE1 contains the bit field to enable and select the edge detect type for the external wakeup input pins LLWU_P3-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. See the Reset details for more information.

Address: LLWU_PE1 is 4007_C000h base + 0h offset = 4007_C000h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE3		WUPE2		WUPE1		WUPE0	
Reset	0	0	0	0	0	0	0	0

LLWU_PE1 field descriptions

Field	Description
7–6 WUPE3	Wakeup Pin Enable for LLWU_P3 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE2	Wakeup Pin Enable for LLWU_P2 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE1	Wakeup Pin Enable for LLWU_P1 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
1–0 WUPE0	Wakeup Pin Enable for LLWU_P0 Enables and configures the edge detection for the wakeup pin.

Table continues on the next page...

LLWU_PE1 field descriptions (continued)

Field	Description
	00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.3.2 LLWU Pin Enable 2 Register (LLWU_PE2)

LLWU_PE2 contains the bit field to enable and select the edge detect type for the external wakeup input pins LLWU_P7-LLWU_P4.

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. See the Reset details for more information.

Address: LLWU_PE2 is 4007_C000h base + 1h offset = 4007_C001h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE7		WUPE6		WUPE5		WUPE4	
Reset	0	0	0	0	0	0	0	0

LLWU_PE2 field descriptions

Field	Description
7–6 WUPE7	Wakeup Pin Enable for LLWU_P7 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE6	Wakeup Pin Enable for LLWU_P6 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE5	Wakeup Pin Enable for LLWU_P5

Table continues on the next page...

LLWU_PE2 field descriptions (continued)

Field	Description
	Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
1–0 WUPE4	Wakeup Pin Enable for LLWU_P4 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.3.3 LLWU Pin Enable 3 Register (LLWU_PE3)

LLWU_PE3 contains the bit field to enable and select the edge detect type for the external wakeup input pins LLWU_P11-LLWU_P8.

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. See the Reset details for more information.

Address: LLWU_PE3 is 4007_C000h base + 2h offset = 4007_C002h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE11		WUPE10		WUPE9		WUPE8	
Reset	0	0	0	0	0	0	0	0

LLWU_PE3 field descriptions

Field	Description
7–6 WUPE11	Wakeup Pin Enable for LLWU_P11 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

Table continues on the next page...

LLWU_PE3 field descriptions (continued)

Field	Description
5–4 WUPE10	Wakeup Pin Enable for LLWU_P10 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE9	Wakeup Pin Enable for LLWU_P9 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
1–0 WUPE8	Wakeup Pin Enable for LLWU_P8 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.3.4 LLWU Pin Enable 4 Register (LLWU_PE4)

LLWU_PE4 contains the bit field to enable and select the edge detect type for the external wakeup input pins LLWU_P15-LLWU_P12.

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. See the Reset details for more information.

Address: LLWU_PE4 is 4007_C000h base + 3h offset = 4007_C003h

Bit	7	6	5	4	3	2	1	0
Read Write	WUPE15		WUPE14		WUPE13		WUPE12	
Reset	0	0	0	0	0	0	0	0

LLWU_PE4 field descriptions

Field	Description
7–6 WUPE15	Wakeup Pin Enable for LLWU_P15 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
5–4 WUPE14	Wakeup Pin Enable for LLWU_P14 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
3–2 WUPE13	Wakeup Pin Enable for LLWU_P13 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection
1–0 WUPE12	Wakeup Pin Enable for LLWU_P12 Enables and configures the edge detection for the wakeup pin. 00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection

16.3.5 LLWU Module Enable Register (LLWU_ME)

LLWU_ME contains the bits to enable the internal module flag as a wakeup input source for inputs MWUF7-MWUF0.

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. See the Reset details for more information.

Address: LLWU_ME is 4007_C000h base + 4h offset = 4007_C004h

Bit	7	6	5	4	3	2	1	0
Read Write	WUME7	WUME6	WUME5	WUME4	WUME3	WUME2	WUME1	WUME0
Reset	0	0	0	0	0	0	0	0

LLWU_ME field descriptions

Field	Description
7 WUME7	Wakeup Module Enable for Module 7 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
6 WUME6	Wakeup Module Enable for Module 6 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
5 WUME5	Wakeup Module Enable for Module 5 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
4 WUME4	Wakeup Module Enable for Module 4 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
3 WUME3	Wakeup Module Enable for Module 3 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
2 WUME2	Wakeup Module Enable for Module 2 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source
1 WUME1	Wakeup Module Enable for Module 1 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source

Table continues on the next page...

LLWU_ME field descriptions (continued)

Field	Description
0 WUME0	Wakeup Module Enable for Module 0 Enables an internal module as a wakeup source input. 0 Internal module flag not used as wakeup source 1 Internal module flag used as wakeup source

16.3.6 LLWU Flag 1 Register (LLWU_F1)

LLWU_F1 contains the wakeup flags indicating which wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this will be the source causing the CPU interrupt flow. For VLLS, this will be 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 one to the corresponding WUFx bit. The wakeup flag (WUFx) if set will remain set if the associated WUPEx 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. See the Reset details for more information.

Address: LLWU_F1 is 4007_C000h base + 5h offset = 4007_C005h

Bit	7	6	5	4	3	2	1	0
Read	WUF7	WUF6	WUF5	WUF4	WUF3	WUF2	WUF1	WUF0
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_F1 field descriptions

Field	Description
7 WUF7	Wakeup Flag for LLWU_P7 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 WUF7. 0 LLWU_P7 input was not a wakeup source 1 LLWU_P7 input was a wakeup source
6 WUF6	Wakeup Flag for LLWU_P6 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 WUF6.

Table continues on the next page...

LLWU_F1 field descriptions (continued)

Field	Description
	0 LLWU_P6 input was not a wakeup source 1 LLWU_P6 input was a wakeup source
5 WUF5	Wakeup Flag for LLWU_P5 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 WUF5. 0 LLWU_P5 input was not a wakeup source 1 LLWU_P5 input was a wakeup source
4 WUF4	Wakeup Flag for LLWU_P4 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 WUF4. 0 LLWU_P4 input was not a wakeup source 1 LLWU_P4 input was a wakeup source
3 WUF3	Wakeup Flag for LLWU_P3 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 WUF3. 0 LLWU_P3 input was not a wakeup source 1 LLWU_P3 input was a wakeup source
2 WUF2	Wakeup Flag for LLWU_P2 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 WUF2. 0 LLWU_P2 input was not a wakeup source 1 LLWU_P2 input was a wakeup source
1 WUF1	Wakeup Flag for LLWU_P1 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 WUF1. 0 LLWU_P1 input was not a wakeup source 1 LLWU_P1 input was a wakeup source
0 WUF0	Wakeup Flag for LLWU_P0 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 WUF0. 0 LLWU_P0 input was not a wakeup source 1 LLWU_P0 input was a wakeup source

16.3.7 LLWU Flag 2 Register (LLWU_F2)

LLWU_F2 contains the wakeup flags indicating which wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this will be the source causing the CPU interrupt flow. For VLLS, this will be 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 one to the corresponding WUFx bit. The wakeup flag (WUFx) if set will remain set if the associated WUPEx 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. See the Reset details for more information.

Address: LLWU_F2 is 4007_C000h base + 6h offset = 4007_C006h

Bit	7	6	5	4	3	2	1	0
Read	WUF15	WUF14	WUF13	WUF12	WUF11	WUF10	WUF9	WUF8
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

LLWU_F2 field descriptions

Field	Description
7 WUF15	Wakeup Flag for LLWU_P15 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 WUF15. 0 LLWU_P15 input was not a wakeup source 1 LLWU_P15 input was a wakeup source
6 WUF14	Wakeup Flag for LLWU_P14 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 WUF14. 0 LLWU_P14 input was not a wakeup source 1 LLWU_P14 input was a wakeup source
5 WUF13	Wakeup Flag for LLWU_P13 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 WUF13. 0 LLWU_P13 input was not a wakeup source 1 LLWU_P13 input was a wakeup source

Table continues on the next page...

LLWU_F2 field descriptions (continued)

Field	Description
4 WUF12	Wakeup Flag for LLWU_P12 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 WUF12. 0 LLWU_P12 input was not a wakeup source 1 LLWU_P12 input was a wakeup source
3 WUF11	Wakeup Flag for LLWU_P11 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 WUF11. 0 LLWU_P11 input was not a wakeup source 1 LLWU_P11 input was a wakeup source
2 WUF10	Wakeup Flag for LLWU_P10 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 WUF10. 0 LLWU_P10 input was not a wakeup source 1 LLWU_P10 input was a wakeup source
1 WUF9	Wakeup Flag for LLWU_P9 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 WUF9. 0 LLWU_P9 input was not a wakeup source 1 LLWU_P9 input was a wakeup source
0 WUF8	Wakeup Flag for LLWU_P8 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 WUF8. 0 LLWU_P8 input was not a wakeup source 1 LLWU_P8 input was a wakeup source

16.3.8 LLWU Flag 3 Register (LLWU_F3)

LLWU_F3 contains the wakeup flags indicating which internal wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this will be the source causing the CPU interrupt flow. For VLLS, this will be the source causing the MCU reset flow.

For internal peripherals that are capable of running in a low-leakage power mode, such as RTC or CMP modules, the flag from the associated peripheral is accessible as the MWUFx bit. Clearing of the flag will need to be done in the peripheral instead of writing a one to the MWUFx 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. See the Reset details for more information.

Address: LLWU_F3 is 4007_C000h base + 7h offset = 4007_C007h

Bit	7	6	5	4	3	2	1	0
Read	MWUF7	MWUF6	MWUF5	MWUF4	MWUF3	MWUF2	MWUF1	MWUF0
Write								
Reset	0	0	0	0	0	0	0	0

LLWU_F3 field descriptions

Field	Description
7 MWUF7	Wakeup flag for module 7 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. 0 Module 7 input was not a wakeup source 1 Module 7 input was a wakeup source
6 MWUF6	Wakeup flag for module 6 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. 0 Module 6 input was not a wakeup source 1 Module 6 input was a wakeup source
5 MWUF5	Wakeup flag for module 5 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. 0 Module 5 input was not a wakeup source 1 Module 5 input was a wakeup source
4 MWUF4	Wakeup flag for module 4 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. 0 Module 4 input was not a wakeup source 1 Module 4 input was a wakeup source
3 MWUF3	Wakeup flag for module 3 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. 0 Module 3 input was not a wakeup source 1 Module 3 input was a wakeup source

Table continues on the next page...

LLWU_F3 field descriptions (continued)

Field	Description
2 MWUF2	Wakeup flag for module 2 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. 0 Module 2 input was not a wakeup source 1 Module 2 input was a wakeup source
1 MWUF1	Wakeup flag for module 1 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. 0 Module 1 input was not a wakeup source 1 Module 1 input was a wakeup source
0 MWUF0	Wakeup flag for module 0 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. 0 Module 0 input was not a wakeup source 1 Module 0 input was a wakeup source

16.3.9 LLWU Pin Filter 1 Register (LLWU_FILT1)

LLWU_FILT1 is a control and status register that is used to enable/disable the digital filter 1 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. See the Reset details for more information.

Address: LLWU_FILT1 is 4007_C000h base + 8h offset = 4007_C008h

Bit	7	6	5	4	3	2	1	0
Read	FILTF	FILTE		0	FILTSEL			
Write	w1c							
Reset	0	0	0	0	0	0	0	0

LLWU_FILT1 field descriptions

Field	Description
7 FILTF	Filter Detect Flag

Table continues on the next page...

LLWU_FILT1 field descriptions (continued)

Field	Description
	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. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source
6–5 FILTE	Digital Filter on External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
4 Reserved	This read-only field is reserved and always has the value zero.
3–0 FILTSEL	Filter pin select Selects 1 out of the 16 wakeup pins to be muxed into the filter. 0000 Select LLWU_P0 for filter ... ... 1111 Select LLWU_P15 for filter

16.3.10 LLWU Pin Filter 2 Register (LLWU_FILT2)

LLWU_FILT2 is a control and status register that is used to enable/disable the digital filter 2 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. See the Reset details for more information.

Address: LLWU_FILT2 is 4007_C000h base + 9h offset = 4007_C009h

Bit	7	6	5	4	3	2	1	0
Read	FILTF	FILTE		0	FILTSEL			
Write	w1c			▨				
Reset	0	0	0	0	0	0	0	0

LLWU_FILT2 field descriptions

Field	Description
7 FILTF	Filter Detect 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. 0 Pin Filter 2 was not a wakeup source 1 Pin Filter 2 was a wakeup source
6–5 FILTE	Digital Filter on External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled
4 Reserved	This read-only field is reserved and always has the value zero.
3–0 FILTSEL	Filter pin select Selects 1 out of the 16 wakeup pins to be muxed into the filter. 0000 Select LLWU_P0 for filter ... ... 1111 Select LLWU_P15 for filter

16.3.11 LLWU Reset Enable Register (LLWU_RST)

LLWU_RST is a control register that is used to enable/disable the digital filter for the external pin detect and RESET 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. See the Reset details for more information.

Address: LLWU_RST is 4007_C000h base + Ah offset = 4007_C00Ah

Bit	7	6	5	4	3	2	1	0
Read	0						LLRSTE	RSTFILT
Write								
Reset	0	0	0	0	0	0	1	0

LLWU_RST field descriptions

Field	Description
7–2 Reserved	This read-only field is reserved and always has the value zero.
1 LLRSTE	Low Leakage mode RESET enable This bit must be set to allow the device to be reset while in a low-leakage power mode. On devices where Reset is not a dedicated pin, the RESET pin must also be enabled in the explicit port mux control. 0 RESET pin not enabled as a leakage mode exit source 1 RESET pin enabled as a low leakage mode exit source
0 RSTFILT	Digital Filter on RESET Pin Enables the digital filter for the RESET pin during LLS, VLLS3, VLLS2 or VLLS1 modes. 0 Filter not enabled 1 Filter enabled

16.4 Functional description

This on-chip peripheral module is called a low leakage wakeup unit (LLWU) module because it allows internal peripherals and external input pins as a source of wakeup from low leakage modes. It is operational only in LLS and VLLSx 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. Type options are falling, rising, or 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, such that a detected external pin (wakeup or $\overline{\text{RESET}}$) 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. Two wakeup detect filters are available to detect up to two external pins. A separate reset filter is on the $\overline{\text{RESET}}$ pin. Glitch filtering is not provided on the internal modules.

For internal module wakeup operation, the WUMEx bit enables the associated module as a wakeup source.

16.4.1 LLS mode

Wakeup events triggered from either an external pin input or an internal module input result in a CPU interrupt flow to begin user code execution.

An LLS reset event due to RESET pin assertion causes an exit via a system reset. State retention data is lost, the I/O states return to their reset state. The RCM_SRS[WAKEUP] and RCM_SRS[PIN] bits are set and the system executes a reset flow before CPU operation begins with a reset vector fetch.

16.4.2 VLLS modes

In the case of a wakeup due to external pin or internal module wakeup, recovery is always via a reset flow and the RCM_SRS[WAKEUP] is set indicating the low leakage mode was active. State retention data is lost and I/O will be restored after the PMC_REGSC[ACKISO] has been written.

A VLLS exit event due to $\overline{\text{RESET}}$ pin assertion causes an exit via a system reset. State retention data is lost, the I/O states immediately return to their reset state. The RCM_SRS[WAKEUP] and RCM_SRS[PIN] bits are set and the system executes a reset flow before CPU operation begins with a reset vector fetch.

16.4.3 Initialization

For an enabled peripheral wakeup input, the peripheral flag should be cleared by software before entering LLS or VLLSx mode to avoid an immediate exit from the mode.

Flags associated with external input pins (filtered and unfiltered) should also be cleared by software prior to entry to LLS or VLLSx mode.

After enabling an external pin filter or changing the source pin, wait at least 5 LPO clock cycles before entering LLS or VLLSx mode to allow the filter to initialize.

Chapter 17
Miscellaneous Control Module (MCM)

17.1 Introduction

NOTE

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

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

17.1.1 Features

The MCM includes these distinctive features:

- Program-visible information on the platform configuration and revision
- Control and counting logic for ETB almost full
- Error status and interrupts for the cache write buffer
- Error status and interrupts for the core's floating-point unit (FPU)

17.2 Memory Map/Register Descriptions

The memory map and register descriptions below describe the registers using byte addresses.

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	17.2.1/390
E008_000A	Crossbar switch (AXBS) master configuration (MCM_PLAMC)	16	R	003Fh	17.2.2/391
E008_000C	Control register (MCM_CR)	32	R/W	0000_0000h	17.2.3/391
E008_0010	Interrupt status and control register (MCM_ISCR)	32	R	0000_0000h	17.2.4/393
E008_0014	ETB counter control register (MCM_ETBCC)	32	R/W	0000_0000h	17.2.5/396
E008_0018	ETB reload register (MCM_ETBRL)	32	R/W	0000_0000h	17.2.6/397
E008_001C	ETB counter value register (MCM_ETBCNT)	32	R	0000_0000h	17.2.7/398
E008_0020	Fault address register (MCM_FADR)	32	R	Undefined	17.2.8/398
E008_0024	Fault attributes register (MCM_FATR)	32	R	Undefined	17.2.9/399
E008_0028	Fault data register (MCM_FDR)	32	R	Undefined	17.2.10/ 400
E008_0030	Process ID register (MCM_PID)	32	R/W	0000_0000h	17.2.11/ 401

17.2.1 Crossbar switch (AXBS) slave configuration (MCM_PLASC)

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

Address: MCM_PLASC is 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 read-only field is reserved and always has the value zero.
7–0 ASC	Each bit in the ASC field indicates if 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

17.2.2 Crossbar switch (AXBS) master configuration (MCM_PLAMC)

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

Address: MCM_PLAMC is 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	1	1	1	1	1	1

MCM_PLAMC field descriptions

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 AMC	Each bit in the AMC field indicates if there is a corresponding connection to the AXBS master input port. 0 A bus master connection to AXBS input port *n* is absent 1 A bus master connection to AXBS input port *n* is present

17.2.3 Control register (MCM_CR)

The CR register defines the arbitration and protection schemes for the two SRAM arrays.

NOTE

Bits 23-0 are undefined after reset.

Address: MCM_CR is E008_0000h base + Ch offset = E008_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SRAMLWP	SRAMLAP		0	SRAMUWP	SRAMUAP		Reserved							
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	Reserved						Reserved	Reserved								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MCM_CR field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30 SRAMLWP	SRAM_L write protect When this bit is set, writes to SRAM_L array generates a bus error.
29–28 SRAMLAP	SRAM_L arbitration priority Defines the arbitration scheme and priority for the processor and SRAM backdoor accesses to the SRAM_L array. 00 Round robin 01 Special round robin (favors SRAM backoor accesses over the processor) 10 Fixed priority. Processor has highest, backdoor has lowest 11 Fixed priority. Backdoor has highest, processor has lowest
27 Reserved	This read-only field is reserved and always has the value zero.
26 SRAMUWP	SRAM_U write protect When this bit is set, writes to SRAM_U array generates a bus error.
25–24 SRAMUAP	SRAM_U arbitration priority Defines the arbitration scheme and priority for the processor and SRAM backdoor accesses to the SRAM_U array. 00 Round robin 01 Special round robin (favors SRAM backoor accesses over the processor)

Table continues on the next page...

MCM_CR field descriptions (continued)

Field	Description
	10 Fixed priority. Processor has highest, backdoor has lowest 11 Fixed priority. Backdoor has highest, processor has lowest
23–10 Reserved	This field is reserved.
9 Reserved	This field is reserved.
8–0 Reserved	This field is reserved.

17.2.4 Interrupt status and control register (MCM_ISCR)

The MCM_ISCR register defines the configuration and reports status for a number of core-related interrupt exception conditions. It includes the enable and status bits associated with the core's floating-point exceptions, bus errors associated with the core's cache write buffer, and events associated with the debug ETB module. The individual event indicators are first qualified with their exception enables and then logically summed to form an interrupt request sent to the core's NVIC.

Bits 15-8 are read-only indicator flags based on the processor's FPSCR register. Attempted writes to these bits are ignored. Once set, the flags remain asserted until software clears the corresponding FPSCR bit.

Address: MCM_ISCR is E008_0000h base + 10h offset = E008_0010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FIDCE	0		FIXCE	FUFCE	FOFCE	FDZCE	FIOCE	0			CWBEE	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	FIDC	0		FIXC	FUFC	FOFC	FDZC	FIOC	0			CWBER	DHREQ	NMI	IRQ	0
W												w1c		w1c	w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MCM_ISCR field descriptions

Field	Description
31 FIDCE	FPU input denormal interrupt enable 0 Disable interrupt 1 Enable interrupt
30–29 Reserved	This read-only field is reserved and always has the value zero.
28 FIXCE	FPU inexact interrupt enable 0 Disable interrupt 1 Enable interrupt
27 FUFCE	FPU underflow interrupt enable 0 Disable interrupt 1 Enable interrupt
26 FOFCE	FPU overflow interrupt enable 0 Disable interrupt 1 Enable interrupt
25 FDZCE	FPU divide-by-zero interrupt enable 0 Disable interrupt 1 Enable interrupt
24 FIOCE	FPU invalid operation interrupt enable 0 Disable interrupt 1 Enable interrupt
23–21 Reserved	This read-only field is reserved and always has the value zero.
20 CWBEE	Cache write buffer error enable Enables the generation of an interrupt in response to a bus error termination reported on a system bus transfer initiated from the cache's write buffer. 0 Disable error interrupt 1 Enable error interrupt
19–16 Reserved	This read-only field is reserved and always has the value zero.
15 FIDC	FPU input denormal interrupt status This read-only bit is a copy of the core's FPSCR[IDC] bit and signals input denormalized number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IDC] bit. 0 No interrupt 1 Interrupt occurred
14–13 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

MCM_ISCR field descriptions (continued)

Field	Description
12 FIXC	FPU inexact interrupt status This read-only bit is a copy of the core's FPSCR[IXC] bit and signals an inexact number has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IXC] bit. 0 No interrupt 1 Interrupt occurred
11 FUFC	FPU underflow interrupt status This read-only bit is a copy of the core's FPSCR[UFC] bit and signals an underflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[UFC] bit. 0 No interrupt 1 Interrupt occurred
10 FOFC	FPU overflow interrupt status This read-only bit is a copy of the core's FPSCR[OFC] bit and signals an overflow has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[OFC] bit. 0 No interrupt 1 Interrupt occurred
9 FDZC	FPU divide-by-zero interrupt status This read-only bit is a copy of the core's FPSCR[DZC] bit and signals a divide by zero has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[DZC] bit. 0 No interrupt 1 Interrupt occurred
8 FIOC	FPU invalid operation interrupt status This read-only bit is a copy of the core's FPSCR[IOC] bit and signals an illegal operation has been detected in the processor's FPU. Once set, this bit remains set until software clears the FPSCR[IOC] bit. 0 No interrupt 1 Interrupt occurred
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 CWBER	Cache write buffer error status Signals a data transfer from the core's cache write buffer was terminated with a bus error. This bit only sets when the corresponding enable bit (CWBEE) is set. The corresponding core fault address, attributes and write data are typically retrieved from the FADR, FATR, and FDR registers during the interrupt service routine before clearing the CWBER flag. 0 No error 1 Error occurred
3 DHREQ	Debug halt request indicator Indicates that a debug halt request is initiated due to a ETB counter expiration (ETBCC[2:0] = 3b111 & ETBCV[10:0] = 11h0. This bit is cleared when the counter is disabled or when the ETB counter is reloaded.

Table continues on the next page...

MCM_ISCR field descriptions (continued)

Field	Description
	0 No debug halt request 1 Debug halt request initiated
2 NMI	Non-maskable interrupt pending If ETBCC[RSPT] is set to 10b, this bit is set when the ETB counter expires. 0 No pending NMI 1 Due to the ETB counter expiring, an NMI is pending
1 IRQ	Normal interrupt pending If ETBCC[RSPT] is set to 01b, this bit is set when the ETB counter expires. 0 No pending interrupt 1 Due to the ETB counter expiring, a normal interrupt is pending
0 Reserved	This read-only field is reserved and always has the value zero.

17.2.5 ETB counter control register (MCM_ETBCC)

Address: MCM_ETBCC is E008_0000h base + 14h offset = E008_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																										ITDIS	ETDIS	RLRQ	RSPT		CNTEN
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

MCM_ETBCC field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5 ITDIS	ITM-to-TPIU disable Disables the trace path from ITM to TPIU 0 ITM-to-TPIU trace path enabled 1 ITM-to-TPIU trace path disabled
4 ETDIS	ETM-to-TPIU disable Disables the trace path from ETM to TPIU 0 ETM-to-TPIU trace path enabled 1 ETM-to-TPIU trace path disabled
3 RLRQ	Reload request Reloads the ETB packet counter with the MCM_ETBRL RELOAD value.

Table continues on the next page...

MCM_ETBCC field descriptions (continued)

Field	Description
	If IRQ or NMI interrupts were enabled and an NMI or IRQ interrupt was generated on counter expiration, setting this bit clears the pending NMI or IRQ interrupt request. If debug halt was enabled and a debug halt request was asserted on counter expiration, setting this bit clears the debug halt request. 0 No effect 1 Clears pending debug halt, NMI, or IRQ interrupt requests
2–1 RSPT	Response type 00 No response when the ETB count expires 01 Generate a normal interrupt when the ETB count expires 10 Generate an NMI when the ETB count expires 11 Generate a debug halt when the ETB count expires
0 CNTEN	Counter enable Enables the ETB counter. 0 ETB counter disabled 1 ETB counter enabled

17.2.6 ETB reload register (MCM_ETBRL)

Address: MCM_ETBRL is E008_0000h base + 18h offset = E008_0018h

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																					RELOAD										
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

MCM_ETBRL field descriptions

Field	Description
31–11 Reserved	This read-only field is reserved and always has the value zero.
10–0 RELOAD	Byte count reload value Indicates the 0-mod-4 value the counter reloads to. Writing a non-0-mod-4 value to this field results in an bus error

17.2.7 ETB counter value register (MCM_ETBCNT)

Address: MCM_ETBCNT is E008_0000h base + 1Ch offset = E008_001Ch

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																					COUNTER										
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

MCM_ETBCNT field descriptions

Field	Description
31–11 Reserved	This read-only field is reserved and always has the value zero.
10–0 COUNTER	Byte count counter value Indicates the current 0-mod-4 value of the counter.

17.2.8 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: MCM_FADR is 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
31–0 ADDRESS	Fault address

MCM_FADR field descriptions (continued)

Field	Description

17.2.9 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: MCM_FATR is 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 anothercache 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.

Table continues on the next page...

MCM_FATR field descriptions (continued)

Field	Description
30–12 Reserved	This read-only field is reserved and always has the value zero.
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 read-only field is reserved and always has the value zero.
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 read-only field is reserved and always has the value zero.
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

17.2.10 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: MCM_FDR is E008_0000h base + 28h offset = E008_0028h

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	DATA																															
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_FDR field descriptions

Field	Description
31–0 DATA	Fault data

17.2.11 Process ID register (MCM_PID)

This register drives the M0_PID and M1_PID values in the MPU. System software loads this register before passing control to a given user mode process. If the PID of the process does not match the value in this register, a bus error occurs. See the MPU chapter for more details.

Address: MCM_PID is E008_0000h base + 30h offset = E008_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																								PID							
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

MCM_PID field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 PID	M0_PID and M1_PID for MPU Drives the M0_PID and M1_PID values in the MPU

17.3 Functional Description

This section describes the functional description of MCM module.

17.3.1 Interrupts

The MCM generates two interrupt requests:

- Non-maskable interrupt
- Normal interrupt

17.3.1.1 Non-maskable interrupt

The MCM's non-maskable interrupt (NMI) is generated, if:

- MCM_ISCR[ETBN] is set, which is caused by
  - The ETB counter is enabled (MCM_ETBCC[CNTEN] = 1),
  - The ETB count expires, and
  - The response to counter expiration is an NMI (MCM_ETBCC[RSPT] = 10)

17.3.1.2 Normal interrupt

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

- MCM_ISCR[ETBI] is set, which is caused by
  - The ETB counter is enabled (MCM_ETBCC[CNTEN] = 1),
  - The ETB count expires, and
  - The response to counter expiration is a normal interrupt (MCM_ETBCC[RSPT] = 01)
- Cache write buffer error interrupt is enabled (CWBEE) and a cache write buffer error occurs (CWBER)
- FPU input denormal interrupt is enabled (FIDCE) and an input is denormalized (FIDC)
- FPU inexact interrupt is enabled (FIXCE) and a number is inexact (FIXC)
- FPU underflow interrupt is enabled (FUFCE) and an underflow occurs (FUFC)
- FPU overflow interrupt is enabled (FOFCE) and an overflow occurs (FOFC)

- FPU divide-by-zero interrupt is enabled (FDZCE) and a divide-by-zero occurs (FDZC)
- FPU invalid operation interrupt is enabled (FDZCE) and an invalid occurs (FDZC)

17.3.1.3 Determining source of the normal interrupt

To determine the exact source of the normal interrupt qualify the interrupt status flags with the corresponding interrupt enable bits.

1. Form MCM_ISCR[31:16] && MCM_ISCR[15:0]
2. Search the result for asserted flags, which indicate the exact interrupt sources

Chapter 18
Crossbar Switch (AXBS)

18.1 Introduction

NOTE

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

This chapter 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 all bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave. A variety of bus arbitration methods and attributes may be programmed on a slave-by-slave basis.

18.1.1 Features

The crossbar switch includes these distinctive features:

- Symmetric crossbar bus switch implementation
    - Allows concurrent accesses from different masters to different slaves
    - Slave arbitration attributes configured on a slave-by-slave basis
- 32-bit width and support for byte, 2-byte, 4-byte, and 16-byte burst transfers
- Operation at a 1-to-1 clock frequency with the bus masters
- Low-Power Park mode support

18.2 Memory Map / Register Definition

Each slave port of the crossbar switch contains configuration registers. Read- and write-transfers require two bus clock cycles. The registers can be read from and written to only in supervisor mode. Additionally, these registers can be read from or written to only by 32-bit accesses.

A bus error response is returned if an unimplemented location is accessed within the crossbar switch.

The slave registers also feature a bit that, when set, prevents the registers from being written. The registers remain readable, but future write attempts have no effect on the registers and are terminated with a bus error response to the master initiating the write. The core, for example, takes a bus error interrupt.

NOTE

This section shows the registers for all eight master and slave ports. If a master or slave is not used on this particular device, then unexpected results occur when writing to its registers. See the chip configuration details for the exact master/slave assignments for your device.

AXBS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_4000	Priority Registers Slave (AXBS_PRS0)	32	R/W	7654_3210h	18.2.1/407
4000_4010	Control Register (AXBS_CRS0)	32	R/W	0000_0000h	18.2.2/411
4000_4100	Priority Registers Slave (AXBS_PRS1)	32	R/W	7654_3210h	18.2.1/407
4000_4110	Control Register (AXBS_CRS1)	32	R/W	0000_0000h	18.2.2/411
4000_4200	Priority Registers Slave (AXBS_PRS2)	32	R/W	7654_3210h	18.2.1/407
4000_4210	Control Register (AXBS_CRS2)	32	R/W	0000_0000h	18.2.2/411
4000_4300	Priority Registers Slave (AXBS_PRS3)	32	R/W	7654_3210h	18.2.1/407
4000_4310	Control Register (AXBS_CRS3)	32	R/W	0000_0000h	18.2.2/411
4000_4400	Priority Registers Slave (AXBS_PRS4)	32	R/W	7654_3210h	18.2.1/407
4000_4410	Control Register (AXBS_CRS4)	32	R/W	0000_0000h	18.2.2/411
4000_4500	Priority Registers Slave (AXBS_PRS5)	32	R/W	7654_3210h	18.2.1/407
4000_4510	Control Register (AXBS_CRS5)	32	R/W	0000_0000h	18.2.2/411

Table continues on the next page...

AXBS memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_4600	Priority Registers Slave (AXBS_PRS6)	32	R/W	7654_3210h	18.2.1/407
4000_4610	Control Register (AXBS_CRS6)	32	R/W	0000_0000h	18.2.2/411
4000_4700	Priority Registers Slave (AXBS_PRS7)	32	R/W	7654_3210h	18.2.1/407
4000_4710	Control Register (AXBS_CRS7)	32	R/W	0000_0000h	18.2.2/411
4000_4800	Master General Purpose Control Register (AXBS_MGPCR0)	32	R/W	0000_0000h	18.2.3/413
4000_4900	Master General Purpose Control Register (AXBS_MGPCR1)	32	R/W	0000_0000h	18.2.3/413
4000_4A00	Master General Purpose Control Register (AXBS_MGPCR2)	32	R/W	0000_0000h	18.2.3/413
4000_4B00	Master General Purpose Control Register (AXBS_MGPCR3)	32	R/W	0000_0000h	18.2.3/413
4000_4C00	Master General Purpose Control Register (AXBS_MGPCR4)	32	R/W	0000_0000h	18.2.3/413
4000_4D00	Master General Purpose Control Register (AXBS_MGPCR5)	32	R/W	0000_0000h	18.2.3/413
4000_4E00	Master General Purpose Control Register (AXBS_MGPCR6)	32	R/W	0000_0000h	18.2.3/413
4000_4F00	Master General Purpose Control Register (AXBS_MGPCR7)	32	R/W	0000_0000h	18.2.3/413

18.2.1 Priority Registers Slave (AXBS_PRS*n*)

The priority registers (PRSn) set the priority of each master port on a per slave port basis and reside in each slave port. The priority register can be accessed only with 32-bit accesses. After the CRSn[RO] bit is set, the PRSn register can only be read; attempts to write to it have no effect on PRSn and result in a bus-error response to the master initiating the write.

No two available master ports may be programmed with the same priority level. Attempts to program two or more masters with the same priority level result in a bus-error response and the PRSn is not updated.

NOTE

> The possible values for the PRSn fields depend on the number of masters available on the device. See the device's chip configuration details for the number of masters supported.

- If the device contains less than five masters, values 000–011 are valid and writing other values results in an error.
- If the device contains *n* masters where *n* ≥ 5, values 0 to *n*-1 are valid and writing other values results in an error.

Addresses: AXBS_PRS0 is 4000_4000h base + 0h offset = 4000_4000h
AXBS_PRS1 is 4000_4000h base + 100h offset = 4000_4100h
AXBS_PRS2 is 4000_4000h base + 200h offset = 4000_4200h
AXBS_PRS3 is 4000_4000h base + 300h offset = 4000_4300h
AXBS_PRS4 is 4000_4000h base + 400h offset = 4000_4400h
AXBS_PRS5 is 4000_4000h base + 500h offset = 4000_4500h
AXBS_PRS6 is 4000_4000h base + 600h offset = 4000_4600h
AXBS_PRS7 is 4000_4000h base + 700h offset = 4000_4700h

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	M7			0	M6			0	M5			0	M4			0	M3			0	M2			0	M1			0	M0		
W	▨				▨				▨				▨				▨				▨				▨				▨			
Reset	0	1	1	1	0	1	1	0	0	1	0	1	0	1	0	0	0	0	1	1	0	0	1	0	0	0	0	1	0	0	0	0

AXBS_PRS*n* field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30–28 M7	Master 7 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.
27 Reserved	This read-only field is reserved and always has the value zero.
26–24 M6	Master 6 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.
23 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

AXBS_PRS*n* field descriptions (continued)

Field	Description
22–20 M5	Master 5 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.
19 Reserved	This read-only field is reserved and always has the value zero.
18–16 M4	Master 4 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.
15 Reserved	This read-only field is reserved and always has the value zero.
14–12 M3	Master 3 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.
11 Reserved	This read-only field is reserved and always has the value zero.
10–8 M2	Master 2 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.

Table continues on the next page...

AXBS_PRS*n* field descriptions (continued)

Field	Description
7 Reserved	This read-only field is reserved and always has the value zero.
6–4 M1	Master 1 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.
3 Reserved	This read-only field is reserved and always has the value zero.
2–0 M0	Master 0 Priority. Sets the arbitration priority for this port on the associated slave port. 000 This master has level 1, or highest, priority when accessing the slave port. 001 This master has level 2 priority when accessing the slave port. 010 This master has level 3 priority when accessing the slave port. 011 This master has level 4 priority when accessing the slave port. 100 This master has level 5 priority when accessing the slave port. 101 This master has level 6 priority when accessing the slave port. 110 This master has level 7 priority when accessing the slave port. 111 This master has level 8, or lowest, priority when accessing the slave port.

18.2.2 Control Register (AXBS_CRS*n*)

These registers control several features of each slave port and must be accessed using 32-bit accesses. After CRSn[RO] is set, the CRSn can only be read; attempts to write to it have no effect and result in an error response.

Addresses: AXBS_CRS0 is 4000_4000h base + 10h offset = 4000_4010h
AXBS_CRS1 is 4000_4000h base + 110h offset = 4000_4110h
AXBS_CRS2 is 4000_4000h base + 210h offset = 4000_4210h
AXBS_CRS3 is 4000_4000h base + 310h offset = 4000_4310h
AXBS_CRS4 is 4000_4000h base + 410h offset = 4000_4410h
AXBS_CRS5 is 4000_4000h base + 510h offset = 4000_4510h
AXBS_CRS6 is 4000_4000h base + 610h offset = 4000_4610h
AXBS_CRS7 is 4000_4000h base + 710h offset = 4000_4710h

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	RO	HLP	0																				ARB		0		PCTL		0	PARK		
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

AXBS_CRS*n* field descriptions

Field	Description
31 RO	Read Only Forces the slave port's CSRn and PRSn registers to be read-only. After set, only a hardware reset clears it. 0 The slave port's registers are writeable 1 The slave port's registers are read-only and cannot be written. Attempted writes have no effect on the registers and result in a bus error response.
30 HLP	Halt Low Priority Sets the initial arbitration priority for low power mode requests. Setting this bit will not effect the request for low power mode from attaining highest priority once it has control of the slave ports. 0 The low power mode request has the highest priority for arbitration on this slave port 1 The low power mode request has the lowest initial priority for arbitration on this slave port
29–10 Reserved	This read-only field is reserved and always has the value zero.
9–8 ARB	Arbitration Mode Selects the arbitration policy for the slave port. 00 Fixed priority 01 Round-robin, or rotating, priority 10 Reserved 11 Reserved

Table continues on the next page...

AXBS_CRS*n* field descriptions (continued)

Field	Description
7–6 Reserved	This read-only field is reserved and always has the value zero.
5–4 PCTL	Parking Control Determines the slave port's parking control. The low-power park feature results in an overall power savings if the slave port is not saturated. However, this forces an extra latency clock when any master tries to access the slave port while not in use because it is not parked on any master. 00 When no master makes a request, the arbiter parks the slave port on the master port defined by the PARK field 01 When no master makes a request, the arbiter parks the slave port on the last master to be in control of the slave port 10 When no master makes a request, the slave port is not parked on a master and the arbiter drives all outputs to a constant safe state 11 Reserved
3 Reserved	This read-only field is reserved and always has the value zero.
2–0 PARK	Park Determines which master port the current slave port parks on when no masters are actively making requests and the PCTL bits are cleared. **NOTE:** Only select master ports that are actually present on the device. If not, undefined behavior may occur. 000 Park on master port M0 001 Park on master port M1 010 Park on master port M2 011 Park on master port M3 100 Park on master port M4 101 Park on master port M5 110 Park on master port M6 111 Park on master port M7

18.2.3 Master General Purpose Control Register (AXBS_MGPCR*n*)

The MGPCR controls only whether the master's undefined length burst accesses are allowed to complete uninterrupted or whether they can be broken by requests from higher priority masters. The MGPCR can be accessed only in Supervisor mode with 32-bit accesses.

Addresses: AXBS_MGPCR0 is 4000_4000h base + 800h offset = 4000_4800h
AXBS_MGPCR1 is 4000_4000h base + 900h offset = 4000_4900h
AXBS_MGPCR2 is 4000_4000h base + A00h offset = 4000_4A00h
AXBS_MGPCR3 is 4000_4000h base + B00h offset = 4000_4B00h
AXBS_MGPCR4 is 4000_4000h base + C00h offset = 4000_4C00h
AXBS_MGPCR5 is 4000_4000h base + D00h offset = 4000_4D00h
AXBS_MGPCR6 is 4000_4000h base + E00h offset = 4000_4E00h
AXBS_MGPCR7 is 4000_4000h base + F00h offset = 4000_4F00h

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																													AULB		
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

AXBS_MGPCR*n* field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value zero.
2–0 AULB	Arbitrates On Undefined Length Bursts Determines whether, and when, the crossbar switch arbitrates away the slave port the master owns when the master is performing undefined length burst accesses. 000 No arbitration is allowed during an undefined length burst 001 Arbitration is allowed at any time during an undefined length burst 010 Arbitration is allowed after four beats of an undefined length burst 011 Arbitration is allowed after eight beats of an undefined length burst 100 Arbitration is allowed after 16 beats of an undefined length burst 101 Reserved 110 Reserved 111 Reserved

18.3 Functional Description

18.3.1 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 peripheral'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.

A master is given control of the targeted slave port only after a previous access to a different slave port completes, regardless of its priority on the newly targeted slave port. This prevents deadlock from occurring when:

- A higher priority master has:
  - An outstanding request to one slave port that has a long response time and
  - A pending access to a different slave port, and
- A lower priority master is also making a request to the same slave port as the pending access of the higher priority master.

After the master has control of the slave port it is targeting, the master remains in control of that slave port until it gives up the slave port by running an IDLE cycle or by leaving that slave port for its next access.

The master could also lose control of the slave port if another higher priority master makes a request to the slave port; however, if the master is running a fixed-length burst transfer it retains control of the slave port until that transfer completes. Based on MGPCR[AULB], the master either retains control of the slave port when doing undefined length incrementing burst transfers or loses the bus to a higher priority master.

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 can park the slave port on the master port indicated by CRS*n*[PARK]. This is done to save the initial clock of arbitration delay that otherwise would be seen if the master had to arbitrate to gain control of the slave port. The slave port can also be put into Low Power Park mode to save power, by using CRS*n*[PCTL].

18.3.2 Register coherency

The operation of the crossbar is affected as soon as a register is written. The values of the registers do not track with slave-port-related master accesses, but instead track only with slave accesses.

The MGPCRx[AULB] bits are the exception to this rule. The update of these bits is only recognized when the master on that master port runs an IDLE cycle, even though the slave bus cycle to write them will have already terminated successfully. If the MGPCRx[AULB] bits are written between two burst accesses, the new AULB encodings do not take effect until an IDLE cycle is initiated by the master on that master port.

18.3.3 Arbitration

The crossbar switch supports two arbitration schemes:

- A fixed-priority comparison algorithm
- A round-robin fairness algorithm

The arbitration scheme is independently programmable for each slave port.

18.3.3.1 Arbitration during undefined length bursts

Arbitration points during an undefined length burst are defined by the current master's MGPCR[AULB] field setting. When a defined length is imposed on the burst via the AULB bits, the undefined length burst is treated as a single or series of single back-to-back fixed-length burst accesses.

The following figure illustrates an example:

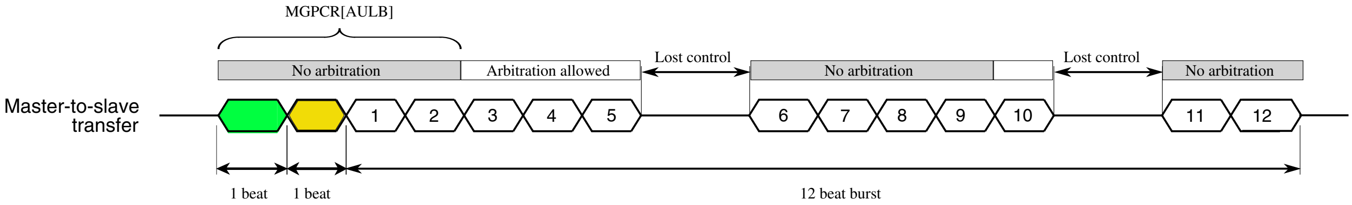


Figure 18-28. Undefined length burst example

In this example, a master runs an undefined length burst and the MGPCR[AULB] bits indicate arbitration occurs after the fourth beat of the burst. The master runs two sequential beats and then starts what will be a 12-beat undefined length burst access to a new address within the same slave port region as the previous access. The crossbar does not allow an arbitration point until the fourth overall access, or the second beat of the second burst. At that point, all remaining accesses are open for arbitration until the master loses control of the slave port.

Assume the master loses control of the slave port after the fifth beat of the second burst. After the master regains control of the slave port no arbitration point is available until after the master has run four more beats of its burst. After the fourth beat of the now continued burst, or the ninth beat of the second burst from the master's perspective, is taken, all beats of the burst are once again open for arbitration until the master loses control of the slave port.

Assume the master again loses control of the slave port on the fifth beat of the third now continued burst, or the 10th beat of the second burst from the master's perspective. After the master regains control of the slave port, it is allowed to complete its final two beats of its burst without facing arbitration.

Note

Fixed-length burst accesses are not affected by the AULB bits. All fixed-length burst accesses lock out arbitration until the last beat of the fixed-length burst.

18.3.3.2 Fixed-priority operation

When operating in Fixed-Priority mode, each master is assigned a unique priority level in the priority registers (PRS*n*) . If two masters request access to a slave port, the master with the highest priority in the selected priority register gains control over the slave 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 18-29. How AXBS grants control of a slave port to a master

When	Then AXBS 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 a fixed length burst transfer or a locked transfer. • The requesting master's priority level is higher than that of the current master.	At the end of the burst transfer or locked transfer
The master is running an undefined length burst transfer.	At the next arbitration point **NOTE:** Arbitration points for an undefined length burst are defined in the MGPCR for each master.
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

18.3.3.3 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, accounting for locked and fixed-length burst transfers. 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, 5, and then 0.

Parking may continue to be used in a round-robin mode, but does not affect the round-robin pointer unless the parked master actually performs a transfer. Handoff occurs to the next master in line after one cycle of arbitration. If the slave port is put into low-power park mode, the round-robin pointer is reset to point at master port 0, giving it the highest priority.

18.3.3.4 Priority assignment

Each master port must be assigned a unique 3-bit priority level. If an attempt is made to program multiple master ports with the same priority level within the priority registers (PRS*n*), the crossbar switch responds with a bus error and the registers are not updated.

18.4 Initialization/application information

No initialization is required by or for the crossbar switch. Hardware reset ensures all the register bits used by the crossbar switch are properly initialized to a valid state. However, settings and priorities may be programmed to achieve maximum system performance.

Chapter 19
Memory Protection Unit (MPU)

19.1 Introduction

NOTE

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

The Memory Protection Unit (MPU) provides hardware access control for all memory references generated in the device.

19.2 Overview

The MPU concurrently monitors all system bus transactions and evaluates their appropriateness using pre-programmed region descriptors that define memory spaces and their access rights. 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 a protection error response.

19.2.1 Block Diagram

A simplified block diagram of the MPU module is shown in the following figure. The hardware's two-dimensional connection matrix is clearly visible with the basic access evaluation macro shown as the replicated submodule block. The crossbar switch slave ports are shown on the left, the region descriptor registers 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 the access protection error. For details of the access evaluation macro, see Access Evaluation Macro.

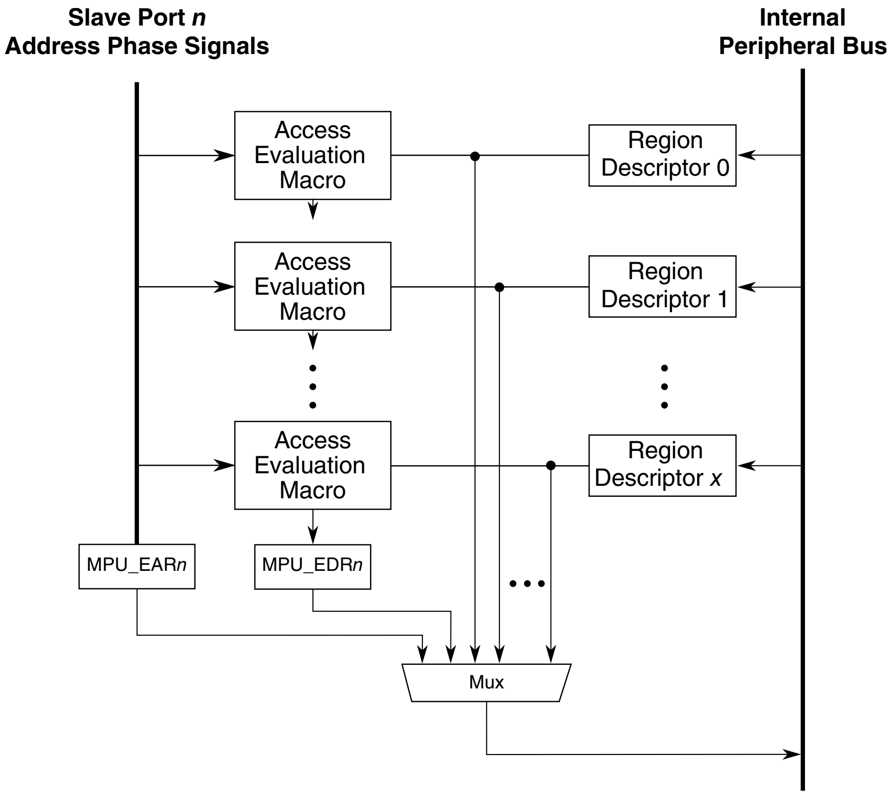


Figure 19-1. MPU Block Diagram

19.2.2 Features

The MPU implements a two-dimensional hardware array of memory region descriptors and the crossbar slave ports to continuously monitor the legality of every memory reference generated by each bus master in the system. The feature set includes:

- 16 program-visible 128-bit region descriptors, accessible by four 32-bit words each
    - Each region descriptor defines a modulo-32 byte space, aligned anywhere in memory
        - Region sizes can vary from 32 bytes to 4 Gbytes
    - Two access control permissions defined in a single descriptor word
        - Masters 0–3: read, write, and execute attributes for supervisor and user accesses
        - Masters 4–7: read and write attributes
    - Hardware-assisted maintenance of the descriptor valid bit minimizes coherency issues

  - Alternate programming model view of the access control permissions word
  - Priority given to granting permission over denying access for overlapping region descriptors
- Detects access protection errors if a memory reference does not hit in any memory region, or if the reference is illegal in all hit memory regions. If an access error occurs, the reference is terminated with an error response, and the MPU inhibits the bus cycle being sent to the targeted slave device.
- Error registers (per slave port) capture the last faulting address, attributes, and other information
- Global MPU enable/disable control bit

19.3 Memory Map/Register Definition

The programming model is partitioned into three groups: control/status registers, the data structure containing the region descriptors, and the alternate view of the region descriptor access control values.

The programming model can only be referenced using 32-bit accesses. Attempted references using different access sizes, to undefined (reserved) addresses, or with a non-supported access type (a write to a read-only register, or a read of a write-only register) generate an error termination.

The programming model can be accessed only in supervisor mode.

NOTE

See the chip configuration details for any chip-specific register information for this module.

MPU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_D000	Control/Error Status Register (MPU_CESR)	32	R/W	0081_8201h	19.3.1/425
4000_D010	Error Address Register, Slave Port n (MPU_EAR0)	32	R	Undefined	19.3.2/426
4000_D014	Error Detail Register, Slave Port n (MPU_EDR0)	32	R	Undefined	19.3.3/427
4000_D018	Error Address Register, Slave Port n (MPU_EAR1)	32	R	Undefined	19.3.2/426
4000_D01C	Error Detail Register, Slave Port n (MPU_EDR1)	32	R	Undefined	19.3.3/427

Table continues on the next page...

MPU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_D020	Error Address Register, Slave Port n (MPU_EAR2)	32	R	Undefined	19.3.2/426
4000_D024	Error Detail Register, Slave Port n (MPU_EDR2)	32	R	Undefined	19.3.3/427
4000_D028	Error Address Register, Slave Port n (MPU_EAR3)	32	R	Undefined	19.3.2/426
4000_D02C	Error Detail Register, Slave Port n (MPU_EDR3)	32	R	Undefined	19.3.3/427
4000_D030	Error Address Register, Slave Port n (MPU_EAR4)	32	R	Undefined	19.3.2/426
4000_D034	Error Detail Register, Slave Port n (MPU_EDR4)	32	R	Undefined	19.3.3/427
4000_D400	Region Descriptor n, Word 0 (MPU_RGD0_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D404	Region Descriptor n, Word 1 (MPU_RGD0_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D408	Region Descriptor n, Word 2 (MPU_RGD0_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D40C	Region Descriptor n, Word 3 (MPU_RGD0_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D410	Region Descriptor n, Word 0 (MPU_RGD1_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D414	Region Descriptor n, Word 1 (MPU_RGD1_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D418	Region Descriptor n, Word 2 (MPU_RGD1_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D41C	Region Descriptor n, Word 3 (MPU_RGD1_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D420	Region Descriptor n, Word 0 (MPU_RGD2_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D424	Region Descriptor n, Word 1 (MPU_RGD2_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D428	Region Descriptor n, Word 2 (MPU_RGD2_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D42C	Region Descriptor n, Word 3 (MPU_RGD2_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D430	Region Descriptor n, Word 0 (MPU_RGD3_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D434	Region Descriptor n, Word 1 (MPU_RGD3_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D438	Region Descriptor n, Word 2 (MPU_RGD3_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D43C	Region Descriptor n, Word 3 (MPU_RGD3_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D440	Region Descriptor n, Word 0 (MPU_RGD4_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D444	Region Descriptor n, Word 1 (MPU_RGD4_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D448	Region Descriptor n, Word 2 (MPU_RGD4_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D44C	Region Descriptor n, Word 3 (MPU_RGD4_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D450	Region Descriptor n, Word 0 (MPU_RGD5_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D454	Region Descriptor n, Word 1 (MPU_RGD5_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D458	Region Descriptor n, Word 2 (MPU_RGD5_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D45C	Region Descriptor n, Word 3 (MPU_RGD5_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D460	Region Descriptor n, Word 0 (MPU_RGD6_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D464	Region Descriptor n, Word 1 (MPU_RGD6_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D468	Region Descriptor n, Word 2 (MPU_RGD6_WORD2)	32	R/W	0000_0000h	19.3.6/429

Table continues on the next page...

MPU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_D46C	Region Descriptor n, Word 3 (MPU_RGD6_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D470	Region Descriptor n, Word 0 (MPU_RGD7_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D474	Region Descriptor n, Word 1 (MPU_RGD7_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D478	Region Descriptor n, Word 2 (MPU_RGD7_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D47C	Region Descriptor n, Word 3 (MPU_RGD7_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D480	Region Descriptor n, Word 0 (MPU_RGD8_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D484	Region Descriptor n, Word 1 (MPU_RGD8_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D488	Region Descriptor n, Word 2 (MPU_RGD8_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D48C	Region Descriptor n, Word 3 (MPU_RGD8_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D490	Region Descriptor n, Word 0 (MPU_RGD9_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D494	Region Descriptor n, Word 1 (MPU_RGD9_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D498	Region Descriptor n, Word 2 (MPU_RGD9_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D49C	Region Descriptor n, Word 3 (MPU_RGD9_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D4A0	Region Descriptor n, Word 0 (MPU_RGD10_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D4A4	Region Descriptor n, Word 1 (MPU_RGD10_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D4A8	Region Descriptor n, Word 2 (MPU_RGD10_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D4AC	Region Descriptor n, Word 3 (MPU_RGD10_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D4B0	Region Descriptor n, Word 0 (MPU_RGD11_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D4B4	Region Descriptor n, Word 1 (MPU_RGD11_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D4B8	Region Descriptor n, Word 2 (MPU_RGD11_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D4BC	Region Descriptor n, Word 3 (MPU_RGD11_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D4C0	Region Descriptor n, Word 0 (MPU_RGD12_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D4C4	Region Descriptor n, Word 1 (MPU_RGD12_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D4C8	Region Descriptor n, Word 2 (MPU_RGD12_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D4CC	Region Descriptor n, Word 3 (MPU_RGD12_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D4D0	Region Descriptor n, Word 0 (MPU_RGD13_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D4D4	Region Descriptor n, Word 1 (MPU_RGD13_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D4D8	Region Descriptor n, Word 2 (MPU_RGD13_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D4DC	Region Descriptor n, Word 3 (MPU_RGD13_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D4E0	Region Descriptor n, Word 0 (MPU_RGD14_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D4E4	Region Descriptor n, Word 1 (MPU_RGD14_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D4E8	Region Descriptor n, Word 2 (MPU_RGD14_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D4EC	Region Descriptor n, Word 3 (MPU_RGD14_WORD3)	32	R/W	0000_0000h	19.3.7/432

Table continues on the next page...

MPU memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_D4F0	Region Descriptor n, Word 0 (MPU_RGD15_WORD0)	32	R/W	0000_0000h	19.3.4/428
4000_D4F4	Region Descriptor n, Word 1 (MPU_RGD15_WORD1)	32	R/W	0000_001Fh	19.3.5/429
4000_D4F8	Region Descriptor n, Word 2 (MPU_RGD15_WORD2)	32	R/W	0000_0000h	19.3.6/429
4000_D4FC	Region Descriptor n, Word 3 (MPU_RGD15_WORD3)	32	R/W	0000_0000h	19.3.7/432
4000_D800	Region Descriptor Alternate Access Control n (MPU_RGDAAC0)	32	R/W	0000_0000h	19.3.8/433
4000_D804	Region Descriptor Alternate Access Control n (MPU_RGDAAC1)	32	R/W	0000_0000h	19.3.8/433
4000_D808	Region Descriptor Alternate Access Control n (MPU_RGDAAC2)	32	R/W	0000_0000h	19.3.8/433
4000_D80C	Region Descriptor Alternate Access Control n (MPU_RGDAAC3)	32	R/W	0000_0000h	19.3.8/433
4000_D810	Region Descriptor Alternate Access Control n (MPU_RGDAAC4)	32	R/W	0000_0000h	19.3.8/433
4000_D814	Region Descriptor Alternate Access Control n (MPU_RGDAAC5)	32	R/W	0000_0000h	19.3.8/433
4000_D818	Region Descriptor Alternate Access Control n (MPU_RGDAAC6)	32	R/W	0000_0000h	19.3.8/433
4000_D81C	Region Descriptor Alternate Access Control n (MPU_RGDAAC7)	32	R/W	0000_0000h	19.3.8/433
4000_D820	Region Descriptor Alternate Access Control n (MPU_RGDAAC8)	32	R/W	0000_0000h	19.3.8/433
4000_D824	Region Descriptor Alternate Access Control n (MPU_RGDAAC9)	32	R/W	0000_0000h	19.3.8/433
4000_D828	Region Descriptor Alternate Access Control n (MPU_RGDAAC10)	32	R/W	0000_0000h	19.3.8/433
4000_D82C	Region Descriptor Alternate Access Control n (MPU_RGDAAC11)	32	R/W	0000_0000h	19.3.8/433
4000_D830	Region Descriptor Alternate Access Control n (MPU_RGDAAC12)	32	R/W	0000_0000h	19.3.8/433
4000_D834	Region Descriptor Alternate Access Control n (MPU_RGDAAC13)	32	R/W	0000_0000h	19.3.8/433
4000_D838	Region Descriptor Alternate Access Control n (MPU_RGDAAC14)	32	R/W	0000_0000h	19.3.8/433
4000_D83C	Region Descriptor Alternate Access Control n (MPU_RGDAAC15)	32	R/W	0000_0000h	19.3.8/433

19.3.1 Control/Error Status Register (MPU_CESR)

Address: MPU_CESR is 4000_D000h base + 0h offset = 4000_D000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SPERR								1	0			HRL			
W	w1c															
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NSP				NRGD				0							VLD
W																
Reset	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

MPU_CESR field descriptions

Field	Description
31–24 SPERR	Slave port n error Indicates a captured error in EARn and EDRn. This bit is set when the hardware detects an error and records the faulting address and attributes. It is cleared by writing one to it. If another error is captured at the exact same cycle as the write, the flag remains set. A find-first-one instruction (or equivalent) can detect the presence of a captured error. The following shows the correspondence between the bit number and slave port number: • Bit 31 corresponds to slave port 0. • Bit 30 corresponds to slave port 1. • Bit 29 corresponds to slave port 2. • Bit 28 corresponds to slave port 3. • Bit 27 corresponds to slave port 4. • Bit 26 corresponds to slave port 5. • Bit 25 corresponds to slave port 6. • Bit 24 corresponds to slave port 7. 0 No error has occurred for slave port n. 1 An error has occurred for slave port n.
23 Reserved	This read-only field is reserved and always has the value one.
22–20 Reserved	This read-only field is reserved and always has the value zero.
19–16 HRL	Hardware revision level Specifies the MPU's hardware and definition revision level. It can be read by software to determine the functional definition of the module.
15–12 NSP	Number of slave ports Specifies the number of slave ports connected to the MPU.
11–8 NRGD	Number of region descriptors Indicates the number of region descriptors implemented in the MPU. 0000 8 region descriptors 0001 12 region descriptors 0010 16 region descriptors

Table continues on the next page...

MPU_CESR field descriptions (continued)

Field	Description
7–1 Reserved	This read-only field is reserved and always has the value zero.
0 VLD	Valid (global enable/disable for the MPU) 0 MPU is disabled. All accesses from all bus masters are allowed. 1 MPU is enabled

19.3.2 Error Address Register, Slave Port n (MPU_EAR*n*)

When the MPU detects an access error on slave port n, the 32-bit reference address is captured in this read-only register and the corresponding bit in CESR[SPERR] set. Additional information about the faulting access is captured in the corresponding EDRn at the same time. This register and the corresponding EDRn contain the most recent access error; there are no hardware interlocks with CESR[SPERR], as the error registers are always loaded upon the occurrence of each protection violation.

Addresses: MPU_EAR0 is 4000_D000h base + 10h offset = 4000_D010h
MPU_EAR1 is 4000_D000h base + 18h offset = 4000_D018h
MPU_EAR2 is 4000_D000h base + 20h offset = 4000_D020h
MPU_EAR3 is 4000_D000h base + 28h offset = 4000_D028h
MPU_EAR4 is 4000_D000h base + 30h offset = 4000_D030h

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	EADDR																															
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.

MPU_EAR*n* field descriptions

Field	Description
31–0 EADDR	Error address Indicates the reference address from slave port n that generated the access error

19.3.3 Error Detail Register, Slave Port n (MPU_EDR*n*)

When the MPU detects an access error on slave port n, 32 bits of error detail are captured in this read-only register and the corresponding bit in CESR[SPERR] is set. Information on the faulting address is captured in the corresponding EARn register at the same time. This register and the corresponding EARn register contain the most recent access error; there are no hardware interlocks with CESR[SPERR] as the error registers are always loaded upon the occurrence of each protection violation.

Addresses: MPU_EDR0 is 4000_D000h base + 14h offset = 4000_D014h
MPU_EDR1 is 4000_D000h base + 1Ch offset = 4000_D01Ch
MPU_EDR2 is 4000_D000h base + 24h offset = 4000_D024h
MPU_EDR3 is 4000_D000h base + 2Ch offset = 4000_D02Ch
MPU_EDR4 is 4000_D000h base + 34h offset = 4000_D034h

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td colspan="16">EACD</td></tr>
<tr><td>W</td><td colspan="16"></td></tr>
<tr><td>Reset</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td></tr>
</table>

<table>
<tr><td>Bit</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="8">EPID</td><td colspan="4">EMN</td><td colspan="3">EATTR</td><td>ERW</td></tr>
<tr><td>W</td><td colspan="8"></td><td colspan="4"></td><td colspan="3"></td><td></td></tr>
<tr><td>Reset</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td></tr>
</table>

* Notes:
- x = Undefined at reset.

MPU_EDR*n* field descriptions

Field	Description
31–16 EACD	Error access control detail Indicates the region descriptor with the access error. If EDRn contains a captured error and EACD is cleared, an access did not hit in any region descriptor. If only a single EACD bit is set, the protection error was caused by a single non-overlapping region descriptor. If two or more EACD bits are set, the protection error was caused by an overlapping set of region descriptors.
15–8 EPID	Error process identification Records the process identifier of the faulting reference. The process identifier is typically driven only by processor cores; for other bus masters, this field is cleared.
7–4 EMN	Error master number Indicates the bus master that generated the access error.

Table continues on the next page...

MPU_EDR*n* field descriptions (continued)

Field	Description
3–1 EATTR	Error attributes Indicates attribute information about the faulting reference. **NOTE:** All other encodings are reserved. 000 User mode, instruction access 001 User mode, data access 010 Supervisor mode, instruction access 011 Supervisor mode, data access
0 ERW	Error read/write Indicates the access type of the faulting reference. 0 Read 1 Write

19.3.4 Region Descriptor n, Word 0 (MPU_RGD_WORD0)

The first word of the region descriptor defines the 0-modulo-32 byte start address of the memory region. Writes to this register clear the region descriptor's valid bit (RGDn_WORD3[VLD]).

Addresses: 4000_D000h base + 400h offset + (16d × *n*), where *n* = 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	SRTADDR																											0				
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

MPU_RGD*n*_WORD0 field descriptions

Field	Description
31–5 SRTADDR	Start address Defines the most significant bits of the 0-modulo-32 byte start address of the memory region.
4–0 Reserved	This read-only field is reserved and always has the value zero.

19.3.5 Region Descriptor n, Word 1 (MPU_RGD_WORD1)

The second word of the region descriptor defines the 31-modulo-32 byte end address of the memory region. Writes to this register clear the region descriptor's valid bit (RGDn_WORD3[VLD]).

Addresses: 4000_D000h base + 404h offset + (16d × *n*), where *n* = 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	ENDADDR																											Reserved				
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	1	1	1	1	1

MPU_RGD*n*_WORD1 field descriptions

Field	Description
31–5 ENDADDR	End address Defines the most significant bits of the 31-modulo-32 byte end address of the memory region. **NOTE:** The MPU does not verify that ENDADDR ≥ SRTADDR.
4–0 Reserved	This field is reserved.

19.3.6 Region Descriptor n, Word 2 (MPU_RGD_WORD2)

The third word of the region descriptor defines the access control rights of the memory region. The access control privileges depend on two broad classifications of bus masters:

- Bus masters 0–3 have a 5-bit field defining separate privilege rights for user and supervisor mode accesses, as well as the optional inclusion of a process identification field within the definition.
- Bus masters 4–7 are limited to separate read and write permissions.

For the privilege rights of bus masters 0–3, there are three flags associated with this function:

- Read (r) refers to accessing the referenced memory address using an operand (data) fetch
- Write (w) refers to updating the referenced memory address using a store (data) instruction
- Execute (x) refers to reading the referenced memory address using an instruction fetch

Writes to RGDn_WORD2 clear the region descriptor's valid bit (RGDn_WORD3[VLD]). If only updating the access controls, write to RGDAACn instead because stores to these locations do not affect the descriptor's valid bit.

Addresses: 4000_D000h base + 408h offset + (16d × *n*), where *n* = 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	M7RE	M7WE	M6RE	M6WE	M5RE	M5WE	M4RE	M4WE	M3PE	M3SM		M3UM			M2PE	M2SM		M2UM			M1PE	M1SM		M1UM			M0PE	M0SM		M0UM		
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

MPU_RGD*n*_WORD2 field descriptions

Field	Description
31 M7RE	Bus master 7 read enable. 0 Bus master 7 reads terminate with an access error and the read is not performed 1 Bus master 7 reads allowed
30 M7WE	Bus master 7 write enable 0 Bus master 7 writes terminate with an access error and the write is not performed 1 Bus master 7 writes allowed
29 M6RE	Bus master 6 read enable. 0 Bus master 6 reads terminate with an access error and the read is not performed 1 Bus master 6 reads allowed
28 M6WE	Bus master 6 write enable 0 Bus master 6 writes terminate with an access error and the write is not performed 1 Bus master 6 writes allowed
27 M5RE	Bus master 5 read enable. 0 Bus master 5 reads terminate with an access error and the read is not performed 1 Bus master 5 reads allowed
26 M5WE	Bus master 5 write enable 0 Bus master 5 writes terminate with an access error and the write is not performed 1 Bus master 5 writes allowed
25 M4RE	Bus master 4 read enable. 0 Bus master 4 reads terminate with an access error and the read is not performed 1 Bus master 4 reads allowed
24 M4WE	Bus master 4 write enable 0 Bus master 4 writes terminate with an access error and the write is not performed 1 Bus master 4 writes allowed
23 M3PE	Bus master 3 process identifier enable. 0 Do not include the process identifier in the evaluation 1 Include the process identifier and mask (RGDn_WORD3) in the region hit evaluation

Table continues on the next page...

MPU_RGD*n*_WORD2 field descriptions (continued)

Field	Description
22–21 M3SM	Bus master 3 supervisor mode access control Defines the access controls for bus master 3 in supervisor mode 00 r/w/x; read, write and execute allowed 01 r/x; read and execute allowed, but no write 10 r/w; read and write allowed, but no execute 11 Same as user mode defined in M3UM
20–18 M3UM	Bus master 3 user mode access control Defines the access controls for bus master 3 in user mode. M3UM consists of three independent bits, enabling read (r), write (w), and execute (x) permissions. 0 An attempted access of that mode may be terminated with an access error (if not allowed by another descriptor) and the access not performed. 1 Allows the given access type to occur
17 M2PE	Bus master 2 process identifier enable See M3PE description
16–15 M2SM	Bus master 2 supervisor mode access control See M3SM description
14–12 M2UM	Bus master 2 user mode access control See M3UM description
11 M1PE	Bus master 1 process identifier enable See M1PE description
10–9 M1SM	Bus master 1 supervisor mode access control See M3SM description
8–6 M1UM	Bus master 1 user mode access control See M3UM description
5 M0PE	Bus master 0 process identifier enable See M0PE description
4–3 M0SM	Bus master 0 supervisor mode access control See M3SM description
2–0 M0UM	Bus master 0 user mode access control See M3UM description

19.3.7 Region Descriptor n, Word 3 (MPU_RGD_WORD3)

The fourth word of the region descriptor contains the optional process identifier and mask, plus the region descriptor's valid bit.

Addresses: 4000_D000h base + 40Ch offset + (16d × *n*), where *n* = 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	PID								PIDMASK								0															VLD
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

MPU_RGD*n*_WORD3 field descriptions

Field	Description
31–24 PID	Process identifier Specifies the process identifier that is included in the region hit determination if RGDn_WORD2[MxPE] is set. PIDMASK can mask individual bits in this field.
23–16 PIDMASK	Process identifier mask Provides a masking capability so that multiple process identifiers can be included as part of the region hit determination. If a bit in PIDMASK is set, then the corresponding PID bit is ignored in the comparison. This field and PID are included in the region hit determination if RGDn_WORD2[MxPE] is set. For more information on the handling of the PID and PIDMASK, see "Access Evaluation - Hit Determination."
15–1 Reserved	This read-only field is reserved and always has the value zero.
0 VLD	Valid Signals the region descriptor is valid. Any write to RGDn_WORD0–2 clears this bit. 0 Region descriptor is invalid 1 Region descriptor is valid

19.3.8 Region Descriptor Alternate Access Control n (MPU_RGDAAC*n*)

Since software may adjust only the access controls within a region descriptor (RGDn_WORD2) as different tasks execute, an alternate programming view of this 32-bit entity is available. Writing to this register does not affect the descriptor's valid bit.

Addresses: 4000_D000h base + 800h offset + (4d × *n*), where *n* = 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	M7RE	M7WE	M6RE	M6WE	M5RE	M5WE	M4RE	M4WE	M3PE	M3SM		M3UM			M2PE	M2SM		M2UM			M1PE	M1SM		M1UM			M0PE	M0SM		M0UM		
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

MPU_RGDAAC*n* field descriptions

Field	Description
31 M7RE	Bus master 7 read enable. 0 Bus master 7 reads terminate with an access error and the read is not performed 1 Bus master 7 reads allowed
30 M7WE	Bus master 7 write enable 0 Bus master 7 writes terminate with an access error and the write is not performed 1 Bus master 7 writes allowed
29 M6RE	Bus master 6 read enable. 0 Bus master 6 reads terminate with an access error and the read is not performed 1 Bus master 6 reads allowed
28 M6WE	Bus master 6 write enable 0 Bus master 6 writes terminate with an access error and the write is not performed 1 Bus master 6 writes allowed
27 M5RE	Bus master 5 read enable. 0 Bus master 5 reads terminate with an access error and the read is not performed 1 Bus master 5 reads allowed
26 M5WE	Bus master 5 write enable 0 Bus master 5 writes terminate with an access error and the write is not performed 1 Bus master 5 writes allowed
25 M4RE	Bus master 4 read enable. 0 Bus master 4 reads terminate with an access error and the read is not performed 1 Bus master 4 reads allowed

Table continues on the next page...

MPU_RGDAAC*n* field descriptions (continued)

Field	Description
24 M4WE	Bus master 4 write enable 0 Bus master 4 writes terminate with an access error and the write is not performed 1 Bus master 4 writes allowed
23 M3PE	Bus master 3 process identifier enable. 0 Do not include the process identifier in the evaluation 1 Include the process identifier and mask (RGDn.RGDAAC) in the region hit evaluation
22–21 M3SM	Bus master 3 supervisor mode access control Defines the access controls for bus master 3 in supervisor mode 00 r/w/x; read, write and execute allowed 01 r/x; read and execute allowed, but no write 10 r/w; read and write allowed, but no execute 11 Same as user mode defined in M3UM
20–18 M3UM	Bus master 3 user mode access control Defines the access controls for bus master 3 in user mode. M3UM consists of three independent bits, enabling read (r), write (w), and execute (x) permissions. 0 An attempted access of that mode may be terminated with an access error (if not allowed by another descriptor) and the access not performed. 1 Allows the given access type to occur
17 M2PE	Bus master 2 process identifier enable See M3PE description.
16–15 M2SM	Bus master 2 supervisor mode access control See M3SM description.
14–12 M2UM	Bus master 2 user mode access control See M3UM description.
11 M1PE	Bus master 1 process identifier enable See M3PE description.
10–9 M1SM	Bus master 1 supervisor mode access control See M3SM description.
8–6 M1UM	Bus master 1 user mode access control See M3UM description.
5 M0PE	Bus master 0 process identifier enable See M3PE description.
4–3 M0SM	Bus master 0 supervisor mode access control See M3SM description.

Table continues on the next page...

MPU_RGDAAC*n* field descriptions (continued)

Field	Description
2–0 M0UM	Bus master 0 user mode access control See M3UM description.

19.4 Functional Description

In this section, the functional operation of the MPU is detailed, including the operation of the access evaluation macro and the handling of error-terminated bus cycles.

19.4.1 Access Evaluation Macro

The basic operation of the MPU is performed in the access evaluation macro, a hardware structure replicated in the two-dimensional connection matrix. As shown in the following figure, the access evaluation macro inputs the crossbar bus address phase signals and the contents of a region descriptor (RGD*n*) and performs two major functions: region hit determination and detection of an access protection violation. The following figure shows a functional block diagram.

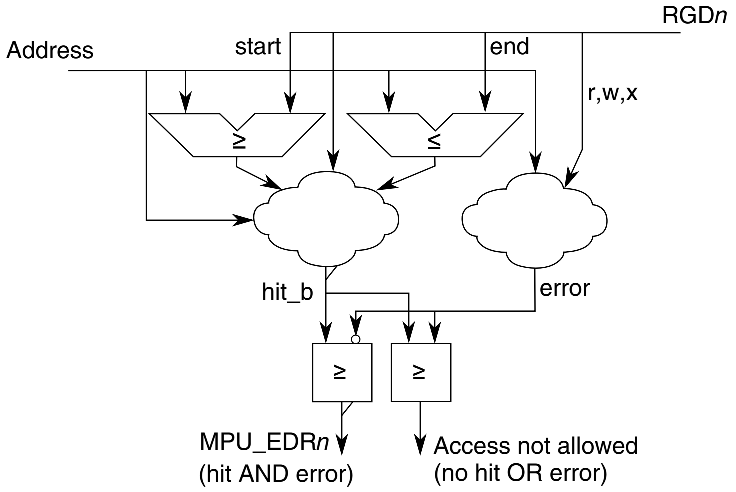


Figure 19-100. MPU Access Evaluation Macro

19.4.1.1 Hit Determination

To determine if the current reference hits in the given region, two magnitude comparators are used with the region's start and end addresses. The boolean equation for this portion of the hit determination is:

```
region_hit = ((addr[31:5] >= RGDn_Word0[SRTADDR]) & (addr[31:5] <= RGDn_Word1[ENDADDR])) &
RGDn_Word3[VLD]
```

where addr is the current reference address, RGD*n*_Word0[SRTADDR] and RGD*n*_Word1[ENDADDR] are the start and end addresses, and RGD*n*_Word3[VLD] is the valid bit.

NOTE

The MPU does not verify that ENDADDR ≥ SRTADDR.

In addition to the comparison of the reference address versus the region descriptor's start and end addresses, the optional process identifier is examined against the region descriptor's PID and PIDMASK fields. A process identifier hit term is formed as follows:

```
 pid_hit = RGDn_Word2[MxPE] | ((current_pid |
            RGDn_Word3[PIDMASK]) == (RGDn_Word3[PID] | RGDn_Word3[PIDMASK]))
```

where the current_pid is the selected process identifier from the current bus master, and RGD*n*_Word3[PID] and RGD*n*_Word3[PIDMASK] are the process identifier fields from region descriptor *n*. For bus masters that do not output a process identifier, the MPU forces the pid_hit term to assert.

19.4.1.2 Privilege Violation Determination

While the access evaluation macro is determining region hit, the logic is also evaluating if the current access is allowed by the permissions defined in the region descriptor. Using the master and supervisor/user mode signals, a set of effective permissions is generated from the appropriate fields in the region descriptor. The protection violation logic then evaluates the access against the effective permissions using the specification shown below.

Table 19-100. Protection Violation Definition

Description	MxUM			Protection Violation?
	r	w	x	
Instruction fetch read	—	—	0	Yes, no execute permission
	—	—	1	No, access is allowed
Data read	0	—	—	Yes, no read permission
	1	—	—	No, access is allowed
Data write	—	0	—	Yes, no write permission
	—	1	—	No, access is allowed

19.4.2 Putting It All Together and Error Terminations

For each slave port monitored, the MPU performs a reduction-AND of all the individual terms from each access evaluation macro. This expression then terminates the bus cycle with an error and reports a protection error for three conditions:

1. If the access does not hit in any region descriptor, a protection error is reported.
2. If the access hits in a single region descriptor and that region signals a protection violation, a protection error is reported.
3. If the access hits in multiple (overlapping) regions and all regions signal protection violations, a protection error is reported.

As shown in the third condition, granting permission is a higher priority than denying access for overlapping regions. This approach is more flexible to system software in region descriptor assignments. For an example of the use of overlapping region descriptors, see Application Information.

19.4.3 Power Management

Disabling the MPU by clearing CESR[VLD] minimizes power dissipation. To minimize the power dissipation of an enabled MPU, invalidate unused region descriptors by clearing the associated RGDn_Word3[VLD] bits.

19.5 Initialization Information

At system startup, load the appropriate number of region descriptors, including setting RGD*n*_Word3[VLD]. Setting CESR[VLD] enables the module.

If the system requires that all the loaded region descriptors be enabled simultaneously, first ensure that the entire MPU is disabled (CESR[VLD]=0).

Note

> A region descriptor must be set to allow access to the MPU registers if further changes are needed.

19.6 Application Information

In an operational system, interfacing with the MPU is generally classified into the following activities:

- Creating a new memory region—Load the appropriate region descriptor into an available RGD*n*, using four sequential 32-bit writes. The hardware assists in the maintenance of the valid bit, so if this approach is followed, there are no coherency issues with the multi-cycle descriptor writes. (Clearing RGD*n*_Word3[VLD] deletes/ removes an existing memory region.)
- Altering only access privileges—To not affect the valid bit, write to the alternate version of the access control word (RGDAAC*n*), so there are no coherency issues involved with the update. When the write completes, the memory region's access rights switch instantaneously to the new value.
- Changing a region's start and end addresses—Write a minimum of three words to the region descriptor (RGD*n*_Word{0,1,3}). Word 0 and 1 redefine the start and end addresses, respectively. Word 3 re-enables the region descriptor valid bit. In most situations, all four words of the region descriptor are rewritten.
- Accessing the MPU—Allocate a region descriptor to restrict MPU access to supervisor mode from a specific master.
- Detecting an access error—The current bus cycle is terminated with an error response and EAR*n* and EDR*n* capture information on the faulting reference. The error-terminated bus cycle typically initiates an error response in the originating bus master. For example, a processor core may respond with a bus error exception, while a data movement bus master may respond with an error interrupt. The processor can retrieve the captured error address and detail information simply by reading E{A,D}R*n*. CESR[SPERR] signals which error registers contain captured fault data.
- Overlapping region descriptors—Applying overlapping regions often reduces the number of descriptors required for a given set of access controls. In the overlapping memory space, the protection rights of the corresponding region descriptors are logically summed together (the boolean OR operator).

  The following dual-core system example contains four bus masters: the two processors (CP0, CP1) and two DMA engines (DMA1, a traditional data movement engine transferring data between RAM and peripherals and DMA2, a second engine transferring data to/from the RAM only). Consider the following region descriptor assignments:

Table 19-101. Overlapping Region Descriptor Example

Region Description	RGDn		CP0	CP1	DMA1	DMA2	
CP0 code	0		rwx	r--	—	—	Flash
CP1 code	1		r--	rwx	—	—	
CP0 data & stack	2		rw-	—	—	—	RAM
CP0 → CP1 shared data	2	3	r--	r--	—	—	
CP1 → CP0 shared data	4						
CP1 data & stack	4		—	rw-	—	—	
Shared DMA data	5		rw-	rw-	rw	rw	
MPU	6		rw-	rw-	—	—	Peripheral space
Peripherals	7		rw-	rw-	rw	—	

In this example, there are eight descriptors used to span nine regions in the three main spaces of the system memory map (flash, RAM, and peripheral space). Each region indicates the specific permissions for each of the four bus masters and this definition provides an appropriate set of shared, private and executable memory spaces.

Of particular interest are the two overlapping spaces: region descriptors 2 & 3 and 3 & 4.

The space defined by RGD2 with no overlap is a private data and stack area that provides read/write access to CP0 only. The overlapping space between RGD2 and RGD3 defines a shared data space for passing data from CP0 to CP1 and the access controls are defined by the logical OR of the two region descriptors. Thus, CP0 has (rw- | r--) = (rw-) permissions, while CP1 has (--- | r--) = (r--) permission in this space. Both DMA engines are excluded from this shared processor data region. The overlapping spaces between RGD3 and RGD4 defines another shared data space, this one for passing data from CP1 to CP0. For this overlapping space, CP0 has (r-- | ---) = (r--) permission, while CP1 has (rw- | r--) = (rw-) permission. The non-overlapped space of RGD4 defines a private data and stack area for CP1 only.

The space defined by RGD5 is a shared data region, accessible by all four bus masters. Finally, the slave peripheral space mapped onto the IPS bus is partitioned into two regions: one containing the MPU's programming model accessible only to the two processor cores and the remaining peripheral region accessible to both processors and the traditional DMA1 master.

This simple example is intended to show one possible application of the capabilities of the MPU in a typical system.

Chapter 20
Peripheral Bridge (AIPS-Lite)

20.1 Introduction

NOTE

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

The peripheral bridge (AIPS-Lite) converts the crossbar switch interface to an interface to access a majority of peripherals on the device.

The peripheral bridge supports up to 128 peripherals. The peripheral bridge occupies a 64 MB portion of the address space. The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

20.1.1 Features

Key features of the peripheral bridge are:

- Supports up to 128 peripherals
- Supports 8-, 16-, and 32-bit width peripheral slots
- Each independently configurable peripheral includes a clock enable, which allows peripherals to operate at any speed less than the system clock rate.
- Programming model provides memory protection functionality

20.1.2 General operation

The peripherals connected to the peripheral bridge are modules that contain readable/writable control and status registers. The system masters read and write these registers through the peripheral bridge. The peripheral bridge generates module enables, the module address, transfer attributes, byte enables, and write data as inputs to the peripherals. The peripheral bridge captures read data from the peripheral interface and drives it to the crossbar switch.

The register maps of the peripherals are located on 4 KB boundaries. Each peripheral is allocated one 4 KB block of the memory map.

The peripheral bridge (AIPS-Lite) memory map is illustrated as follows.

Addresses	Description
Base + 0x000_0000 - 0x000_0FFF	Module #0
Base + 0x000_1000 - 0x000_1FFF	Module #1
...	...
Base + 0x007_F000 - 0x007_FFFF	Module #127

20.2 Memory map/register definition

The peripheral bridge registers are 32-bit registers and can only be accessed in supervisor mode by trusted bus masters. Additionally, these registers must only be read from or written to by a 32-bit aligned access. The peripheral bridge registers are mapped into the PACR0 address space.

Two system clocks are required for read accesses, and three system clocks are required for write accesses to the peripheral bridge registers.

NOTE

The number of fields and registers available depends on the device-specific implementation of the peripheral bridge module. See the Chip Configuration chapter for more information.

AIPS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_0000	Master Privilege Register A (AIPS0_MPRA)	32	R/W	Undefined	20.2.1/443
4000_0020	Peripheral Access Control Register (AIPS0_PACRA)	32	R/W	4444_4444h	20.2.2/447
4000_0024	Peripheral Access Control Register (AIPS0_PACRB)	32	R/W	4444_4444h	20.2.2/447
4000_0028	Peripheral Access Control Register (AIPS0_PACRC)	32	R/W	4444_4444h	20.2.2/447
4000_002C	Peripheral Access Control Register (AIPS0_PACRD)	32	R/W	4444_4444h	20.2.2/447

Table continues on the next page...

AIPS memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_0040	Peripheral Access Control Register (AIPS0_PACRE)	32	R/W	Undefined	20.2.3/453
4000_0044	Peripheral Access Control Register (AIPS0_PACRF)	32	R/W	Undefined	20.2.3/453
4000_0048	Peripheral Access Control Register (AIPS0_PACRG)	32	R/W	Undefined	20.2.3/453
4000_004C	Peripheral Access Control Register (AIPS0_PACRH)	32	R/W	Undefined	20.2.3/453
4000_0050	Peripheral Access Control Register (AIPS0_PACRI)	32	R/W	Undefined	20.2.3/453
4000_0054	Peripheral Access Control Register (AIPS0_PACRJ)	32	R/W	Undefined	20.2.3/453
4000_0058	Peripheral Access Control Register (AIPS0_PACRK)	32	R/W	Undefined	20.2.3/453
4000_005C	Peripheral Access Control Register (AIPS0_PACRL)	32	R/W	Undefined	20.2.3/453
4000_0060	Peripheral Access Control Register (AIPS0_PACRM)	32	R/W	Undefined	20.2.3/453
4000_0064	Peripheral Access Control Register (AIPS0_PACRN)	32	R/W	Undefined	20.2.3/453
4000_0068	Peripheral Access Control Register (AIPS0_PACRO)	32	R/W	Undefined	20.2.3/453
4000_006C	Peripheral Access Control Register (AIPS0_PACRP)	32	R/W	Undefined	20.2.3/453
4008_0000	Master Privilege Register A (AIPS1_MPRA)	32	R/W	Undefined	20.2.1/443
4008_0020	Peripheral Access Control Register (AIPS1_PACRA)	32	R/W	4444_4444h	20.2.2/447
4008_0024	Peripheral Access Control Register (AIPS1_PACRB)	32	R/W	4444_4444h	20.2.2/447
4008_0028	Peripheral Access Control Register (AIPS1_PACRC)	32	R/W	4444_4444h	20.2.2/447
4008_002C	Peripheral Access Control Register (AIPS1_PACRD)	32	R/W	4444_4444h	20.2.2/447
4008_0040	Peripheral Access Control Register (AIPS1_PACRE)	32	R/W	Undefined	20.2.3/453
4008_0044	Peripheral Access Control Register (AIPS1_PACRF)	32	R/W	Undefined	20.2.3/453
4008_0048	Peripheral Access Control Register (AIPS1_PACRG)	32	R/W	Undefined	20.2.3/453
4008_004C	Peripheral Access Control Register (AIPS1_PACRH)	32	R/W	Undefined	20.2.3/453
4008_0050	Peripheral Access Control Register (AIPS1_PACRI)	32	R/W	Undefined	20.2.3/453
4008_0054	Peripheral Access Control Register (AIPS1_PACRJ)	32	R/W	Undefined	20.2.3/453
4008_0058	Peripheral Access Control Register (AIPS1_PACRK)	32	R/W	Undefined	20.2.3/453
4008_005C	Peripheral Access Control Register (AIPS1_PACRL)	32	R/W	Undefined	20.2.3/453
4008_0060	Peripheral Access Control Register (AIPS1_PACRM)	32	R/W	Undefined	20.2.3/453
4008_0064	Peripheral Access Control Register (AIPS1_PACRN)	32	R/W	Undefined	20.2.3/453
4008_0068	Peripheral Access Control Register (AIPS1_PACRO)	32	R/W	Undefined	20.2.3/453
4008_006C	Peripheral Access Control Register (AIPS1_PACRP)	32	R/W	Undefined	20.2.3/453

20.2.1 Master Privilege Register A (AIPSx_MPRA)

The MPRA register specifies identical 4-bit fields defining the access-privilege level associated with a bus master in the device to the various peripherals. The register provides one field per bus master.

NOTE

At reset, the default value loaded into the MPROT[7-0] fields is device-specific. See the Chip Configuration details for the value on your particular device.

Accesses to registers or register fields which correspond to master or peripheral locations which are not implemented return zeros on reads, and are ignored on writes.

Each master is assigned depending on its connection to the crossbar switch master ports. See your device-specific Chip Configuration details for information about the master assignments to these registers.

Addresses: AIPS0_MPRA is 4000_0000h base + 0h offset = 4000_0000h
AIPS1_MPRA is 4008_0000h base + 0h offset = 4008_0000h

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	MTR0	MTW0	MPL0	0	MTR1	MTW1	MPL1	0	MTR2	MTW2	MPL2	0	MTR3	MTW3	MPL3	0	MTR4	MTW4	MPL4	0	MTR5	MTW5	MPL5	0	MTR6	MTW6	MPL6	0	MTR7	MTW7	MPL7
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.

AIPSx_MPRA field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30 MTR0	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
29 MTW0	Master trusted for writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
28 MPL0	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
27 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

AIPSx_MPRA field descriptions (continued)

Field	Description
26 MTR1	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
25 MTW1	Master trusted for writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
24 MPL1	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
23 Reserved	This read-only field is reserved and always has the value zero.
22 MTR2	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
21 MTW2	Master trusted for writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
20 MPL2	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
19 Reserved	This read-only field is reserved and always has the value zero.
18 MTR3	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
17 MTW3	Master trusted for writes Determines whether the master is trusted for write accesses.

Table continues on the next page...

AIPSx_MPRA field descriptions (continued)

Field	Description
	0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
16 MPL3	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
15 Reserved	This read-only field is reserved and always has the value zero.
14 MTR4	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
13 MTW4	Master trusted for writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
12 MPL4	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
11 Reserved	This read-only field is reserved and always has the value zero.
10 MTR5	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
9 MTW5	Master trusted for writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
8 MPL5	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
7 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

AIPSx_MPRA field descriptions (continued)

Field	Description
6 MTR6	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
5 MTW6	Master trusted for writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
4 MPL6	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.
3 Reserved	This read-only field is reserved and always has the value zero.
2 MTR7	Master trusted for read Determines whether the master is trusted for read accesses. 0 This master is not trusted for read accesses. 1 This master is trusted for read accesses.
1 MTW7	Master trusted for writes Determines whether the master is trusted for write accesses. 0 This master is not trusted for write accesses. 1 This master is trusted for write accesses.
0 MPL7	Master privilege level Specifies how the privilege level of the master is determined. 0 Accesses from this master are forced to user-mode. 1 Accesses from this master are not forced to user-mode.

20.2.2 Peripheral Access Control Register (AIPSx_PACR*n*)

Each of the peripherals has a four-bit PACR[0:127] field which defines the access levels supported by the given module. Eight PACR fields are grouped together to form a 32-bit PACR[A:P] register:

- PACRA-P define the access levels for the 128 peripherals

The peripheral assignments to each PACR register is defined by the memory map slot that the peripherals are assigned.See the device's Memory Map details for the assignments for your particular device.

NOTE

The reset value of the PACRA-D registers is 0x4444_4444.

The following table shows the top-level structure of the PACR registers.

Offset	Register	[31:28]	[27:24]	[23:20]	[19:16]	[15:12]	[11:8]	[7:4]	[3:0]
0x20	PACRA	PACR0	PACR1	PACR2	PACR3	PACR4	PACR5	PACR6	PACR7
0x24	PACRB	PACR8	PACR9	PACR10	PACR11	PACR12	PACR13	PACR14	PACR15
0x28	PACRC	PACR16	PACR17	PACR18	PACR19	PACR20	PACR21	PACR22	PACR23
0x2C	PACRD	PACR24	PACR25	PACR26	PACR27	PACR28	PACR29	PACR30	PACR31
0x30	Reserved								
0x34	Reserved								
0x38	Reserved								
0x3C	Reserved								
0x40	PACRE	PACR32	PACR33	PACR34	PACR35	PACR36	PACR37	PACR38	PACR39
0x44	PACRF	PACR40	PACR41	PACR42	PACR43	PACR44	PACR45	PACR46	PACR47
0x48	PACRG	PACR48	PACR49	PACR50	PACR51	PACR52	PACR53	PACR54	PACR55
0x4C	PACRH	PACR56	PACR57	PACR58	PACR59	PACR60	PACR61	PACR62	PACR63
0x50	PACRI	PACR64	PACR65	PACR66	PACR67	PACR68	PACR69	PACR70	PACR71
0x54	PACRJ	PACR72	PACR73	PACR74	PACR75	PACR76	PACR77	PACR78	PACR79
0x58	PACRK	PACR80	PACR81	PACR82	PACR83	PACR84	PACR85	PACR86	PACR87
0x5C	PACRL	PACR88	PACR89	PACR90	PACR91	PACR92	PACR93	PACR94	PACR95
0x60	PACRM	PACR96	PACR97	PACR98	PACR99	PACR100	PACR101	PACR102	PACR103
0x64	PACRN	PACR104	PACR105	PACR106	PACR107	PACR108	PACR109	PACR110	PACR111
0x68	PACRO	PACR112	PACR113	PACR114	PACR115	PACR116	PACR117	PACR118	PACR119
0x6C	PACRP	PACR120	PACR121	PACR122	PACR123	PACR124	PACR125	PACR126	PACR127

Addresses: AIPS0_PACRA is 4000_0000h base + 20h offset = 4000_0020h
AIPS0_PACRB is 4000_0000h base + 24h offset = 4000_0024h
AIPS0_PACRC is 4000_0000h base + 28h offset = 4000_0028h
AIPS0_PACRD is 4000_0000h base + 2Ch offset = 4000_002Ch
AIPS1_PACRA is 4008_0000h base + 20h offset = 4008_0020h
AIPS1_PACRB is 4008_0000h base + 24h offset = 4008_0024h
AIPS1_PACRC is 4008_0000h base + 28h offset = 4008_0028h
AIPS1_PACRD is 4008_0000h base + 2Ch offset = 4008_002Ch

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	SP0	WP0	TP0	0	SP1	WP1	TP1	0	SP2	WP2	TP2	0	SP3	WP3	TP3	0	SP4	WP4	TP4	0	SP5	WP5	TP5	0	SP6	WP6	TP6	0	SP7	WP7	TP7
W																																
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

AIPSx_PACR*n* field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30 SP0	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
29 WP0	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
28 TP0	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
27 Reserved	This read-only field is reserved and always has the value zero.
26 SP1	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPSx_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
25 WP1	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
24 TP1	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
23 Reserved	This read-only field is reserved and always has the value zero.
22 SP2	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
21 WP2	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
20 TP2	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
19 Reserved	This read-only field is reserved and always has the value zero.
18 SP3	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for

Table continues on the next page...

AIPSx_PACR*n* field descriptions (continued)

Field	Description
	the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
17 WP3	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
16 TP3	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
15 Reserved	This read-only field is reserved and always has the value zero.
14 SP4	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
13 WP4	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
12 TP4	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
11 Reserved	This read-only field is reserved and always has the value zero.
10 SP5	Supervisor protect

Table continues on the next page...

AIPSx_PACR*n* field descriptions (continued)

Field	Description
	Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
9 WP5	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
8 TP5	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
7 Reserved	This read-only field is reserved and always has the value zero.
6 SP6	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
5 WP6	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
4 TP6	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates . 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
3 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

AIPSx_PACR*n* field descriptions (continued)

Field	Description
2 SP7	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates . 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
1 WP7	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
0 TP7	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

20.2.3 Peripheral Access Control Register (AIPSx_PACR*n*)

Each of the peripherals has a four-bit PACR[0:127] field which defines the access levels supported by the given module. Eight PACR fields are grouped together to form a 32-bit PACR[A:P] register:

- PACRA-P define the access levels for the 128 peripherals

The peripheral assignments to each PACR register is defined by the memory map slot that the peripherals are assigned.See the device's Memory Map details for the assignments for your particular device.

NOTE

The reset value of the PACRE-P depends on your device's configuration.

Addresses: 4000_0000h base + 40h offset + (4d × *n*), where *n* = 0d to 11d

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	SP0	WP0	TP0	0	SP1	WP1	TP1	0	SP2	WP2	TP2	0	SP3	WP3	TP3	0	SP4	WP4	TP4	0	SP5	WP5	TP5	0	SP6	WP6	TP6	0	SP7	WP7	TP7
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.

AIPSx_PACR*n* field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30 SP0	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
29 WP0	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
28 TP0	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
27 Reserved	This read-only field is reserved and always has the value zero.
26 SP1	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
25 WP1	Write protect

Table continues on the next page...

AIPSx_PACR*n* field descriptions (continued)

Field	Description
	Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
24 TP1	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
23 Reserved	This read-only field is reserved and always has the value zero.
22 SP2	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
21 WP2	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
20 TP2	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
19 Reserved	This read-only field is reserved and always has the value zero.
18 SP3	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.

Table continues on the next page...

AIPSx_PACR*n* field descriptions (continued)

Field	Description
17 WP3	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
16 TP3	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
15 Reserved	This read-only field is reserved and always has the value zero.
14 SP4	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
13 WP4	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
12 TP4	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
11 Reserved	This read-only field is reserved and always has the value zero.
10 SP5	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates.

Table continues on the next page...

AIPSx_PACR*n* field descriptions (continued)

Field	Description
	0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
9 WP5	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
8 TP5	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
7 Reserved	This read-only field is reserved and always has the value zero.
6 SP6	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for the master must be set. If not, access terminates with an error response and no peripheral access initiates. 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
5 WP6	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
4 TP6	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates . 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.
3 Reserved	This read-only field is reserved and always has the value zero.
2 SP7	Supervisor protect Determines whether the peripheral requires supervisor privilege level for access. When this bit is set, the master privilege level must indicate the supervisor access attribute , and the MPROTn[MPL] control bit for

Table continues on the next page...

AIPSx_PACRn field descriptions (continued)

Field	Description
	the master must be set. If not, access terminates with an error response and no peripheral access initiates . 0 This peripheral does not require supervisor privilege level for accesses. 1 This peripheral requires supervisor privilege level for accesses.
1 WP7	Write protect Determines whether the peripheral allows write accesss. When this bit is set and a write access is attempted, access terminates with an error response and no peripheral access initiates. 0 This peripheral allows write accesses. 1 This peripheral is write protected.
0 TP7	Trusted protect Determines whether the peripheral allows accesses from an untrusted master. When this bit is set and an access is attempted by an untrusted master, the access terminates with an error response and no peripheral access initiates. 0 Accesses from an untrusted master are allowed. 1 Accesses from an untrusted master are not allowed.

20.3 Functional Description

The peripheral bridge serves as an interface between the crossbar switch and the slave peripheral bus. It functions as a protocol translator.

Accesses which fall within the address space of the peripheral bridge are decoded to provide individual module selects for peripheral devices on the slave bus interface.

20.3.1 Access support

Aligned and misaligned 32-bit and 16-bit accesses, as well as 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 which is larger (in size) than the targeted port, an error response is generated.

Chapter 21
Direct memory access multiplexer (DMAMUX)

21.1 Introduction

NOTE

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

21.1.1 Overview

The DMA Mux routes up to 63 DMA sources (called slots) to be mapped to any of the 16 DMA channels. This is illustrated in the following figure.

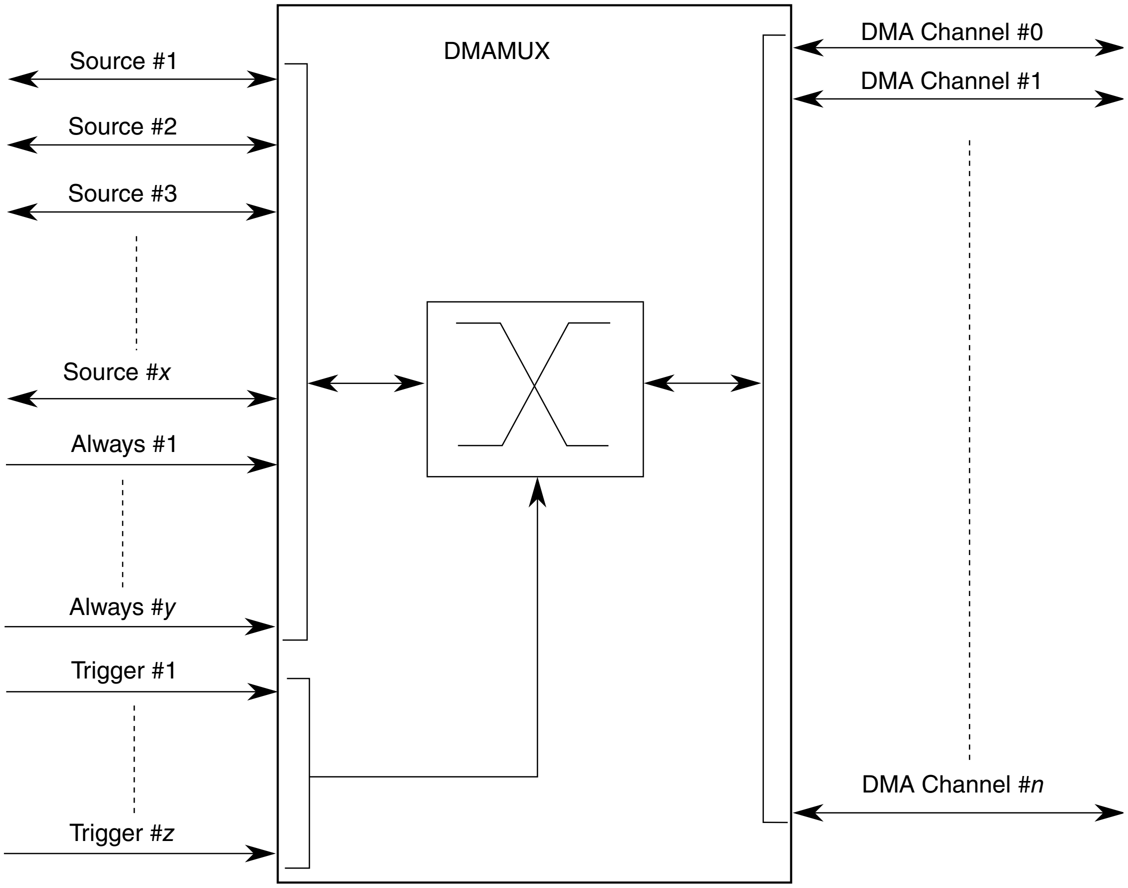


Figure 21-1. DMA MUX block diagram

21.1.2 Features

The DMA channel MUX provides these features:

- 52 peripheral slots + 10 always-on slots can be routed to 16 channels.
- 16 independently selectable DMA channel routers.
    - The first 4 channels additionally provide a trigger functionality.
- Each channel router can be assigned to one of the 52 possible peripheral DMA slots or to one of the 10 always-on slots.

21.1.3 Modes of operation

The following operating modes are available:

- Disabled mode

In this mode, the DMA channel is disabled. Since 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 (e.g. changing the period of a DMA trigger).

- Normal mode

  In this mode, a DMA source (such as DSPI transmit or DSPI receive) is routed directly to the specified DMA channel. The operation of the DMA MUX 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 only available for channels 0-3.

21.2 External signal description

The DMA MUX has no external pins.

21.3 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMA MUX.

The following table shows the memory map for the DMA MUX.

All registers are accessible via 8-bit, 16-bit or 32-bit accesses. However, 16-bit accesses must be aligned to 16-bit boundaries, and 32-bit accesses must be aligned to 32-bit boundaries. As an example, CHCFG0 through CHCFG3 are accessible by a 32-bit read/write to address 'base + 0x00', but performing a 32-bit access to address 'base + 0x01' is illegal.

DMAMUX memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_1000	Channel Configuration Register (DMAMUX0_CHCFG0)	8	R/W	00h	21.3.1/463
4002_1001	Channel Configuration Register (DMAMUX0_CHCFG1)	8	R/W	00h	21.3.1/463
4002_1002	Channel Configuration Register (DMAMUX0_CHCFG2)	8	R/W	00h	21.3.1/463
4002_1003	Channel Configuration Register (DMAMUX0_CHCFG3)	8	R/W	00h	21.3.1/463
4002_1004	Channel Configuration Register (DMAMUX0_CHCFG4)	8	R/W	00h	21.3.1/463
4002_1005	Channel Configuration Register (DMAMUX0_CHCFG5)	8	R/W	00h	21.3.1/463
4002_1006	Channel Configuration Register (DMAMUX0_CHCFG6)	8	R/W	00h	21.3.1/463
4002_1007	Channel Configuration Register (DMAMUX0_CHCFG7)	8	R/W	00h	21.3.1/463
4002_1008	Channel Configuration Register (DMAMUX0_CHCFG8)	8	R/W	00h	21.3.1/463
4002_1009	Channel Configuration Register (DMAMUX0_CHCFG9)	8	R/W	00h	21.3.1/463
4002_100A	Channel Configuration Register (DMAMUX0_CHCFG10)	8	R/W	00h	21.3.1/463
4002_100B	Channel Configuration Register (DMAMUX0_CHCFG11)	8	R/W	00h	21.3.1/463
4002_100C	Channel Configuration Register (DMAMUX0_CHCFG12)	8	R/W	00h	21.3.1/463
4002_100D	Channel Configuration Register (DMAMUX0_CHCFG13)	8	R/W	00h	21.3.1/463
4002_100E	Channel Configuration Register (DMAMUX0_CHCFG14)	8	R/W	00h	21.3.1/463
4002_100F	Channel Configuration Register (DMAMUX0_CHCFG15)	8	R/W	00h	21.3.1/463
4002_2000	Channel Configuration Register (DMAMUX1_CHCFG0)	8	R/W	00h	21.3.1/463
4002_2001	Channel Configuration Register (DMAMUX1_CHCFG1)	8	R/W	00h	21.3.1/463
4002_2002	Channel Configuration Register (DMAMUX1_CHCFG2)	8	R/W	00h	21.3.1/463
4002_2003	Channel Configuration Register (DMAMUX1_CHCFG3)	8	R/W	00h	21.3.1/463
4002_2004	Channel Configuration Register (DMAMUX1_CHCFG4)	8	R/W	00h	21.3.1/463
4002_2005	Channel Configuration Register (DMAMUX1_CHCFG5)	8	R/W	00h	21.3.1/463
4002_2006	Channel Configuration Register (DMAMUX1_CHCFG6)	8	R/W	00h	21.3.1/463
4002_2007	Channel Configuration Register (DMAMUX1_CHCFG7)	8	R/W	00h	21.3.1/463
4002_2008	Channel Configuration Register (DMAMUX1_CHCFG8)	8	R/W	00h	21.3.1/463
4002_2009	Channel Configuration Register (DMAMUX1_CHCFG9)	8	R/W	00h	21.3.1/463
4002_200A	Channel Configuration Register (DMAMUX1_CHCFG10)	8	R/W	00h	21.3.1/463
4002_200B	Channel Configuration Register (DMAMUX1_CHCFG11)	8	R/W	00h	21.3.1/463
4002_200C	Channel Configuration Register (DMAMUX1_CHCFG12)	8	R/W	00h	21.3.1/463
4002_200D	Channel Configuration Register (DMAMUX1_CHCFG13)	8	R/W	00h	21.3.1/463
4002_200E	Channel Configuration Register (DMAMUX1_CHCFG14)	8	R/W	00h	21.3.1/463
4002_200F	Channel Configuration Register (DMAMUX1_CHCFG15)	8	R/W	00h	21.3.1/463

21.3.1 Channel Configuration Register (DMAMUXx_CHCFG*n*)

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.

NOTE

Before changing the trigger or source settings a DMA channel must be disabled via the CHCFGn[ENBL] bit.

Addresses: 4002_1000h base + 0h offset + (1d × *n*), where *n* = 0d to 15d

Bit	7	6	5	4	3	2	1	0
Read Write	ENBL	TRIG	SOURCE					
Reset	0	0	0	0	0	0	0	0

DMAMUXx_CHCFG*n* field descriptions

Field	Description
7 ENBL	DMA Channel Enable Enables the DMA channel 0 DMA channel is disabled. This mode is primarily used during configuration of the DMA Mux. The DMA has separate channel enables/disables, which should be used to disable or re-configure a DMA channel. 1 DMA channel is enabled
6 TRIG	DMA Channel Trigger Enable Enables the periodic trigger capability for the triggered DMA channel 0 Triggering is disabled. If triggering is disabled, and the ENBL bit is set, the DMA Channel will simply route the specified source to the DMA channel. (normal mode) 1 Triggering is enabled. If triggering is enabled, and the ENBL bit is set, the DMAMUX is in periodic trigger mode.
5–0 SOURCE	DMA Channel Source (slot) Specifies which DMA source, if any, is routed to a particular DMA channel. Please check your device's Chip Configuration details for further details about the peripherals and their slot numbers.

21.4 Functional description

This section provides the functional description of the DMA MUX.

The primary purpose of the DMA MUX is to provide flexibility in the system's use of the available DMA channels. As such, configuration of the DMA MUX 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 DMA MUX may be changed during the normal operation of the system.

Functionally, the DMA MUX channels may be divided into two classes: Channels, which implement the normal routing functionality plus periodic triggering capability, and channels, which implement only the normal routing functionality.

21.4.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first four channels of the DMA MUX 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. Please refer to Periodic Interrupt Timer chapter for more information on this topic.

Note

> Because of the dynamic nature of the system (i.e. 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.

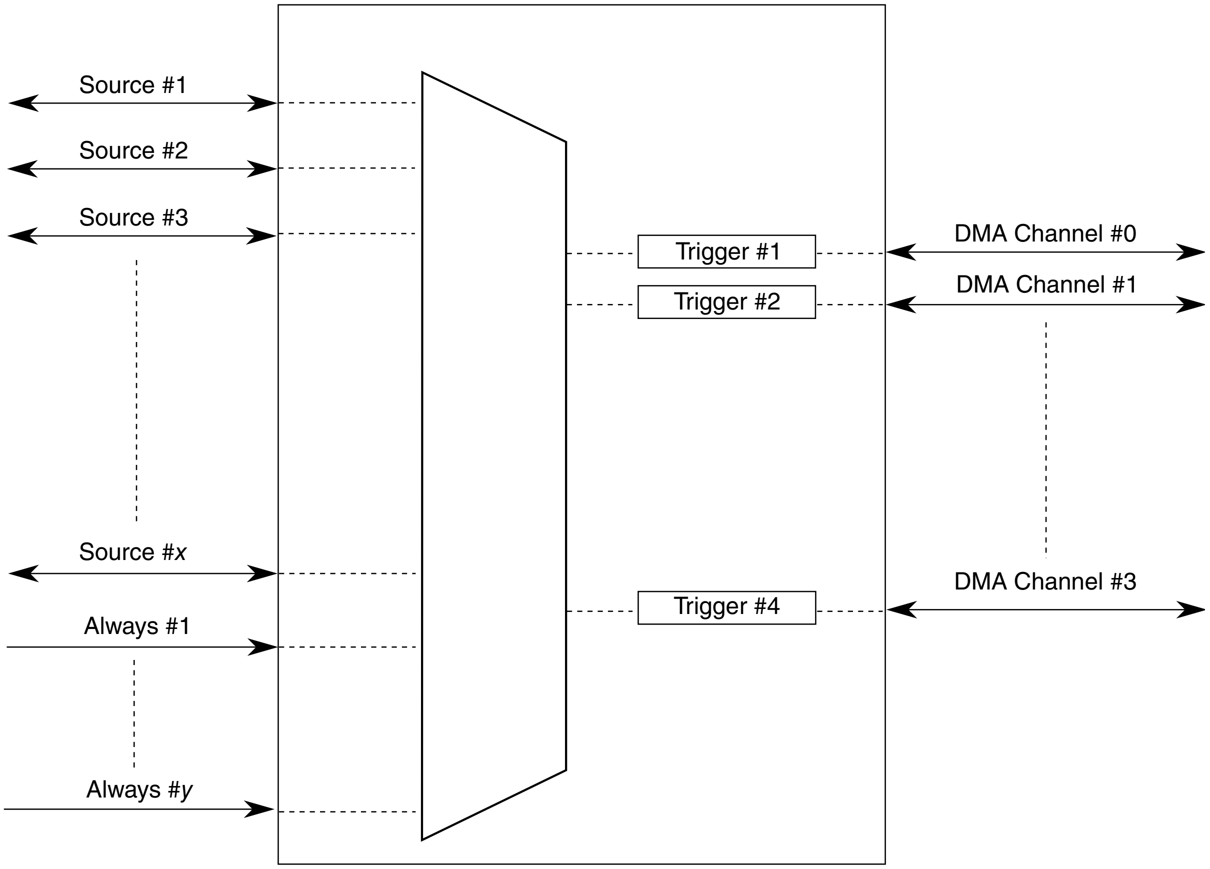


Figure 21-53. DMA MUX triggered channels

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.

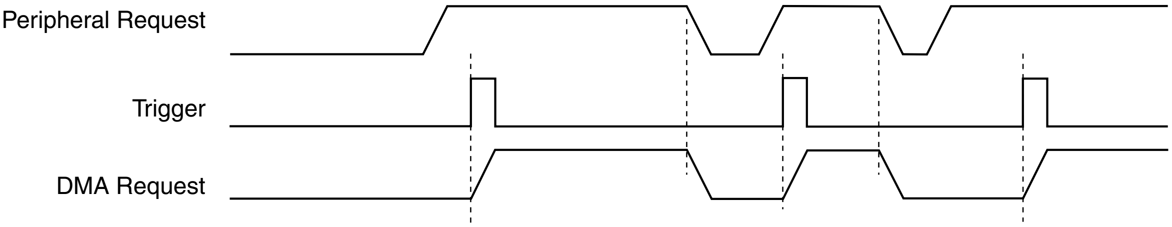


Figure 21-54. DMA MUX channel triggering: normal operation

Once the DMA request has been serviced, the peripheral will negate its request, effectively resetting the gating mechanism until the peripheral re-asserts 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, that triggered will be ignored. This situation is illustrated in the following figure.

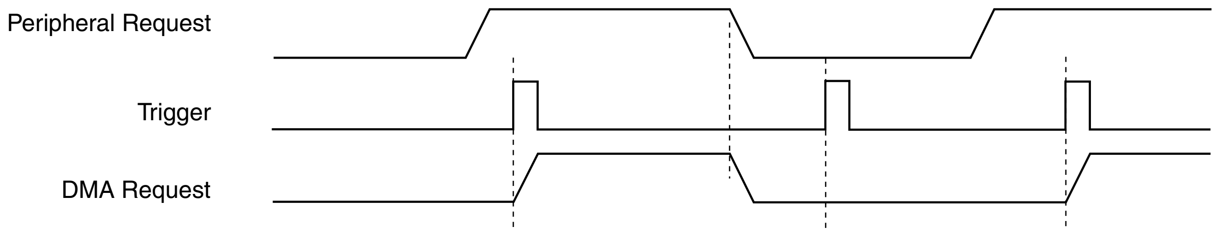


Figure 21-55. DMA MUX channel triggering: ignored trigger

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. Once setup, 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 5μs (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 (which includes resolution, range of values, etc.) may be found in the Periodic Interrupt Timer chapter.

21.4.2 DMA channels with no triggering capability

The other channels of the DMA MUX provide the normal routing functionality as described in Modes of operation.

21.4.3 "Always enabled" DMA sources

In addition to the peripherals that can be used as DMA sources, there are 10 additional DMA sources that are "always enabled". Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the "always enabled" sources provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Doing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins, either un-throttled (that is as fast as possible), or periodically (using the DMA triggering capability).
- Doing DMA transfers from memory to memory—Moving data from memory to memory, typically as fast as possible, sometimes with software activation.
- Doing 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 source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require 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 re-activation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will incur on the system. For this option, the DMA channel should be disabled in the DMA channel MUX.
- Use explicit software re-activation. In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to re-activate the channel (by writing to the DMA registers) *after every minor loop*. For this option, the DMA channel should be disabled in the DMA channel MUX.
- Use a "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 re-activation. For this option, the DMA channel should be enabled and pointing to an "always enabled" source. Note that the re-activation 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.

21.5 Initialization/application information

This section provides instructions for initializing the DMA channel MUX.

21.5.1 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.

21.5.2 Enabling and configuring sources

Enabling a source with periodic triggering

1. Determine with which DMA channel the source will be associated. Note that only the first 4 DMA channels have periodic triggering capability
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits 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] bits are set

Configure source #5 transmit for use with DMA channel 2, with periodic triggering capability

1. Write 0x00 to CHCFG2 (base address + 0x02)
2. Configure channel 2 in the DMA, including enabling the channel
3. Configure a timer for the desired trigger interval
4. Write 0xC5 to CHCFG2 (base address + 0x02)

The following code example illustrates steps #1 and #4 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0xFC084000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCONFIG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCONFIG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCONFIG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCONFIG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCONFIG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCONFIG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCONFIG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCONFIG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCONFIG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCONFIG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCONFIG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCONFIG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCONFIG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCONFIG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCONFIG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCONFIG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
```

```
:
*CHCONFIG2 = 0x00;
*CHCONFIG2 = 0xC5;
```

Enabling a source without periodic triggering

1. Determine with which DMA channel the source will be associated. Note that only the first 4 DMA channels have periodic triggering capability
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits 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 the CHCFG[ENBL] is set while the CHCFG[TRIG] bit is cleared

Configure source #5 Transmit for use with DMA channel 2, with no periodic triggering capability.

1. Write 0x00 to CHCFG2 (base address + 0x02)
2. Configure channel 2 in the DMA, including enabling the channel
3. Write 0x85 to CHCFG2 (base address + 0x02)

The following code example illustrates steps #1 and #3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0xFC084000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCONFIG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCONFIG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCONFIG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCONFIG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCONFIG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCONFIG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCONFIG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCONFIG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCONFIG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCONFIG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCONFIG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCONFIG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCONFIG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCONFIG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCONFIG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCONFIG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCONFIG2 = 0x00;
*CHCONFIG2 = 0x85;
```

Disabling 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. Please refer to the appropriate section for more details.

Switching the source of a DMA channel

1. Disable the DMA channel in the DMA and re-configure 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] bits are set

Switch DMA channel 8 from source #5 transmit to source #7 transmit

1. In the DMA configuration registers, disable DMA channel 8 and re-configure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability
2. Write 0x00 to CHCFG8 (base address + 0x08)
3. Write 0x87 to CHCFG8 (base address + 0x08). (In this example, setting the CHCFG[TRIG] bit would have no effect, due to the assumption that channels 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     0xFC084000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCONFIG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCONFIG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCONFIG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCONFIG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCONFIG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCONFIG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCONFIG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCONFIG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCONFIG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCONFIG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCONFIG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCONFIG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCONFIG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCONFIG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCONFIG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCONFIG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCONFIG8 = 0x00;
*CHCONFIG8 = 0x87;
```

Chapter 22
Direct Memory Access Controller (eDMA)

22.1 Introduction

NOTE

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

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- and destination-address calculations
    - Data-movement operations
- Local memory containing transfer control descriptors for each of the 32 channels

22.1.1 Block diagram

This diagram illustrates the eDMA module.

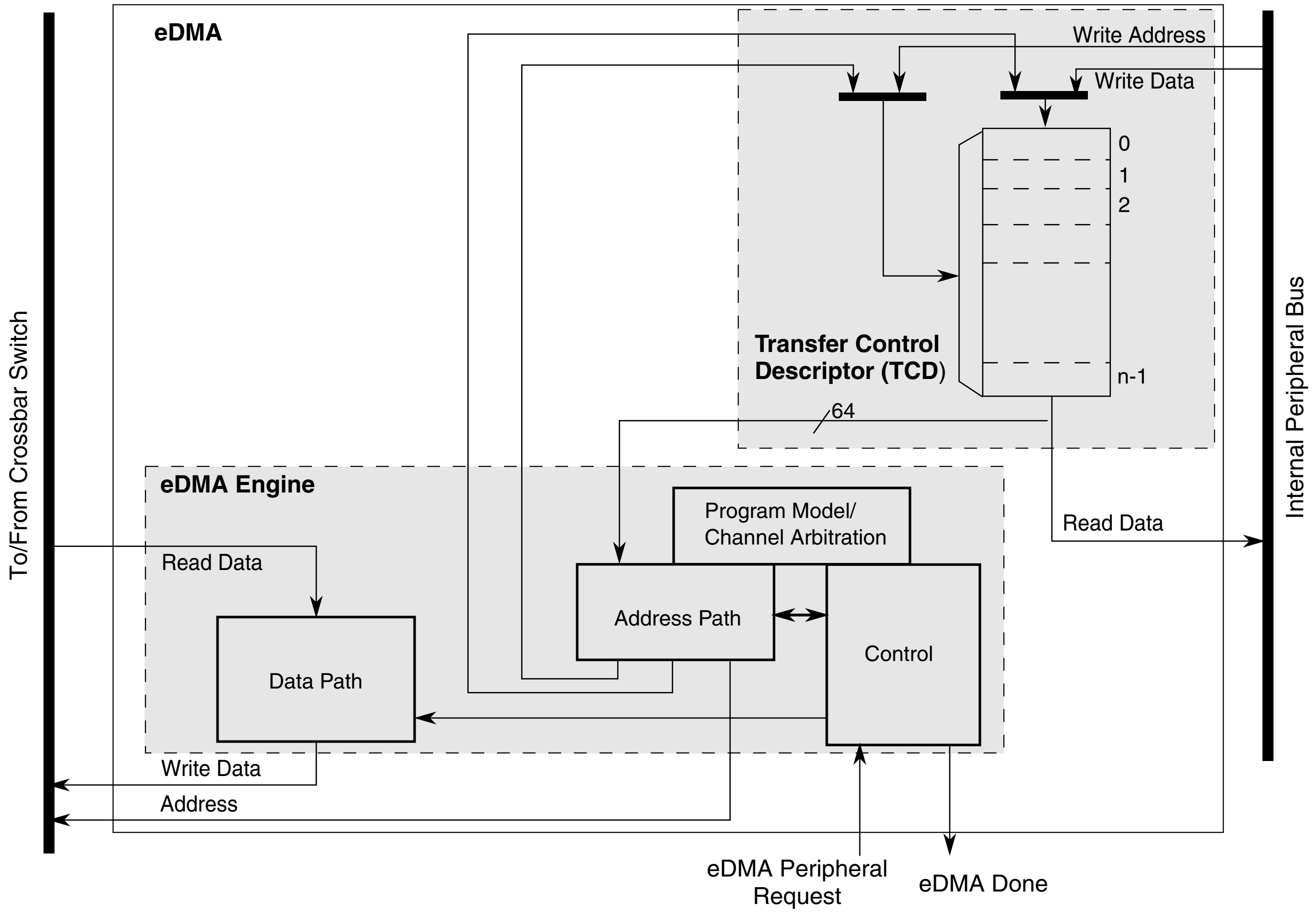


Figure 22-1. eDMA block diagram

22.1.2 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 22-1. 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 16 bytes of register storage 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 22-2. 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 simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage is implemented using a single-port, synchronous RAM array.

22.1.3 Features

The eDMA is a highly-programmable data-transfer engine optimized to minimize the 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 data packet 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-byte burst 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 optional interrupt requests
  - One interrupt per channel, optionally asserted at completion of major iteration count
  - Optional error terminations per channel and logically summed together to form one error interrupt to the interrupt controller
- Optional support for scatter/gather DMA processing
- Support for complex data structures
- Support to cancel transfers via software

In the discussion of this module, *n* is used to reference the channel number.

22.2 Modes of operation

The eDMA operates in the following modes:

Table 22-3. 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.

22.3 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 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.

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.

DMA memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_8000	Control Register (DMA_CR)	32	R/W	0000_0000h	22.3.1/503
4000_8004	Error Status Register (DMA_ES)	32	R	0000_0000h	22.3.2/505
4000_800C	Enable Request Register (DMA_ERQ)	32	R/W	0000_0000h	22.3.3/508
4000_8014	Enable Error Interrupt Register (DMA_EEI)	32	R/W	0000_0000h	22.3.4/511
4000_8018	Clear Enable Error Interrupt Register (DMA_CEEI)	8	W (always reads zero)	00h	22.3.5/515
4000_8019	Set Enable Error Interrupt Register (DMA_SEEI)	8	W (always reads zero)	00h	22.3.6/516
4000_801A	Clear Enable Request Register (DMA_CERQ)	8	W (always reads zero)	00h	22.3.7/517
4000_801B	Set Enable Request Register (DMA_SERQ)	8	W (always reads zero)	00h	22.3.8/518
4000_801C	Clear DONE Status Bit Register (DMA_CDNE)	8	W (always reads zero)	00h	22.3.9/519
4000_801D	Set START Bit Register (DMA_SSRT)	8	W (always reads zero)	00h	22.3.10/ 520

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_801E	Clear Error Register (DMA_CERR)	8	W (always reads zero)	00h	22.3.11/ 521
4000_801F	Clear Interrupt Request Register (DMA_CINT)	8	W (always reads zero)	00h	22.3.12/ 522
4000_8024	Interrupt Request Register (DMA_INT)	32	R/W	0000_0000h	22.3.13/ 522
4000_802C	Error Register (DMA_ERR)	32	R/W	0000_0000h	22.3.14/ 526
4000_8034	Hardware Request Status Register (DMA_HRS)	32	R/W	0000_0000h	22.3.15/ 530
4000_8100	Channel n Priority Register (DMA_DCHPRI3)	8	R/W	Undefined	22.3.16/ 534
4000_8101	Channel n Priority Register (DMA_DCHPRI2)	8	R/W	Undefined	22.3.16/ 534
4000_8102	Channel n Priority Register (DMA_DCHPRI1)	8	R/W	Undefined	22.3.16/ 534
4000_8103	Channel n Priority Register (DMA_DCHPRI0)	8	R/W	Undefined	22.3.16/ 534
4000_8104	Channel n Priority Register (DMA_DCHPRI7)	8	R/W	Undefined	22.3.16/ 534
4000_8105	Channel n Priority Register (DMA_DCHPRI6)	8	R/W	Undefined	22.3.16/ 534
4000_8106	Channel n Priority Register (DMA_DCHPRI5)	8	R/W	Undefined	22.3.16/ 534
4000_8107	Channel n Priority Register (DMA_DCHPRI4)	8	R/W	Undefined	22.3.16/ 534
4000_8108	Channel n Priority Register (DMA_DCHPRI11)	8	R/W	Undefined	22.3.16/ 534
4000_8109	Channel n Priority Register (DMA_DCHPRI10)	8	R/W	Undefined	22.3.16/ 534
4000_810A	Channel n Priority Register (DMA_DCHPRI9)	8	R/W	Undefined	22.3.16/ 534
4000_810B	Channel n Priority Register (DMA_DCHPRI8)	8	R/W	Undefined	22.3.16/ 534
4000_810C	Channel n Priority Register (DMA_DCHPRI15)	8	R/W	Undefined	22.3.16/ 534
4000_810D	Channel n Priority Register (DMA_DCHPRI14)	8	R/W	Undefined	22.3.16/ 534

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_810E	Channel n Priority Register (DMA_DCHPRI13)	8	R/W	Undefined	22.3.16/ 534
4000_810F	Channel n Priority Register (DMA_DCHPRI12)	8	R/W	Undefined	22.3.16/ 534
4000_8110	Channel n Priority Register (DMA_DCHPRI19)	8	R/W	Undefined	22.3.16/ 534
4000_8111	Channel n Priority Register (DMA_DCHPRI18)	8	R/W	Undefined	22.3.16/ 534
4000_8112	Channel n Priority Register (DMA_DCHPRI17)	8	R/W	Undefined	22.3.16/ 534
4000_8113	Channel n Priority Register (DMA_DCHPRI16)	8	R/W	Undefined	22.3.16/ 534
4000_8114	Channel n Priority Register (DMA_DCHPRI23)	8	R/W	Undefined	22.3.16/ 534
4000_8115	Channel n Priority Register (DMA_DCHPRI22)	8	R/W	Undefined	22.3.16/ 534
4000_8116	Channel n Priority Register (DMA_DCHPRI21)	8	R/W	Undefined	22.3.16/ 534
4000_8117	Channel n Priority Register (DMA_DCHPRI20)	8	R/W	Undefined	22.3.16/ 534
4000_8118	Channel n Priority Register (DMA_DCHPRI27)	8	R/W	Undefined	22.3.16/ 534
4000_8119	Channel n Priority Register (DMA_DCHPRI26)	8	R/W	Undefined	22.3.16/ 534
4000_811A	Channel n Priority Register (DMA_DCHPRI25)	8	R/W	Undefined	22.3.16/ 534
4000_811B	Channel n Priority Register (DMA_DCHPRI24)	8	R/W	Undefined	22.3.16/ 534
4000_811C	Channel n Priority Register (DMA_DCHPRI31)	8	R/W	Undefined	22.3.16/ 534
4000_811D	Channel n Priority Register (DMA_DCHPRI30)	8	R/W	Undefined	22.3.16/ 534
4000_811E	Channel n Priority Register (DMA_DCHPRI29)	8	R/W	Undefined	22.3.16/ 534
4000_811F	Channel n Priority Register (DMA_DCHPRI28)	8	R/W	Undefined	22.3.16/ 534
4000_9000	TCD Source Address (DMA_TCD0_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9004	TCD Signed Source Address Offset (DMA_TCD0_SOFF)	16	R/W	Undefined	22.3.18/ 535

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9006	TCD Transfer Attributes (DMA_TCD0_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9008	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD0_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9008	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD0_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9008	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD0_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_900C	TCD Last Source Address Adjustment (DMA_TCD0_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9010	TCD Destination Address (DMA_TCD0_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9014	TCD Signed Destination Address Offset (DMA_TCD0_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9016	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9016	DMA_TCD0_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9018	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD0_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_901C	TCD Control and Status (DMA_TCD0_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD0_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_901E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD0_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9020	TCD Source Address (DMA_TCD1_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9024	TCD Signed Source Address Offset (DMA_TCD1_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9026	TCD Transfer Attributes (DMA_TCD1_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9028	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD1_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9028	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD1_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9028	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD1_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_902C	TCD Last Source Address Adjustment (DMA_TCD1_SLAST)	32	R/W	Undefined	22.3.23/ 539

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9030	TCD Destination Address (DMA_TCD1_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9034	TCD Signed Destination Address Offset (DMA_TCD1_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9036	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9036	DMA_TCD1_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9038	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD1_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_903C	TCD Control and Status (DMA_TCD1_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD1_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_903E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD1_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9040	TCD Source Address (DMA_TCD2_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9044	TCD Signed Source Address Offset (DMA_TCD2_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9046	TCD Transfer Attributes (DMA_TCD2_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9048	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD2_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9048	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD2_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9048	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD2_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_904C	TCD Last Source Address Adjustment (DMA_TCD2_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9050	TCD Destination Address (DMA_TCD2_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9054	TCD Signed Destination Address Offset (DMA_TCD2_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9056	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9056	DMA_TCD2_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9058	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD2_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_905C	TCD Control and Status (DMA_TCD2_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD2_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_905E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD2_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9060	TCD Source Address (DMA_TCD3_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9064	TCD Signed Source Address Offset (DMA_TCD3_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9066	TCD Transfer Attributes (DMA_TCD3_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9068	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD3_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9068	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD3_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9068	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD3_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_906C	TCD Last Source Address Adjustment (DMA_TCD3_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9070	TCD Destination Address (DMA_TCD3_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9074	TCD Signed Destination Address Offset (DMA_TCD3_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9076	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9076	DMA_TCD3_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9078	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD3_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_907C	TCD Control and Status (DMA_TCD3_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD3_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_907E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD3_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9080	TCD Source Address (DMA_TCD4_SADDR)	32	R/W	Undefined	22.3.17/ 535

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9084	TCD Signed Source Address Offset (DMA_TCD4_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9086	TCD Transfer Attributes (DMA_TCD4_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9088	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD4_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9088	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD4_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9088	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD4_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_908C	TCD Last Source Address Adjustment (DMA_TCD4_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9090	TCD Destination Address (DMA_TCD4_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9094	TCD Signed Destination Address Offset (DMA_TCD4_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9096	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9096	DMA_TCD4_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9098	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD4_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_909C	TCD Control and Status (DMA_TCD4_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD4_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_909E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD4_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_90A0	TCD Source Address (DMA_TCD5_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_90A4	TCD Signed Source Address Offset (DMA_TCD5_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_90A6	TCD Transfer Attributes (DMA_TCD5_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_90A8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD5_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD5_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_90A8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD5_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90AC	TCD Last Source Address Adjustment (DMA_TCD5_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_90B0	TCD Destination Address (DMA_TCD5_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_90B4	TCD Signed Destination Address Offset (DMA_TCD5_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_90B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_90B6	DMA_TCD5_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_90B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD5_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_90BC	TCD Control and Status (DMA_TCD5_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD5_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_90BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD5_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_90C0	TCD Source Address (DMA_TCD6_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_90C4	TCD Signed Source Address Offset (DMA_TCD6_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_90C6	TCD Transfer Attributes (DMA_TCD6_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_90C8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD6_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD6_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_90C8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD6_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_90CC	TCD Last Source Address Adjustment (DMA_TCD6_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_90D0	TCD Destination Address (DMA_TCD6_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_90D4	TCD Signed Destination Address Offset (DMA_TCD6_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_90D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_90D6	DMA_TCD6_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_90D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD6_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_90DC	TCD Control and Status (DMA_TCD6_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD6_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_90DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD6_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_90E0	TCD Source Address (DMA_TCD7_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_90E4	TCD Signed Source Address Offset (DMA_TCD7_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_90E6	TCD Transfer Attributes (DMA_TCD7_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_90E8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD7_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD7_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_90E8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD7_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_90EC	TCD Last Source Address Adjustment (DMA_TCD7_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_90F0	TCD Destination Address (DMA_TCD7_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_90F4	TCD Signed Destination Address Offset (DMA_TCD7_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_90F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_90F6	DMA_TCD7_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_90F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD7_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_90FC	TCD Control and Status (DMA_TCD7_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD7_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_90FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD7_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9100	TCD Source Address (DMA_TCD8_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9104	TCD Signed Source Address Offset (DMA_TCD8_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9106	TCD Transfer Attributes (DMA_TCD8_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9108	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD8_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9108	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD8_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9108	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD8_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_910C	TCD Last Source Address Adjustment (DMA_TCD8_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9110	TCD Destination Address (DMA_TCD8_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9114	TCD Signed Destination Address Offset (DMA_TCD8_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9116	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9116	DMA_TCD8_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9118	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD8_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_911C	TCD Control and Status (DMA_TCD8_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_911E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD8_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_911E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD8_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9120	TCD Source Address (DMA_TCD9_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9124	TCD Signed Source Address Offset (DMA_TCD9_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9126	TCD Transfer Attributes (DMA_TCD9_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9128	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD9_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9128	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD9_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9128	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD9_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_912C	TCD Last Source Address Adjustment (DMA_TCD9_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9130	TCD Destination Address (DMA_TCD9_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9134	TCD Signed Destination Address Offset (DMA_TCD9_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9136	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9136	DMA_TCD9_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9138	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD9_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_913C	TCD Control and Status (DMA_TCD9_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_913E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD9_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_913E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD9_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9140	TCD Source Address (DMA_TCD10_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9144	TCD Signed Source Address Offset (DMA_TCD10_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9146	TCD Transfer Attributes (DMA_TCD10_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9148	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD10_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9148	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD10_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9148	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD10_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_914C	TCD Last Source Address Adjustment (DMA_TCD10_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9150	TCD Destination Address (DMA_TCD10_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9154	TCD Signed Destination Address Offset (DMA_TCD10_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9156	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9156	DMA_TCD10_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9158	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD10_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_915C	TCD Control and Status (DMA_TCD10_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_915E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD10_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_915E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD10_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9160	TCD Source Address (DMA_TCD11_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9164	TCD Signed Source Address Offset (DMA_TCD11_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9166	TCD Transfer Attributes (DMA_TCD11_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9168	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD11_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9168	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD11_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9168	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD11_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_916C	TCD Last Source Address Adjustment (DMA_TCD11_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9170	TCD Destination Address (DMA_TCD11_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9174	TCD Signed Destination Address Offset (DMA_TCD11_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9176	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9176	DMA_TCD11_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9178	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD11_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_917C	TCD Control and Status (DMA_TCD11_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_917E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD11_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_917E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD11_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9180	TCD Source Address (DMA_TCD12_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9184	TCD Signed Source Address Offset (DMA_TCD12_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9186	TCD Transfer Attributes (DMA_TCD12_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9188	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD12_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9188	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD12_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9188	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD12_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_918C	TCD Last Source Address Adjustment (DMA_TCD12_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9190	TCD Destination Address (DMA_TCD12_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9194	TCD Signed Destination Address Offset (DMA_TCD12_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9196	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9196	DMA_TCD12_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9198	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD12_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_919C	TCD Control and Status (DMA_TCD12_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_919E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD12_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_919E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD12_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_91A0	TCD Source Address (DMA_TCD13_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_91A4	TCD Signed Source Address Offset (DMA_TCD13_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_91A6	TCD Transfer Attributes (DMA_TCD13_ATTR)	16	R/W	Undefined	22.3.19/ 536

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_91A8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD13_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_91A8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD13_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_91A8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD13_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_91AC	TCD Last Source Address Adjustment (DMA_TCD13_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_91B0	TCD Destination Address (DMA_TCD13_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_91B4	TCD Signed Destination Address Offset (DMA_TCD13_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_91B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_91B6	DMA_TCD13_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_91B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD13_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_91BC	TCD Control and Status (DMA_TCD13_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_91BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD13_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_91BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD13_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_91C0	TCD Source Address (DMA_TCD14_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_91C4	TCD Signed Source Address Offset (DMA_TCD14_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_91C6	TCD Transfer Attributes (DMA_TCD14_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_91C8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD14_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_91C8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD14_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_91C8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD14_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_91CC	TCD Last Source Address Adjustment (DMA_TCD14_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_91D0	TCD Destination Address (DMA_TCD14_DADDR)	32	R/W	Undefined	22.3.24/ 540

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_91D4	TCD Signed Destination Address Offset (DMA_TCD14_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_91D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_91D6	DMA_TCD14_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_91D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD14_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_91DC	TCD Control and Status (DMA_TCD14_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_91DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD14_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_91DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD14_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_91E0	TCD Source Address (DMA_TCD15_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_91E4	TCD Signed Source Address Offset (DMA_TCD15_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_91E6	TCD Transfer Attributes (DMA_TCD15_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_91E8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD15_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_91E8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD15_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_91E8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD15_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_91EC	TCD Last Source Address Adjustment (DMA_TCD15_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_91F0	TCD Destination Address (DMA_TCD15_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_91F4	TCD Signed Destination Address Offset (DMA_TCD15_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_91F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_91F6	DMA_TCD15_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_91F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD15_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_91FC	TCD Control and Status (DMA_TCD15_CSR)	16	R/W	Undefined	22.3.29/ 543

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_91FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD15_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_91FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD15_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9200	TCD Source Address (DMA_TCD16_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9204	TCD Signed Source Address Offset (DMA_TCD16_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9206	TCD Transfer Attributes (DMA_TCD16_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9208	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD16_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9208	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD16_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9208	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD16_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_920C	TCD Last Source Address Adjustment (DMA_TCD16_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9210	TCD Destination Address (DMA_TCD16_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9214	TCD Signed Destination Address Offset (DMA_TCD16_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9216	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD16_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9216	DMA_TCD16_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9218	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD16_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_921C	TCD Control and Status (DMA_TCD16_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_921E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD16_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_921E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD16_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9220	TCD Source Address (DMA_TCD17_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9224	TCD Signed Source Address Offset (DMA_TCD17_SOFF)	16	R/W	Undefined	22.3.18/ 535

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9226	TCD Transfer Attributes (DMA_TCD17_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9228	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD17_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9228	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD17_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9228	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD17_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_922C	TCD Last Source Address Adjustment (DMA_TCD17_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9230	TCD Destination Address (DMA_TCD17_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9234	TCD Signed Destination Address Offset (DMA_TCD17_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9236	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD17_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9236	DMA_TCD17_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9238	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD17_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_923C	TCD Control and Status (DMA_TCD17_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_923E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD17_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_923E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD17_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9240	TCD Source Address (DMA_TCD18_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9244	TCD Signed Source Address Offset (DMA_TCD18_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9246	TCD Transfer Attributes (DMA_TCD18_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9248	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD18_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9248	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD18_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9248	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD18_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_924C	TCD Last Source Address Adjustment (DMA_TCD18_SLAST)	32	R/W	Undefined	22.3.23/ 539

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9250	TCD Destination Address (DMA_TCD18_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9254	TCD Signed Destination Address Offset (DMA_TCD18_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9256	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD18_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9256	DMA_TCD18_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9258	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD18_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_925C	TCD Control and Status (DMA_TCD18_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_925E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD18_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_925E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD18_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9260	TCD Source Address (DMA_TCD19_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9264	TCD Signed Source Address Offset (DMA_TCD19_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9266	TCD Transfer Attributes (DMA_TCD19_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9268	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD19_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9268	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD19_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9268	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD19_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_926C	TCD Last Source Address Adjustment (DMA_TCD19_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9270	TCD Destination Address (DMA_TCD19_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9274	TCD Signed Destination Address Offset (DMA_TCD19_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9276	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD19_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9276	DMA_TCD19_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9278	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD19_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_927C	TCD Control and Status (DMA_TCD19_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_927E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD19_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_927E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD19_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9280	TCD Source Address (DMA_TCD20_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9284	TCD Signed Source Address Offset (DMA_TCD20_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9286	TCD Transfer Attributes (DMA_TCD20_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9288	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD20_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9288	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD20_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9288	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD20_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_928C	TCD Last Source Address Adjustment (DMA_TCD20_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9290	TCD Destination Address (DMA_TCD20_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9294	TCD Signed Destination Address Offset (DMA_TCD20_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9296	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD20_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9296	DMA_TCD20_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9298	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD20_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_929C	TCD Control and Status (DMA_TCD20_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_929E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD20_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_929E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD20_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_92A0	TCD Source Address (DMA_TCD21_SADDR)	32	R/W	Undefined	22.3.17/ 535

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_92A4	TCD Signed Source Address Offset (DMA_TCD21_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_92A6	TCD Transfer Attributes (DMA_TCD21_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_92A8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD21_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_92A8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD21_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_92A8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD21_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_92AC	TCD Last Source Address Adjustment (DMA_TCD21_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_92B0	TCD Destination Address (DMA_TCD21_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_92B4	TCD Signed Destination Address Offset (DMA_TCD21_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_92B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD21_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_92B6	DMA_TCD21_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_92B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD21_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_92BC	TCD Control and Status (DMA_TCD21_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_92BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD21_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_92BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD21_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_92C0	TCD Source Address (DMA_TCD22_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_92C4	TCD Signed Source Address Offset (DMA_TCD22_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_92C6	TCD Transfer Attributes (DMA_TCD22_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_92C8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD22_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_92C8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD22_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_92C8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD22_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_92CC	TCD Last Source Address Adjustment (DMA_TCD22_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_92D0	TCD Destination Address (DMA_TCD22_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_92D4	TCD Signed Destination Address Offset (DMA_TCD22_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_92D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD22_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_92D6	DMA_TCD22_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_92D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD22_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_92DC	TCD Control and Status (DMA_TCD22_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_92DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD22_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_92DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD22_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_92E0	TCD Source Address (DMA_TCD23_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_92E4	TCD Signed Source Address Offset (DMA_TCD23_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_92E6	TCD Transfer Attributes (DMA_TCD23_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_92E8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD23_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_92E8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD23_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_92E8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD23_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_92EC	TCD Last Source Address Adjustment (DMA_TCD23_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_92F0	TCD Destination Address (DMA_TCD23_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_92F4	TCD Signed Destination Address Offset (DMA_TCD23_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_92F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD23_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_92F6	DMA_TCD23_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_92F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD23_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_92FC	TCD Control and Status (DMA_TCD23_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_92FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD23_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_92FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD23_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9300	TCD Source Address (DMA_TCD24_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9304	TCD Signed Source Address Offset (DMA_TCD24_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9306	TCD Transfer Attributes (DMA_TCD24_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9308	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD24_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9308	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD24_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9308	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD24_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_930C	TCD Last Source Address Adjustment (DMA_TCD24_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9310	TCD Destination Address (DMA_TCD24_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9314	TCD Signed Destination Address Offset (DMA_TCD24_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9316	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD24_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9316	DMA_TCD24_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9318	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD24_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_931C	TCD Control and Status (DMA_TCD24_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_931E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD24_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_931E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD24_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9320	TCD Source Address (DMA_TCD25_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9324	TCD Signed Source Address Offset (DMA_TCD25_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9326	TCD Transfer Attributes (DMA_TCD25_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9328	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD25_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9328	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD25_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9328	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD25_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_932C	TCD Last Source Address Adjustment (DMA_TCD25_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9330	TCD Destination Address (DMA_TCD25_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9334	TCD Signed Destination Address Offset (DMA_TCD25_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9336	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD25_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9336	DMA_TCD25_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9338	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD25_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_933C	TCD Control and Status (DMA_TCD25_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_933E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD25_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_933E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD25_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9340	TCD Source Address (DMA_TCD26_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9344	TCD Signed Source Address Offset (DMA_TCD26_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9346	TCD Transfer Attributes (DMA_TCD26_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9348	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD26_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9348	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD26_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9348	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD26_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_934C	TCD Last Source Address Adjustment (DMA_TCD26_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9350	TCD Destination Address (DMA_TCD26_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9354	TCD Signed Destination Address Offset (DMA_TCD26_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9356	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD26_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9356	DMA_TCD26_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9358	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD26_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_935C	TCD Control and Status (DMA_TCD26_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_935E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD26_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_935E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD26_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9360	TCD Source Address (DMA_TCD27_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9364	TCD Signed Source Address Offset (DMA_TCD27_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9366	TCD Transfer Attributes (DMA_TCD27_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9368	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD27_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9368	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD27_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9368	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD27_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_936C	TCD Last Source Address Adjustment (DMA_TCD27_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9370	TCD Destination Address (DMA_TCD27_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9374	TCD Signed Destination Address Offset (DMA_TCD27_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9376	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD27_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_9376	DMA_TCD27_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9378	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD27_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_937C	TCD Control and Status (DMA_TCD27_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_937E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD27_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_937E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD27_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_9380	TCD Source Address (DMA_TCD28_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_9384	TCD Signed Source Address Offset (DMA_TCD28_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_9386	TCD Transfer Attributes (DMA_TCD28_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_9388	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD28_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_9388	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD28_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_9388	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD28_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_938C	TCD Last Source Address Adjustment (DMA_TCD28_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_9390	TCD Destination Address (DMA_TCD28_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_9394	TCD Signed Destination Address Offset (DMA_TCD28_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_9396	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD28_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_9396	DMA_TCD28_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_9398	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD28_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_939C	TCD Control and Status (DMA_TCD28_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_939E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD28_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_939E	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD28_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_93A0	TCD Source Address (DMA_TCD29_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_93A4	TCD Signed Source Address Offset (DMA_TCD29_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_93A6	TCD Transfer Attributes (DMA_TCD29_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_93A8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD29_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_93A8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD29_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_93A8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD29_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_93AC	TCD Last Source Address Adjustment (DMA_TCD29_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_93B0	TCD Destination Address (DMA_TCD29_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_93B4	TCD Signed Destination Address Offset (DMA_TCD29_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_93B6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD29_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_93B6	DMA_TCD29_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_93B8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD29_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_93BC	TCD Control and Status (DMA_TCD29_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_93BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD29_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_93BE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD29_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_93C0	TCD Source Address (DMA_TCD30_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_93C4	TCD Signed Source Address Offset (DMA_TCD30_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_93C6	TCD Transfer Attributes (DMA_TCD30_ATTR)	16	R/W	Undefined	22.3.19/ 536

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_93C8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD30_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_93C8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD30_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_93C8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD30_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_93CC	TCD Last Source Address Adjustment (DMA_TCD30_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_93D0	TCD Destination Address (DMA_TCD30_DADDR)	32	R/W	Undefined	22.3.24/ 540
4000_93D4	TCD Signed Destination Address Offset (DMA_TCD30_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_93D6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD30_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_93D6	DMA_TCD30_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_93D8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD30_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_93DC	TCD Control and Status (DMA_TCD30_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_93DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD30_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_93DE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD30_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547
4000_93E0	TCD Source Address (DMA_TCD31_SADDR)	32	R/W	Undefined	22.3.17/ 535
4000_93E4	TCD Signed Source Address Offset (DMA_TCD31_SOFF)	16	R/W	Undefined	22.3.18/ 535
4000_93E6	TCD Transfer Attributes (DMA_TCD31_ATTR)	16	R/W	Undefined	22.3.19/ 536
4000_93E8	TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD31_NBYTES_MLNO)	32	R/W	Undefined	22.3.20/ 537
4000_93E8	TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD31_NBYTES_MLOFFNO)	32	R/W	Undefined	22.3.21/ 537
4000_93E8	TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD31_NBYTES_MLOFFYES)	32	R/W	Undefined	22.3.22/ 538
4000_93EC	TCD Last Source Address Adjustment (DMA_TCD31_SLAST)	32	R/W	Undefined	22.3.23/ 539
4000_93F0	TCD Destination Address (DMA_TCD31_DADDR)	32	R/W	Undefined	22.3.24/ 540

Table continues on the next page...

DMA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_93F4	TCD Signed Destination Address Offset (DMA_TCD31_DOFF)	16	R/W	Undefined	22.3.25/ 540
4000_93F6	TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD31_CITER_ELINKYES)	16	R/W	Undefined	22.3.26/ 541
4000_93F6	DMA_TCD31_CITER_ELINKNO	16	R/W	Undefined	22.3.27/ 542
4000_93F8	TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD31_DLASTSGA)	32	R/W	Undefined	22.3.28/ 543
4000_93FC	TCD Control and Status (DMA_TCD31_CSR)	16	R/W	Undefined	22.3.29/ 543
4000_93FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD31_BITER_ELINKYES)	16	R/W	Undefined	22.3.30/ 546
4000_93FE	TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD31_BITER_ELINKNO)	16	R/W	Undefined	22.3.31/ 547

22.3.1 Control Register (DMA_CR)

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 without regard to priority.

NOTE

> For proper 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 3 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 without regard to priority.

Address: DMA_CR is 4000_8000h base + 0h offset = 4000_8000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														CX	ECX
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				GRP1PRI		GRP0PRI		EMLM	CLM	HALT	HOE	ERGA	ERCA	EDBG	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_CR field descriptions

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value zero.
17 CX	Cancel Transfer 0 Normal operation 1 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 0 Normal operation 1 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 ES register and generating an optional error interrupt.
15–12 Reserved	This read-only field is reserved and always has the value zero.
11–10 GRP1PRI	Channel Group 1 Priority Group 1 priority level when fixed priority group arbitration is enabled.
9–8 GRP0PRI	Channel Group 0 Priority Group 0 priority level when fixed priority group arbitration is enabled.
7 EMLM	Enable Minor Loop Mapping

Table continues on the next page...

DMA_CR field descriptions (continued)

Field	Description
	0 Disabled. TCDn.word2 is defined as a 32-bit NBYTES field. 1 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 0 A minor loop channel link made to itself goes through channel arbitration before being activated again. 1 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 0 Normal operation 1 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 0 Normal operation 1 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 0 Fixed priority arbitration is used for selection among the groups. 1 Round robin arbitration is used for selection among the groups.
2 ERCA	Enable Round Robin Channel Arbitration 0 Fixed priority arbitration is used for channel selection within each group. 1 Round robin arbitration is used for channel selection within each group.
1 EDBG	Enable Debug 0 When in debug mode, the DMA continues to operate. 1 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 Reserved	This read-only field is reserved and always has the value zero.

22.3.2 Error Status Register (DMA_ES)

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

See the Error Reporting and Handling section for more details.

Address: DMA_ES is 4000_8000h base + 4h offset = 4000_8004h

Bit	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

Bit	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

DMA_ES field descriptions

Field	Description
31 VLD	Logical OR of all ERR status bits 0 No ERR bits are set 1 At least one ERR bit is set indicating a valid error exists that has not been cleared
30–17 Reserved	This read-only field is reserved and always has the value zero.
16 ECX	Transfer Cancelled 0 No cancelled transfers 1 The last recorded entry was a cancelled transfer by the error cancel transfer input
15 GPE	Group Priority Error 0 No group priority error 1 The last recorded error was a configuration error among the group priorities. All group priorities are not unique.
14 CPE	Channel Priority Error 0 No channel priority error 1 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	This read-only field is reserved and always has the value zero.
12–8 ERRCHN	Error Channel Number or Cancelled Channel Number The channel number of the last recorded error, excluding GPE and CPE errors, or last recorded error cancelled transfer.
7 SAE	Source Address Error

Table continues on the next page...

DMA_ES field descriptions (continued)

Field	Description
	0 No source address configuration error. 1 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 0 No source offset configuration error 1 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 0 No destination address configuration error 1 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 0 No destination offset configuration error 1 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 0 No NBYTES/CITER configuration error 1 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], or • TCDn_CITER[CITER] is equal to zero, or • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK]
2 SGE	Scatter/Gather Configuration Error 0 No scatter/gather configuration error 1 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 0 No source bus error 1 The last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0 No destination bus error 1 The last recorded error was a bus error on a destination write

22.3.3 Enable Request Register (DMA_ERQ)

The ERQ register provide a bit map for the 32 implemented 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. The {S,C}ERQ 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 the ERQ.

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.

Address: DMA_ERQ is 4000_8000h base + Ch offset = 4000_800Ch

Bit	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

Bit	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

DMA_ERQ field descriptions

Field	Description
31 ERQ31	Enable DMA Request 31 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
30 ERQ30	Enable DMA Request 30 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
29 ERQ29	Enable DMA Request 29 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
28 ERQ28	Enable DMA Request 28 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
27 ERQ27	Enable DMA Request 27 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
26 ERQ26	Enable DMA Request 26

Table continues on the next page...

DMA_ERQ field descriptions (continued)

Field	Description
	0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
25 ERQ25	Enable DMA Request 25 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
24 ERQ24	Enable DMA Request 24 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
23 ERQ23	Enable DMA Request 23 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
22 ERQ22	Enable DMA Request 22 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
21 ERQ21	Enable DMA Request 21 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
20 ERQ20	Enable DMA Request 20 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
19 ERQ19	Enable DMA Request 19 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
18 ERQ18	Enable DMA Request 18 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
17 ERQ17	Enable DMA Request 17 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
16 ERQ16	Enable DMA Request 16 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
15 ERQ15	Enable DMA Request 15 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

Table continues on the next page...

DMA_ERQ field descriptions (continued)

Field	Description
14 ERQ14	Enable DMA Request 14 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
13 ERQ13	Enable DMA Request 13 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
12 ERQ12	Enable DMA Request 12 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
11 ERQ11	Enable DMA Request 11 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
10 ERQ10	Enable DMA Request 10 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
9 ERQ9	Enable DMA Request 9 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
8 ERQ8	Enable DMA Request 8 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
7 ERQ7	Enable DMA Request 7 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
6 ERQ6	Enable DMA Request 6 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
5 ERQ5	Enable DMA Request 5 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
4 ERQ4	Enable DMA Request 4 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
3 ERQ3	Enable DMA Request 3

Table continues on the next page...

DMA_ERQ field descriptions (continued)

Field	Description
	0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
2 ERQ2	Enable DMA Request 2 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
1 ERQ1	Enable DMA Request 1 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled
0 ERQ0	Enable DMA Request 0 0 The DMA request signal for the corresponding channel is disabled 1 The DMA request signal for the corresponding channel is enabled

22.3.4 Enable Error Interrupt Register (DMA_EEI)

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. The {S,C}EEI are provided so 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.

Address: DMA_EEI is 4000_8000h base + 14h offset = 4000_8014h

Bit	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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EEI15	EEI14	EEI13	EEI12	EEI11	EEI10	EEI9	EEI8	EEI7	EEI6	EEI5	EEI4	EEI3	EEI2	EEI1	EEI0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DMA_EEI field descriptions

Field	Description
31 EEI31	Enable Error Interrupt 31 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
30 EEI30	Enable Error Interrupt 30

Table continues on the next page...

DMA_EEI field descriptions (continued)

Field	Description
	0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
29 EEI29	Enable Error Interrupt 29 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
28 EEI28	Enable Error Interrupt 28 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
27 EEI27	Enable Error Interrupt 27 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
26 EEI26	Enable Error Interrupt 26 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
25 EEI25	Enable Error Interrupt 25 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
24 EEI24	Enable Error Interrupt 24 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
23 EEI23	Enable Error Interrupt 23 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
22 EEI22	Enable Error Interrupt 22 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
21 EEI21	Enable Error Interrupt 21 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
20 EEI20	Enable Error Interrupt 20 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
19 EEI19	Enable Error Interrupt 19 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

Table continues on the next page...

DMA_EEI field descriptions (continued)

Field	Description
18 EEI18	Enable Error Interrupt 18 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
17 EEI17	Enable Error Interrupt 17 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
16 EEI16	Enable Error Interrupt 16 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
15 EEI15	Enable Error Interrupt 15 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
14 EEI14	Enable Error Interrupt 14 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
13 EEI13	Enable Error Interrupt 13 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
12 EEI12	Enable Error Interrupt 12 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
11 EEI11	Enable Error Interrupt 11 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
10 EEI10	Enable Error Interrupt 10 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
9 EEI9	Enable Error Interrupt 9 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
8 EEI8	Enable Error Interrupt 8 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
7 EEI7	Enable Error Interrupt 7

Table continues on the next page...

DMA_EEI field descriptions (continued)

Field	Description
	0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
6 EEI6	Enable Error Interrupt 6 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
5 EEI5	Enable Error Interrupt 5 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
4 EEI4	Enable Error Interrupt 4 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
3 EEI3	Enable Error Interrupt 3 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
2 EEI2	Enable Error Interrupt 2 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
1 EEI1	Enable Error Interrupt 1 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request
0 EEI0	Enable Error Interrupt 0 0 The error signal for corresponding channel does not generate an error interrupt 1 The assertion of the error signal for corresponding channel generates an error interrupt request

22.3.5 Clear Enable Error Interrupt Register (DMA_CEEI)

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 write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: DMA_CEEI is 4000_8000h base + 18h offset = 4000_8018h

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	CAEE	0	CEEI				
Reset	0	0	0	0	0	0	0	0

DMA_CEEI field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEE	Clear All Enable Error Interrupts 0 Clear only the EEI bit specified in the CEEI field 1 Clear all bits in EEI
5 Reserved	This field is reserved.
4–0 CEEI	Clear Enable Error Interrupt Clears the corresponding bit in EEI

22.3.6 Set Enable Error Interrupt Register (DMA_SEEI)

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 write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: DMA_SEEI is 4000_8000h base + 19h offset = 4000_8019h

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	SAEE	0	SEEI				
Reset	0	0	0	0	0	0	0	0

DMA_SEEI field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 SAEE	Sets All Enable Error Interrupts 0 Set only the EEI bit specified in the SEEI field. 1 Sets all bits in EEI
5 Reserved	This field is reserved.
4–0 SEEI	Set Enable Error Interrupt Sets the corresponding bit in EEI

22.3.7 Clear Enable Request Register (DMA_CERQ)

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 NOP 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.

Address: DMA_CERQ is 4000_8000h base + 1Ah offset = 4000_801Ah

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	CAER	0	CERQ				
Reset	0	0	0	0	0	0	0	0

DMA_CERQ field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 CAER	Clear All Enable Requests 0 Clear only the ERQ bit specified in the CERQ field 1 Clear all bits in ERQ
5 Reserved	This field is reserved.
4–0 CERQ	Clear Enable Request Clears the corresponding bit in ERQ{H,L}

22.3.8 Set Enable Request Register (DMA_SERQ)

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 write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: DMA_SERQ is 4000_8000h base + 1Bh offset = 4000_801Bh

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	SAER	0	SERQ				
Reset	0	0	0	0	0	0	0	0

DMA_SERQ field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 SAER	Set All Enable Requests 0 Set only the ERQ bit specified in the SERQ field 1 Set all bits in ERQ
5 Reserved	This field is reserved.
4–0 SERQ	Set Enable Request Sets the corresponding bit in ERQ

22.3.9 Clear DONE Status Bit Register (DMA_CDNE)

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 write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: DMA_CDNE is 4000_8000h base + 1Ch offset = 4000_801Ch

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	CADN	0	CDNE				
Reset	0	0	0	0	0	0	0	0

DMA_CDNE field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 CADN	Clears All DONE Bits 0 Clears only the TCDn_CSR[DONE] bit specified in the CDNE field 1 Clears all bits in TCDn_CSR[DONE]
5 Reserved	This field is reserved.
4–0 CDNE	Clear DONE Bit Clears the corresponding bit in TCDn_CSR[DONE]

22.3.10 Set START Bit Register (DMA_SSRT)

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 write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: DMA_SSRT is 4000_8000h base + 1Dh offset = 4000_801Dh

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	SAST	0	SSRT				
Reset	0	0	0	0	0	0	0	0

DMA_SSRT field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 SAST	Set All START Bits (activates all channels) 0 Set only the TCDn_CSR[START] bit specified in the SSRT field 1 Set all bits in TCDn_CSR[START]
5 Reserved	This field is reserved.
4–0 SSRT	Set START Bit Sets the corresponding bit in TCDn_CSR[START]

22.3.11 Clear Error Register (DMA_CERR)

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.

Address: DMA_CERR is 4000_8000h base + 1Eh offset = 4000_801Eh

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	CAEI	0	CERR				
Reset	0	0	0	0	0	0	0	0

DMA_CERR field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 CAEI	Clear All Error Indicators 0 Clear only the ERR bit specified in the CERR field 1 Clear all bits in ERR
5 Reserved	This field is reserved.
4–0 CERR	Clear Error Indicator Clears the corresponding bit in ERR

22.3.12 Clear Interrupt Request Register (DMA_CINT)

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 write multiple-byte registers as a 32-bit word. Reads of this register return all zeroes.

Address: DMA_CINT is 4000_8000h base + 1Fh offset = 4000_801Fh

Bit	7	6	5	4	3	2	1	0
Read	0	0		0				
Write	NOP	CAIR	0	CINT				
Reset	0	0	0	0	0	0	0	0

DMA_CINT field descriptions

Field	Description
7 NOP	0 Normal operation 1 No operation, ignore the other bits in this register
6 CAIR	Clear All Interrupt Requests 0 Clear only the INT bit specified in the CINT field 1 Clear all bits in INT
5 Reserved	This field is reserved.
4–0 CINT	Clear interrupt request Clears the corresponding bit in INT

22.3.13 Interrupt Request Register (DMA_INT)

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 (INTC). 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.

Address: DMA_INT is 4000_8000h base + 24h offset = 4000_8024h

Bit	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

Bit	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

DMA_INT field descriptions

Field	Description
31 INT31	Interrupt Request 31 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
30 INT30	Interrupt Request 30 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
29 INT29	Interrupt Request 29 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
28 INT28	Interrupt Request 28 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
27 INT27	Interrupt Request 27 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

Table continues on the next page...

DMA_INT field descriptions (continued)

Field	Description
26 INT26	Interrupt Request 26 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
25 INT25	Interrupt Request 25 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
24 INT24	Interrupt Request 24 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
23 INT23	Interrupt Request 23 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
22 INT22	Interrupt Request 22 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
21 INT21	Interrupt Request 21 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
20 INT20	Interrupt Request 20 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
19 INT19	Interrupt Request 19 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
18 INT18	Interrupt Request 18 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
17 INT17	Interrupt Request 17 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
16 INT16	Interrupt Request 16 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
15 INT15	Interrupt Request 15

Table continues on the next page...

DMA_INT field descriptions (continued)

Field	Description
	0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
14 INT14	Interrupt Request 14 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
13 INT13	Interrupt Request 13 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
12 INT12	Interrupt Request 12 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
11 INT11	Interrupt Request 11 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
10 INT10	Interrupt Request 10 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
9 INT9	Interrupt Request 9 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
8 INT8	Interrupt Request 8 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
7 INT7	Interrupt Request 7 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
6 INT6	Interrupt Request 6 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
5 INT5	Interrupt Request 5 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
4 INT4	Interrupt Request 4 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

Table continues on the next page...

DMA_INT field descriptions (continued)

Field	Description
3 INT3	Interrupt Request 3 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
2 INT2	Interrupt Request 2 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
1 INT1	Interrupt Request 1 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active
0 INT0	Interrupt Request 0 0 The interrupt request for corresponding channel is cleared 1 The interrupt request for corresponding channel is active

22.3.14 Error Register (DMA_ERR)

The ERR 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, then logically summed across groups of 16 and 32 channels to form several group error interrupt requests which is 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. 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.

Address: DMA_ERR is 4000_8000h base + 2Ch offset = 4000_802Ch

Bit	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

Bit	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

DMA_ERR field descriptions

Field	Description
31 ERR31	Error In Channel 31 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
30 ERR30	Error In Channel 30 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
29 ERR29	Error In Channel 29 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
28 ERR28	Error In Channel 28 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
27 ERR27	Error In Channel 27 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
26 ERR26	Error In Channel 26 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
25 ERR25	Error In Channel 25 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred

Table continues on the next page...

DMA_ERR field descriptions (continued)

Field	Description
24 ERR24	Error In Channel 24 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
23 ERR23	Error In Channel 23 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
22 ERR22	Error In Channel 22 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
21 ERR21	Error In Channel 21 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
20 ERR20	Error In Channel 20 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
19 ERR19	Error In Channel 19 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
18 ERR18	Error In Channel 18 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
17 ERR17	Error In Channel 17 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
16 ERR16	Error In Channel 16 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
15 ERR15	Error In Channel 15 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
14 ERR14	Error In Channel 14 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
13 ERR13	Error In Channel 13

Table continues on the next page...

DMA_ERR field descriptions (continued)

Field	Description
	0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
12 ERR12	Error In Channel 12 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
11 ERR11	Error In Channel 11 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
10 ERR10	Error In Channel 10 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
9 ERR9	Error In Channel 9 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
8 ERR8	Error In Channel 8 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
7 ERR7	Error In Channel 7 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
6 ERR6	Error In Channel 6 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
5 ERR5	Error In Channel 5 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
4 ERR4	Error In Channel 4 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
3 ERR3	Error In Channel 3 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
2 ERR2	Error In Channel 2 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred

Table continues on the next page...

DMA_ERR field descriptions (continued)

Field	Description
1 ERR1	Error In Channel 1 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred
0 ERR0	Error In Channel 0 0 An error in the corresponding channel has not occurred 1 An error in the corresponding channel has occurred

22.3.15 Hardware Request Status Register (DMA_HRS)

The HRS provide 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.

Address: DMA_HRS is 4000_8000h base + 34h offset = 4000_8034h

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	HRS31	HRS30	HRS29	HRS28	HRS27	HRS26	HRS25	HRS24	HRS23	HRS22	HRS21	HRS20	HRS19	HRS18	HRS17	HRS16	HRS15	HRS14	HRS13	HRS12	HRS11	HRS10	HRS9	HRS8	HRS7	HRS6	HRS5	HRS4	HRS3	HRS2	HRS1	HRS0
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

DMA_HRS field descriptions

Field	Description
31 HRS31	Hardware Request Status Channel 31 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
30 HRS30	Hardware Request Status Channel 30 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
29 HRS29	Hardware Request Status Channel 29

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
28 HRS28	Hardware Request Status Channel 28 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
27 HRS27	Hardware Request Status Channel 27 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
26 HRS26	Hardware Request Status Channel 26 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
25 HRS25	Hardware Request Status Channel 25 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
24 HRS24	Hardware Request Status Channel 24 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
23 HRS23	Hardware Request Status Channel 23 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
22 HRS22	Hardware Request Status Channel 22 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
21 HRS21	Hardware Request Status Channel 21 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
20 HRS20	Hardware Request Status Channel 20 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
19 HRS19	Hardware Request Status Channel 19 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
18 HRS18	Hardware Request Status Channel 18 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
17 HRS17	Hardware Request Status Channel 17 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
16 HRS16	Hardware Request Status Channel 16 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
15 HRS15	Hardware Request Status Channel 15 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
14 HRS14	Hardware Request Status Channel 14 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
13 HRS13	Hardware Request Status Channel 13 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
12 HRS12	Hardware Request Status Channel 12 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
11 HRS11	Hardware Request Status Channel 11 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
10 HRS10	Hardware Request Status Channel 10 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
9 HRS9	Hardware Request Status Channel 9 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
8 HRS8	Hardware Request Status Channel 8 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
7 HRS7	Hardware Request Status Channel 7 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
6 HRS6	Hardware Request Status Channel 6

Table continues on the next page...

DMA_HRS field descriptions (continued)

Field	Description
	0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
5 HRS5	Hardware Request Status Channel 5 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
4 HRS4	Hardware Request Status Channel 4 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
3 HRS3	Hardware Request Status Channel 3 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
2 HRS2	Hardware Request Status Channel 2 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
1 HRS1	Hardware Request Status Channel 1 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present
0 HRS0	Hardware Request Status Channel 0 0 A hardware service request for the corresponding channel is not present 1 A hardware service request for the corresponding channel is present

22.3.16 Channel n Priority Register (DMA_DCHPRI*n*)

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.

Addresses: 4000_8000h base + 100h offset + (1d × *n*), where *n* = 0d to 31d

Bit	7	6	5	4	3	2	1	0
Read	ECP	DPA	GRPPRI		CHPRI			
Write			▨					
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

DMA_DCHPRI*n* field descriptions

Field	Description
7 ECP	Enable Channel Preemption. This bit resets to zero. 0 Channel n cannot be suspended by a higher priority channel's service request 1 Channel n can be temporarily suspended by the service request of a higher priority channel
6 DPA	Disable preempt ability. This bit resets to zero. 0 Channel n can suspend a lower priority channel 1 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. These two bits are read only; writes are ignored. **NOTE:** Reset value for the group and channel priority fields, GRPPRI and CHPRI, is equal to the corresponding channel number for each priority register, i.e., DCHPRI31[GRPPRI] = 0b01 and DCHPRI31[CHPRI] equals 0b1111.
3–0 CHPRI	Channel n Arbitration Priority Channel priority when fixed-priority arbitration is enabled

Table continues on the next page...

DMA_DCHPRI*n* field descriptions (continued)

Field	Description
	NOTE: Reset value for the group and channel priority fields, GRPPRI and CHPRI, is equal to the corresponding channel number for each priority register, i.e., DCHPRI31[GRPPRI] = 0b01 and DCHPRI31[CHPRI] equals 0b1111.

22.3.17 TCD Source Address (DMA_TCD_SADDR)

Addresses: 4000_8000h base + 1000h offset + (32d × *n*), where *n* = 0d to 31d

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	SADDR																															
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.

DMA_TCD*n*_SADDR field descriptions

Field	Description
31–0 SADDR	Source Address Memory address pointing to the source data.

22.3.18 TCD Signed Source Address Offset (DMA_TCD_SOFF)

Addresses: 4000_8000h base + 1004h offset + (32d × *n*), where *n* = 0d to 31d

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

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_SOFF field descriptions

Field	Description
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.

22.3.19 TCD Transfer Attributes (DMA_TCD_ATTR)

Addresses: 4000_8000h base + 1006h offset + (32d × *n*), where *n* = 0d to 31d

<table>
<tr><td>Bit</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Read
Write</td><td colspan="5">SMOD</td><td colspan="3">SSIZE</td><td colspan="5">DMOD</td><td colspan="3">DSIZE</td></tr>
<tr><td>Reset</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td><td>x*</td></tr>
</table>

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_ATTR field descriptions

Field	Description
15–11 SMOD	Source Address Modulo 0 Source address modulo feature is disabled ≠0 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 Using a reserved encoding causes a configuration error. 000 8-bit 001 16-bit 010 32-bit 011 Reserved 100 16-byte burst 101 Reserved 110 Reserved 111 Reserved
7–3 DMOD	Destination Address Modulo See the SMOD definition
2–0 DSIZE	Destination Data Transfer Size See the SSIZE definition

22.3.20 TCD Minor Byte Count (Minor Loop Disabled) (DMA_TCD_NBYTES_MLNO)

TCD word 2's register definition depends on the status of minor loop mapping. If minor loop mapping is disabled (CR[EMLM] = 0), TCD word 2 is defined as follows. If minor loop mapping is enabled, see the TCD_NBYTES_MLOFFNO and TCD_NBYTES_MLOFFYES register descriptions for TCD word 2's register definition.

Addresses: 4000_8000h base + 1008h offset + (32d × *n*), where *n* = 0d to 31d

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	NBYTES																															
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.

DMA_TCD*n*_NBYTES_MLNO field descriptions

Field	Description
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. (Although, it may 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. **NOTE:** An NBYTES value of 0x0000_0000 is interpreted as a 4 GB transfer.

22.3.21 TCD Signed Minor Loop Offset (Minor Loop Enabled and Offset Disabled) (DMA_TCD_NBYTES_MLOFFNO)

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.

Addresses: 4000_8000h base + 1008h offset + (32d × *n*), where *n* = 0d to 31d

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	SMLOE	DMLOE	NBYTES																													
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.

DMA_TCD*n*_NBYTES_MLOFFNO field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 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. 0 The minor loop offset is not applied to the DADDR 1 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. (Although, it may 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.

22.3.22 TCD Signed Minor Loop Offset (Minor Loop and Offset Enabled) (DMA_TCD_NBYTES_MLOFFYES)

TCD word 2 is defined as follows if:

- Minor loop mapping is enabled (CR[EMLM] = 1) and
- Minor loop offset 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.

Addresses: 4000_8000h base + 1008h offset + (32d × *n*), where *n* = 0d to 31d

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	SMLOE	DMLOE	MLOFF																				NBYTES									
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.

DMA_TCD*n*_NBYTES_MLOFFYES field descriptions

Field	Description
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0 The minor loop offset is not applied to the SADDR 1 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. 0 The minor loop offset is not applied to the DADDR 1 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. (Although, it may 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.

22.3.23 TCD Last Source Address Adjustment (DMA_TCD_SLAST)

Addresses: 4000_8000h base + 100Ch offset + (32d × *n*), where *n* = 0d to 31d

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	SLAST																															
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.

DMA_TCD*n*_SLAST field descriptions

Field	Description
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.

22.3.24 TCD Destination Address (DMA_TCD_DADDR)

Addresses: 4000_8000h base + 1010h offset + (32d × *n*), where *n* = 0d to 31d

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	DADDR																															
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.

DMA_TCD*n*_DADDR field descriptions

Field	Description
31–0 DADDR	Destination Address Memory address pointing to the destination data.

22.3.25 TCD Signed Destination Address Offset (DMA_TCD_DOFF)

Addresses: 4000_8000h base + 1014h offset + (32d × *n*), where *n* = 0d to 31d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_DOFF field descriptions

Field	Description
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.

22.3.26 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD_CITER_ELINKYES)

If TCDn_CITER[ELINK] is set, the TCDn_CITER register is defined as follows.

Addresses: 4000_8000h base + 1016h offset + (32d × *n*), where *n* = 0d to 31d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	ELINK	0	LINKCH					CITER								
Write		▨														
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_CITER_ELINKYES field descriptions

Field	Description
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. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14 Reserved	This read-only field is reserved and always has the value zero.
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 to the channel defined by these six bits by setting that channel's TCDn_CSR[START] bit.
8–0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents 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 (e.g., 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.

Table continues on the next page...

DMA_TCD*n*_CITER_ELINKYES field descriptions (continued)

Field	Description
	NOTE: If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

22.3.27 TCD Current Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD_CITER_ELINKNO)

If TCDn_CITER[ELINK] is cleared, the TCDn_CITER register is defined as follows.

Addresses: 4000_8000h base + 1016h offset + (32d × *n*), where *n* = 0d to 31d

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

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_CITER_ELINKNO field descriptions

Field	Description
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. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14–0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents 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 (e.g., 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.

22.3.28 TCD Last Destination Address Adjustment/Scatter Gather Address (DMA_TCD_DLASTSGA)

Addresses: 4000_8000h base + 1018h offset + (32d × *n*), where *n* = 0d to 31d

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	DLASTSGA																															
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.

DMA_TCD*n*_DLASTSGA field descriptions

Field	Description
31–0 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. else • 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, else a configuration error is reported.

22.3.29 TCD Control and Status (DMA_TCD_CSR)

Addresses: 4000_8000h base + 101Ch offset + (32d × *n*), where *n* = 0d to 31d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	BWC		0	MAJORLINKCH					DONE	ACTIVE	MAJORELINK	ESG	DREQ	INTHALF	INTMAJOR	START
Write																
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_CSR field descriptions

Field	Description
15–14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. In general, 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. 00 No eDMA engine stalls 01 Reserved 10 eDMA engine stalls for 4 cycles after each r/w 11 eDMA engine stalls for 8 cycles after each r/w
13 Reserved	This read-only field is reserved and always has the value zero.
12–8 MAJORLINKCH	Link Channel Number If (MAJORELINK = 0) then • No channel-to-channel linking (or chaining) is performed after the major loop counter is exhausted. else • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by these six bits 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. This bit resets to zero.
6 ACTIVE	Channel Active This flag signals the channel is currently in execution. It is set when channel service begins, and the eDMA clears it as the minor loop completes or if 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. 0 The channel-to-channel linking is disabled 1 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.

Table continues on the next page...

DMA_TCD*n*_CSR field descriptions (continued)

Field	Description
	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. 0 The current channel's TCD is normal format. 1 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.
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. 0 The channel's ERQ bit is not affected 1 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 (aka ping-pong) schemes or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE:** If BITER is set, do not use INTHALF. Use INTMAJOR instead. 0 The half-point interrupt is disabled 1 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. 0 The end-of-major loop interrupt is disabled 1 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. This bit resets to zero. 0 The channel is not explicitly started 1 The channel is explicitly started via a software initiated service request

22.3.30 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Enabled) (DMA_TCD_BITER_ELINKYES)

If the TCDn_BITER[ELINK] bit is set, the TCDn_BITER register is defined as follows.

Addresses: 4000_8000h base + 101Eh offset + (32d × *n*), where *n* = 0d to 31d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	ELINK	0	LINKCH					BITER								
Write		▨														
Reset	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

DMA_TCD*n*_BITER_ELINKYES field descriptions

Field	Description
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 is reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 The channel-to-channel linking is enabled
14 Reserved	This read-only field is reserved and always has the value zero.
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 these six bits 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 is 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 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.

22.3.31 TCD Beginning Minor Loop Link, Major Loop Count (Channel Linking Disabled) (DMA_TCD_BITER_ELINKNO)

If the TCDn_BITER[ELINK] bit is cleared, the TCDn_BITER register is defined as follows.

Addresses: 4000_8000h base + 101Eh offset + (32d × *n*), where *n* = 0d to 31d

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

* Notes:
- x = Undefined at reset.

DMA_TCD*n*_BITER_ELINKNO field descriptions

Field	Description
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 is reloaded into the CITER field. 0 The channel-to-channel linking is disabled 1 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.

22.4 Functional description

22.4.1 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:

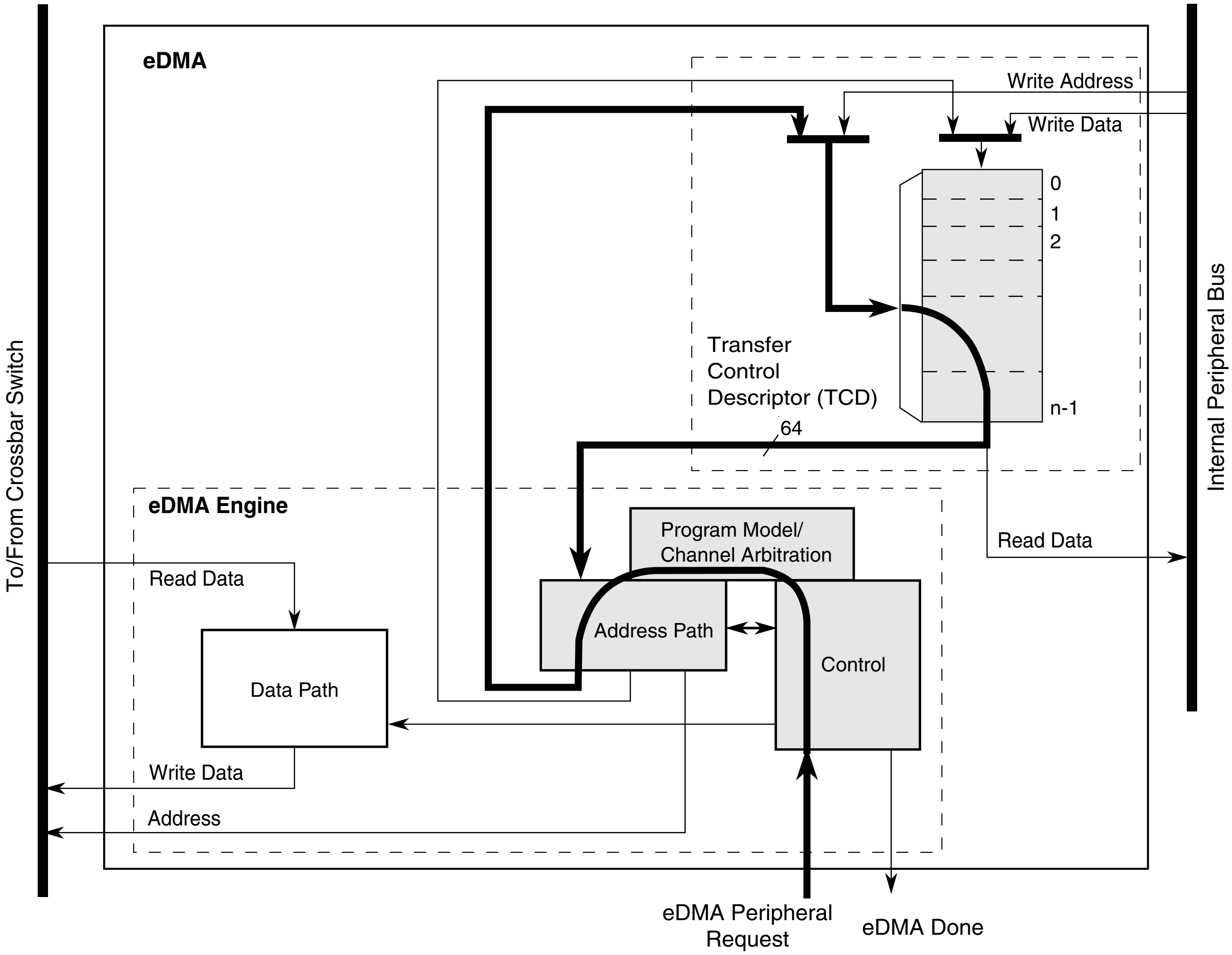


Figure 22-545. eDMA operation, part 1

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:

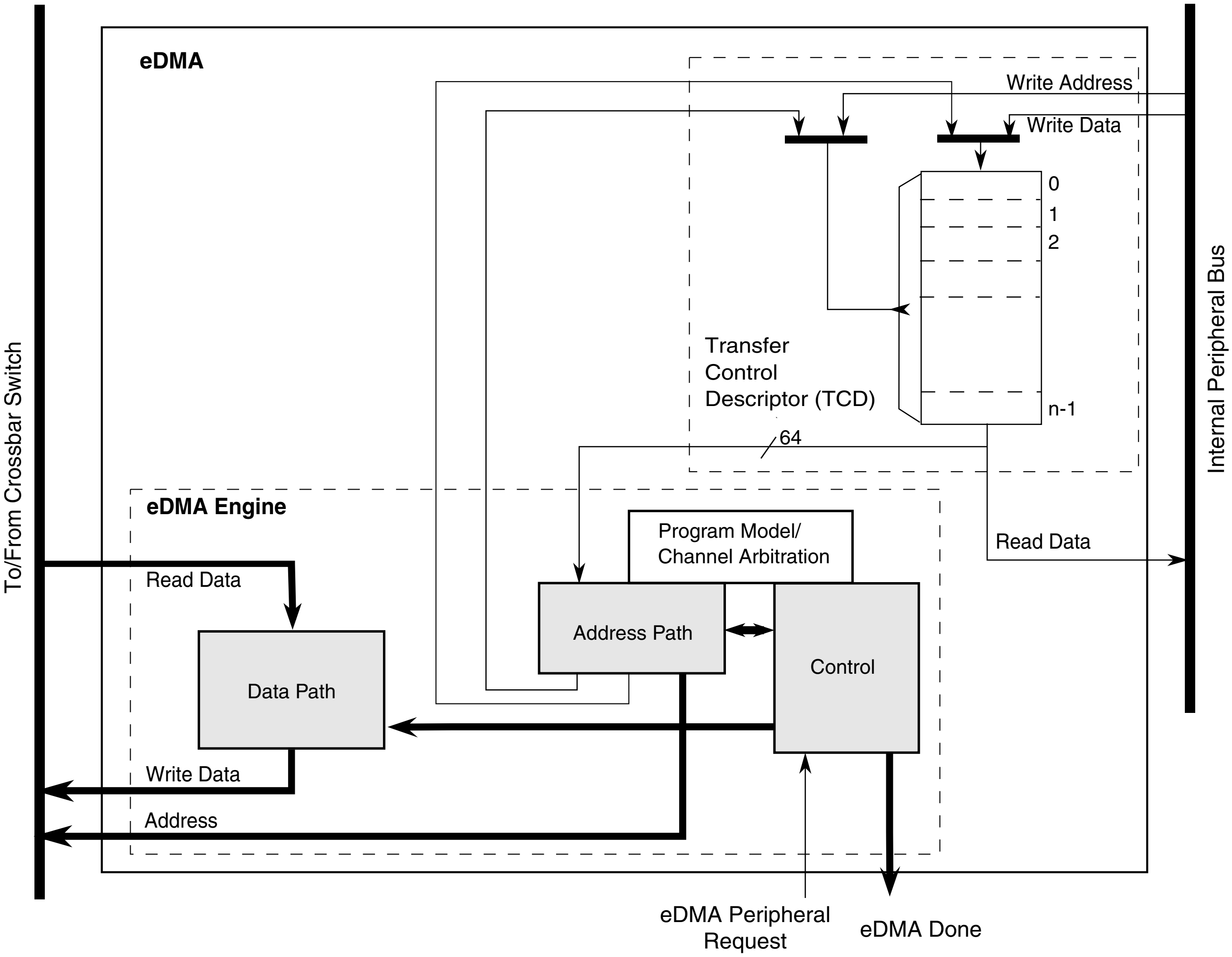


Figure 22-546. eDMA operation, part 2

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, e.g., 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.

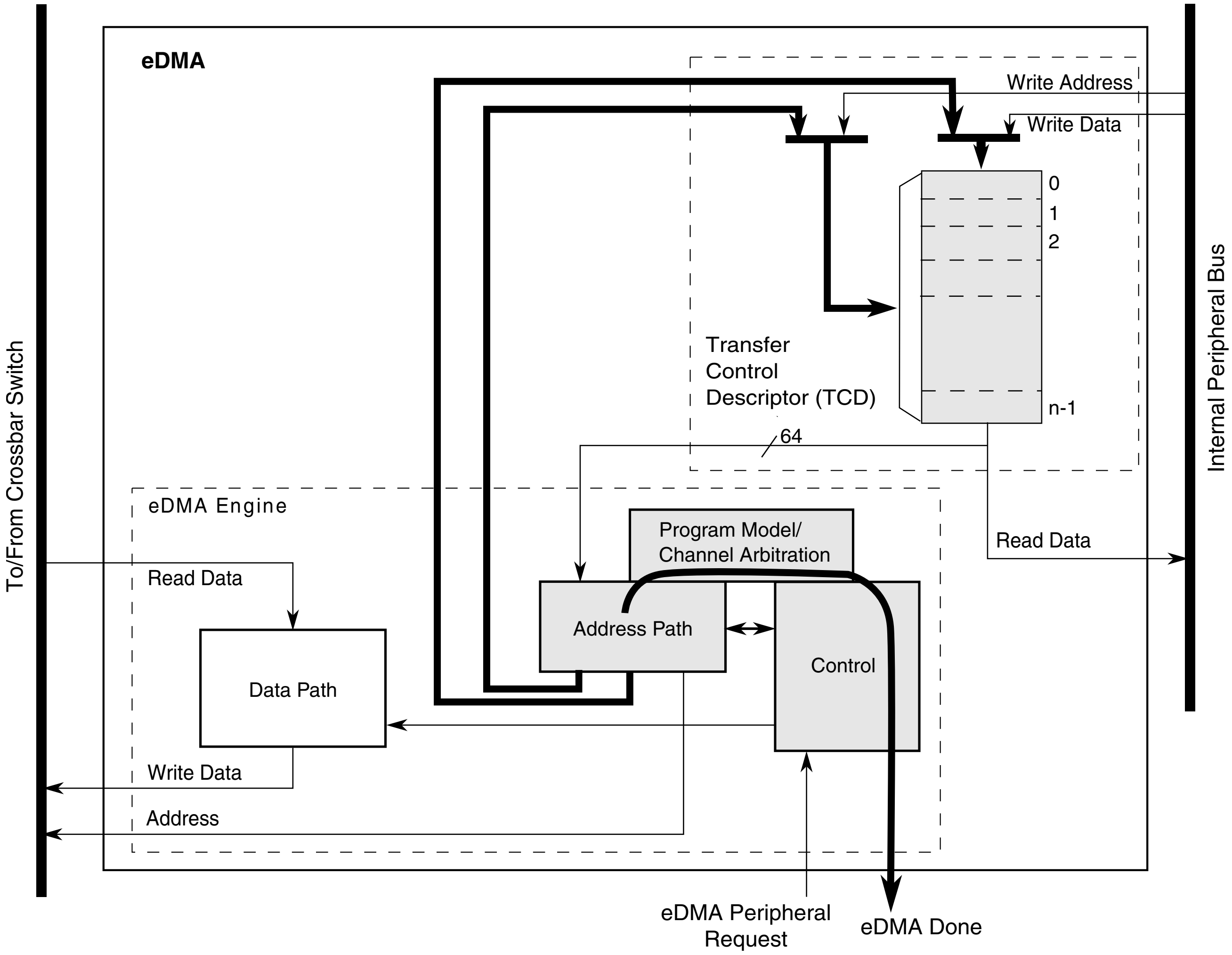


Figure 22-547. eDMA operation, part 3

22.4.2 Error reporting and handling

Channel errors are reported in the ES register 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.
- 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 read or write transaction, which is already pipelined after errant access, has completed. 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 ES register 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.

The occurrence of any error causes the eDMA engine to stop the active channel immediately, 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 ES register. 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.

22.4.3 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.

22.4.4 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.

22.4.4.1 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

This table presents a peak transfer rate comparison.

Table 22-548. eDMA peak transfer rates (Mbytes/sec)

System Speed, Width	Internal SRAM-to- Internal SRAM	32b internal peripheral bus-to- Internal SRAM	Internal SRAM-to- 32b internal peripheral bus
66.7 MHz, 32b	133.3	66.7	53.3
83.3 MHz, 32b	166.7	83.3	66.7
100.0 MHz, 32b	200.0	100.0	80.0
133.3 MHz, 32b	266.7	133.3	106.7
150.0 MHz, 32b	300.0	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.

22.4.4.2 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 22-549. 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.
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.

Table continues on the next page...

Table 22-549. Hardware service request process (continued)

Cycle		Description
With internal peripheral bus read and internal SRAM write	With SRAM read and internal peripheral bus write	
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 22-550. eDMA peak request rate (MReq/sec)

System frequency (MHz)	Request rate with zero wait states	Request rate with wait states
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 22-551. Peak request formula operands

Operand	Description
PEAKreq	Peak request rate
freq	System frequency
entry	Channel startup (4 cycles)
read_ws	Wait states seen during the system bus read data phase

Table continues on the next page...

Table 22-551. Peak request formula operands (continued)

Operand	Description
write_ws	Wait states seen during the system bus write data phase
exit	Channel shutdown (3 cycles)

22.4.4.3 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.

22.5 Initialization/application information

The following sections discuss initialization of the eDMA and programming considerations.

22.5.1 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 22-552. 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).

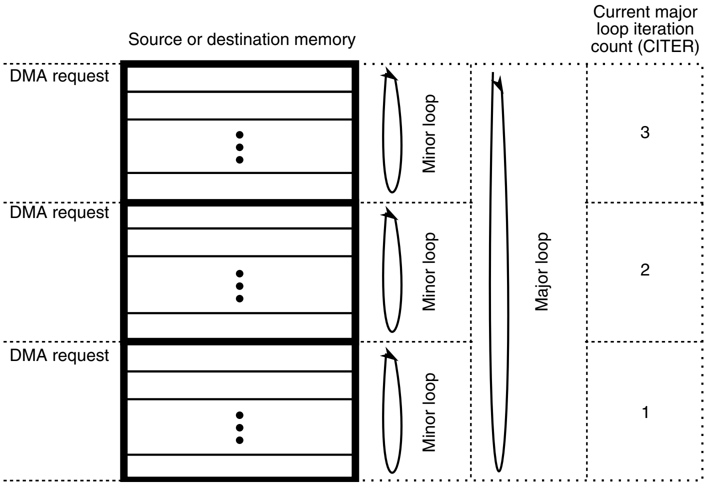


Figure 22-548. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings interrelate.

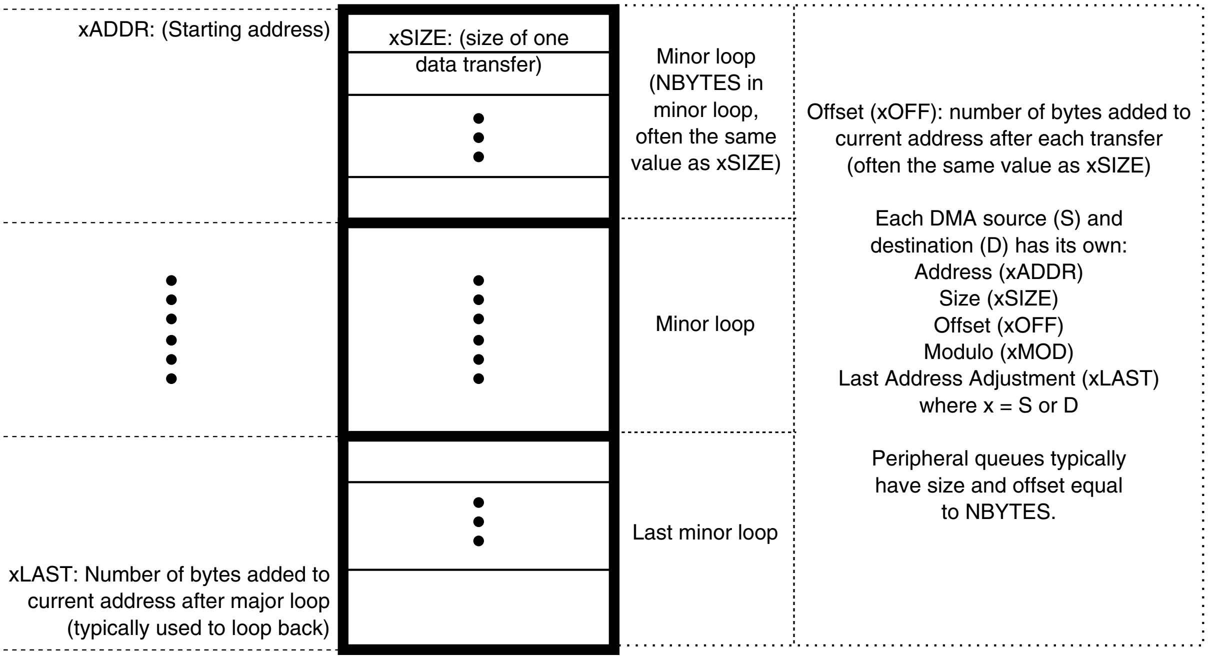

xADDR: (Starting address)
xSIZE: (size of one data transfer)
Minor loop (NBYTES in minor loop, often the same value as xSIZE)
Offset (xOFF): number of bytes added to current address after each transfer (often the same value as xSIZE)
Each DMA source (S) and destination (D) has its own:
Address (xADDR)
Size (xSIZE)
Offset (xOFF)
Modulo (xMOD)
Last Address Adjustment (xLAST)
where x = S or D
Minor loop
Peripheral queues typically have size and offset equal to NBYTES.
Last minor loop
xLAST: Number of bytes added to current address after major loop (typically used to loop back)

Figure 22-549. Memory array terms

22.5.2 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 ES register. 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.

22.5.3 Arbitration mode considerations

22.5.3.1 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.

22.5.3.2 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, channels 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.

22.5.4 Performing DMA transfers

22.5.4.1 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.

22.5.4.2 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.

22.5.4.3 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 22-553. Modulo example

Transfer Number	Address
1	0x12345670
2	0x12345674

Table continues on the next page...

Table 22-553. Modulo example (continued)

Transfer Number	Address
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

22.5.5 Monitoring transfer descriptor status

22.5.5.1 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.

22.5.5.2 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.

22.5.5.3 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.

22.5.6 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 22-554. 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

22.5.7 Dynamic programming

22.5.7.1 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.

22.5.7.2 Dynamic channel linking

Dynamic channel linking Dynamic channel linking is the process of setting the TCD.major.e_link bit during channel execution. 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 TCD.major.e_link bit at the same time the eDMA engine is retiring the channel. The TCD.major.e_link 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.

Step	Action
1	Write 1b to the TCD.major.e_link bit.
2	Read back the TCD.major.e_link bit.
3	Test the TCD.major.e_link request status: • If TCD.major.e_link = 1b, the dynamic link attempt was successful. • If TCD.major.e_link = 0b, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCD.major.e_link 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 TCD.done bit before writing the TCD.major.e_link bit. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

22.5.7.3 Dynamic scatter/gather

Dynamic scatter/gather is the process of setting the TCD.e_sg bit during channel execution. 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 scatter/gather operation by enabling the TCD.e_sg bit at the same time the eDMA engine is retiring the channel. The TCD.e_sg 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 major.linkch field and the e_sg 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 TCD.e_sg bits to zero on any writes to a channel's TCD.word7 if that channel's TCD.done bit is set indicating the major loop is complete.

NOTE

The user must clear the TCD.done bit before writing the TCD.major.e_link or TCD.e_sg bits. The TCD.done bit is cleared automatically by the eDMA engine after a channel begins execution.

22.5.7.3.1 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 TCD.major.e_link bit is zero, the TCD.major.linkch field is not used by the eDMA. In this case, the TCD.major.linkch bits may be used for other purposes. This method uses the TCD.major.linkch field as a TCD indentification (ID).

1. When the descriptors are built, write a unique TCD ID in the TCD.major.linkch field for each TCD associated with a channel using dynamic scatter/gather.
2. Write 1b to theTCD.d_req bit.

   Should a dynamic scatter/gather attempt fail, setting the d_req 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 offest value.

3. Write theTCD.dlast_sga field with the scatter/gather address.
4. Write 1b to the TCD.e_sg bit.
5. Read back the 16 bit TCD control/status field.
6. Test the TCD.e_sg request status and TCD.major.linkch value:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b and the major.linkch (ID) did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If e_sg = 0b and the major.linkch (ID) changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

22.5.7.3.2 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 indentification (ID).

1. Write 1b to theTCD.d_req bit.

   Should a dynamic scatter/gather attempt fail, setting the d_req 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 offest value.

2. Write theTCD.dlast_sga field with the scatter/gather address.
3. Write 1b to the TCD.e_sg bit.
4. Read back the TCD.e_sg bit.
5. Test the TCD.e_sg request status:

   If e_sg = 1b, the dynamic link attempt was successful.

   If e_sg = 0b, read the 32 bit TCD dlast_sga field.

   If e_sg = 0b and the dlast_sga did not change, the attempted dynamic link did not succeed (the channel was already retiring).

   If e_sg = 0b and the dlast_sga changed, the dynamic link attempt was successful (the new TCD's e_sg value cleared the e_sg bit).

Chapter 23 External Watchdog Monitor (EWM)

23.1 Introduction

NOTE

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

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 $\overline{\text{RESET}}$ 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.

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 EWM differs from the internal watchdog in that it does not reset the MCU's CPU and peripherals. The EWM if allowed to time-out, provides an independent $\overline{\text{EWM\_out}}$ pin that when asserted resets or places an external circuit into a safe mode. The CPU 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.

23.1.1 Features

Features of EWM module include:

- Independent LPO clock source
- Programmable time-out period specified in terms of number of EWM LPO clock cycles.

- Windowed refresh option
  - Provides robust check that program flow is faster than expected.
  - Programmable window.
  - Refresh outside window leads to assertion of $\overline{\mathrm{EWM\_out}}$.
- Robust refresh mechanism
  - Write values of 0xB4 and 0x2C to EWM Refresh Register within 15 (*EWM_service_time*) peripheral bus clock cycles.
- One output port, $\overline{\mathrm{EWM\_out}}$, 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 $\overline{\mathrm{EWM\_out}}$ signal.

23.1.2 Modes of Operation

This section describes the module's operating modes.

23.1.2.1 Stop Mode

When the EWM is in stop mode, the CPU services 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 service mechanism: At the exit from stop mode by an interrupt, refresh mechanism state machine starts from the previous state which means, if first service command is written correctly and EWM enters the stop mode immediately, the next command has to be written within the next 15 (*EWM_service_time*) peripheral bus clocks after exiting from stop mode. User must mask all interrupts prior to executing EWM service instructions.

23.1.2.2 Wait Mode

The EWM module treats the stop and wait modes as the same. EWM functionality remains the same in both of these modes.

23.1.2.3 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.

23.1.3 Block Diagram

This figure shows the EWM block diagram.

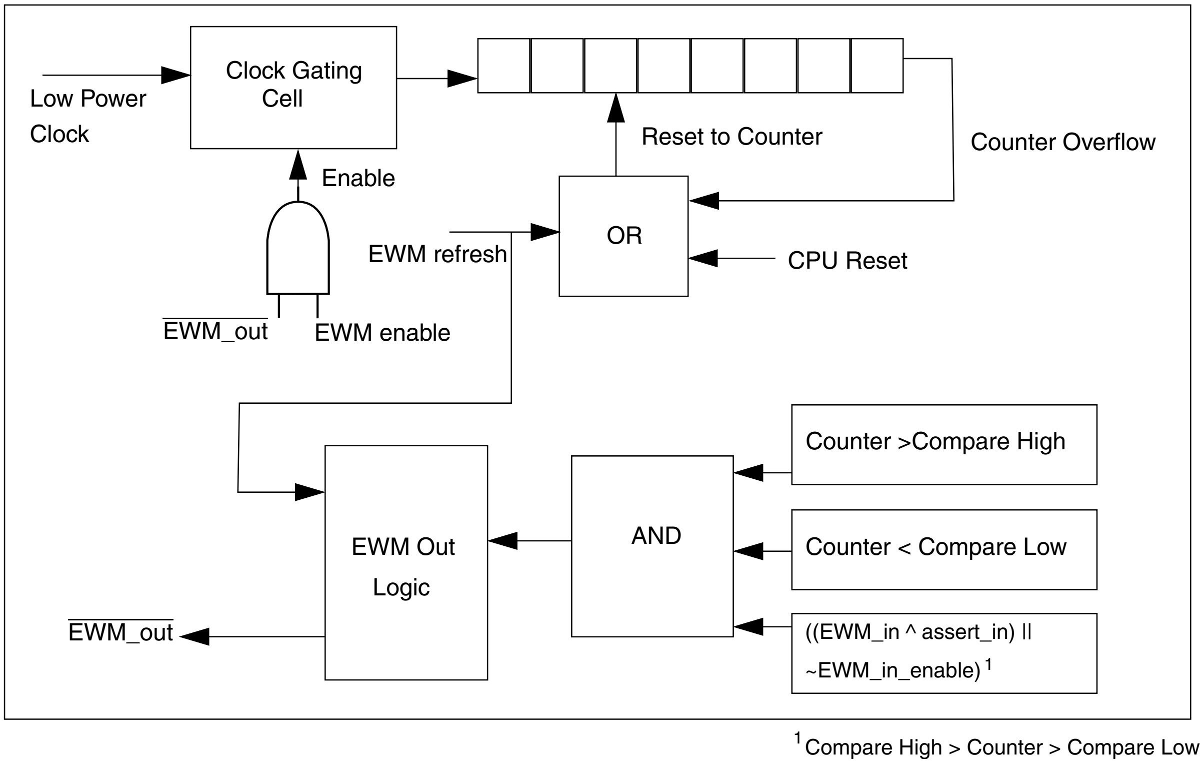


Figure 23-1. EWM Block Diagram

23.2 EWM Signal Descriptions

The EWM has two external signals, as shown in the following table.

Table 23-1. 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
$\overline{\text{EWM\_out}}$	EWM reset out signal	O

23.3 Memory Map/Register Definition

This section contains the module memory map and registers.

EWM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_1000	Control Register (EWM_CTRL)	8	R/W	00h	23.3.1/ 576
4006_1001	Service Register (EWM_SERV)	8	W (always reads zero)	00h	23.3.2/ 577
4006_1002	Compare Low Register (EWM_CMPL)	8	R/W	00h	23.3.3/ 578
4006_1003	Compare High Register (EWM_CMPH)	8	R/W	FFh	23.3.4/ 578

23.3.1 Control Register (EWM_CTRL)

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.

Address: EWM_CTRL is 4006_1000h base + 0h offset = 4006_1000h

Bit	7	6	5	4	3	2	1	0
Read	0				INTEN	INEN	ASSIN	EWMEN
Write								
Reset	0	0	0	0	0	0	0	0

EWM_CTRL field descriptions

Field	Description
7–4 Reserved	This read-only field is reserved and always has the value zero.
3 INTEN	Interrupt Enable. This bit when set and EWM_out 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 ASSIN bit inverts the assert state to a logic one.
0 EWMEN	EWM enable. This bit when set, enables the EWM module. This resets the EWM counter to zero and deasserts the EWM_out signal. Clearing EWMEN bit disables the EWM, and therefore it cannot be enabled until a reset occurs, due to the write-once nature of this bit.

23.3.2 Service Register (EWM_SERV)

The SERV register provides the interface from the CPU to the EWM module. It is write-only and reads of this register return zero.

Address: EWM_SERV is 4006_1000h base + 1h offset = 4006_1001h

Bit	7	6	5	4	3	2	1	0
Read	0							
Write	SERVICE							
Reset	0	0	0	0	0	0	0	0

EWM_SERV field descriptions

Field	Description
7–0 SERVICE	The EWM service 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 service is illegal if either of the following conditions is true.

EWM_SERV field descriptions (continued)

Field	Description
	• 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_service_time*.

23.3.3 Compare Low Register (EWM_CMPL)

The CMPL register is reset to zero after a CPU reset. This provides no minimum time for the CPU to service 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.

Address: EWM_CMPL is 4006_1000h base + 2h offset = 4006_1002h

Bit	7	6	5	4	3	2	1	0
Read Write	COMPAREL							
Reset	0	0	0	0	0	0	0	0

EWM_CMPL field descriptions

Field	Description
7–0 COMPAREL	To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) minimum service time is required.

23.3.4 Compare High Register (EWM_CMPH)

The CMPH register is reset to 0xFF after a CPU reset. This provides a maximum of 256 clocks time, for the CPU to service 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.

Address: EWM_CMPH is 4006_1000h base + 3h offset = 4006_1003h

Bit	7	6	5	4	3	2	1	0
Read Write	COMPAREH							
Reset	1	1	1	1	1	1	1	1

EWM_CMPH field descriptions

Field	Description
7–0 COMPAREH	To prevent runaway code from changing this field, software should write to this field after a CPU reset even if the (default) maximum service time is required.

23.4 Functional Description

The following sections describe functional details of the EWM module.

23.4.1 The $\overline{\text{EWM\_out}}$ Signal

The $\overline{\text{EWM\_out}}$ is a digital output signal used to gate an external circuit (application specific) that controls critical safety functions. For example, the $\overline{\text{EWM\_out}}$ could be connected to the high voltage transistors circuits that control an AC motor in a large appliance.

The $\overline{\text{EWM\_out}}$ signal remains deasserted when the EWM is being regularly serviced by the CPU within the programmable service window, indicating that the application code is executed as expected.

The $\overline{\text{EWM\_out}}$ signal is asserted in any of the following conditions:

- Servicing the EWM when the counter value is less than CMPL value.
- If the EWM counter value reaches the CMPH value, and no EWM service has occurred.
- Servicing the EWM when the counter value is more than CMPL and less than CMPH values and EWM_in signal is asserted.
- After any reset (by the virtue of the external pull-down mechanism on the $\overline{\text{EWM\_out}}$ pin)

On a normal reset, the $\overline{\text{EWM\_out}}$ is asserted. To deassert the $\overline{\text{EWM\_out}}$, set EWMEN bit in the CTRL register to enable the EWM.

If the EWM_out signal shares its pad with a digital I/O pin, on reset this actual pad defers to being an input signal. It takes the EWM_out output condition only after you enable the EWM by the EWMEN bit in the CTRL register.

When the EWM_out pin is asserted, it can only be deasserted by forcing a MCU reset.

Note

> EWM_out pad must be in pull down state when EWM functionality is used and when EWM is under Reset.

23.4.2 The EWM_in Signal

The EWM_in is a digital input signal that allows an external circuit to control the EWM_out signal. For example, in the application, an external circuit monitors a critical safety function, and if there is fault with this circuit's behavior, it can then actively initiate the EWM_out 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 servicing the EWM. This ensures that the EWM_out stays in the deasserted state; otherwise, the EWM_out pin is asserted.

Note

> You must update the CMPH and CMPL registers prior to enabling the EWM. After enabling the EWM, the counter resets to zero, therefore providing a reasonable time after a power-on reset for the external monitoring circuit to stabilize and ensure that the EWM_in pin is deasserted.

23.4.3 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 a CPU reset, or a EWM refresh cycle. The counter value is not accessible to the CPU.

23.4.4 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 service window, which is used by the CPU to service/refresh the EWM module.

- If the CPU services the EWM when the counter value lies between CMPL value and CMPH value, the counter is reset to zero. This is a legal service operation.
- If the CPU executes a EWM service/refresh action outside the legal service window, EWM_out is asserted.

It is illegal to program CMPL and CMPH with same value. In this case, as soon as counter reaches (CMPL + 1), EWM_out is asserted.

23.4.5 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. Therefore, three possible conditions can occur:

Table 23-7. EWM Refresh Mechanisms

Condition	Mechanism
A unique EWM service occurs when CMPL < Counter < CMPH.	The software behaves as expected and the counter of the EWM is reset to zero, and EWM_out pin remains in the deasserted state. **Note:** EWM_in pin is also assumed to be in the deasserted state.
A unique EWM service occurs when Counter < CMPL	The software services the EWM and therefore resets the counter to zero and asserts the EWM_out pin (irrespective of the EWM_in pin). The EWM_out pin is expected to gate critical safety circuits.
Counter value reaches CMPH prior to a unique EWM service	The counter value reaches the CMPH value and no service of the EWM resets the counter to zero and assert the EWM_out pin (irrespective of the EWM_in pin). The EWM_out pin is expected to gate critical safety circuits.

Any illegal service on EWM has no effect on EWM_out.

23.4.6 EWM Interrupt

When $\overline{\text{EWM\_out}}$ 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 $\overline{\text{EWM\_out}}$. The $\overline{\text{EWM\_out}}$ signal can be deasserted only by forcing a system reset.

Chapter 24
Watchdog Timer (WDOG)

24.1 Introduction

NOTE

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

The Watchdog Timer (WDOG) keeps a watch on the system functioning and resets it in case of its failure. Some reasons for such failures are: 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 watchdog brings the system into a safe state of operation. The watchdog monitors the operation of the system by expecting periodic communication from the software, generally known as servicing or refreshing the watchdog. If this periodic refreshing does not occur, the watchdog resets the system.

24.2 Features

The features of the Watchdog Timer (WDOG) include:

- Independent clock source input (independent from CPU/bus clock). Choice between two clock sources:
  - LPO Oscillator
  - External system clock
- Unlock sequence for allowing updates to write-once WDOG control/configuration bits.
- All WDOG control/configuration bits are writable once only within 256 bus clock cycles of being unlocked.

    - You need to always update these bits after unlocking within 256 bus clock cycles. Failure to update these bits, resets the system.
- Programmable time-out period specified in terms of number of WDOG clock cycles.
- Ability to test WDOG timer and reset with a flag indicating watchdog test.
    - Quick test—Small time-out value programmed for quick test.
    - Byte test—Individual bytes of timer tested one at a time.
    - Read-only access to the WDOG timer—Allows dynamic check that WDOG timer is operational.

NOTE

> Reading the watchdog timer counter while running the watchdog on the bus clock might not give the accurate counter value.

- Windowed refresh option
    - Provides robust check that program flow is faster than expected.
    - Programmable window.
    - Refresh outside window leads to reset.
- Robust refresh mechanism
    - Write values of 0xA602 and 0xB480 to WDOG Refresh Register within 20 bus clock cycles.
- Count of WDOG resets as they occur.
- Configurable interrupt on time-out to provide debug breadcrumbs. This is followed by a reset after 256 bus clock cycles.

24.3 Functional Overview

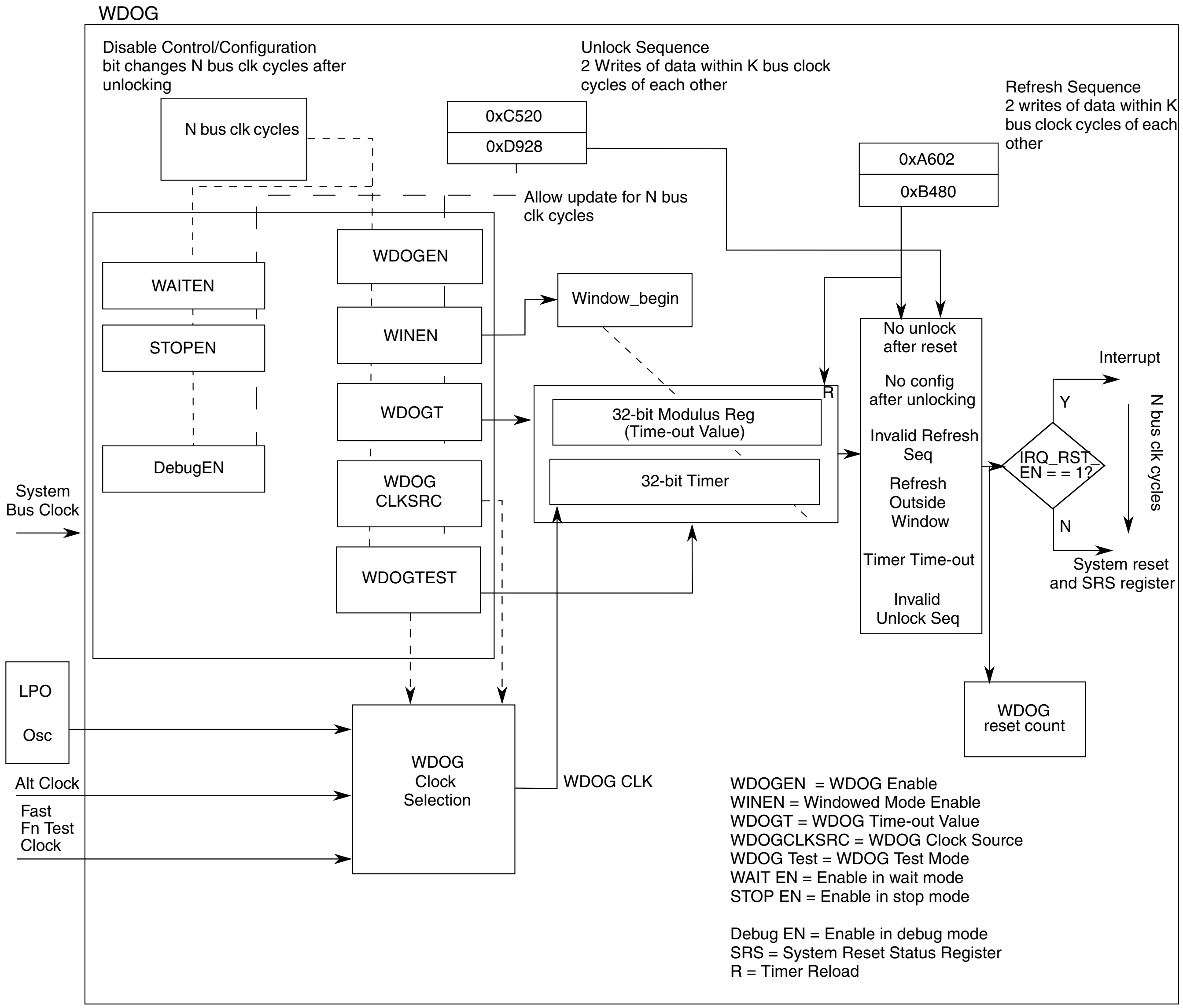

WDOG
Disable Control/Configuration bit changes N bus clk cycles after unlocking
N bus clk cycles
Unlock Sequence 2 Writes of data within K bus clock cycles of each other
0xC520
0xD928
Refresh Sequence 2 writes of data within K bus clock cycles of each other
0xA602
0xB480
Allow update for N bus clk cycles
WAITEN
STOPEN
DebugEN
WDOGEN
WINEN
WDOGT
WDOG CLKSRC
WDOGTEST
Window_begin
32-bit Modulus Reg (Time-out Value)
32-bit Timer
R
No unlock after reset
No config after unlocking
Invalid Refresh Seq
Refresh Outside Window
Timer Time-out
Invalid Unlock Seq
Interrupt
Y
IRQ_RST EN = = 1?
N
N bus clk cycles
System reset and SRS register
WDOG reset count
System Bus Clock
LPO
Osc
Alt Clock
Fast Fn Test Clock
WDOG Clock Selection
WDOG CLK
WDOGEN = WDOG Enable
WINEN = Windowed Mode Enable
WDOGT = WDOG Time-out Value
WDOGCLKSRC = WDOG Clock Source
WDOG Test = WDOG Test Mode
WAIT EN = Enable in wait mode
STOP EN = Enable in stop mode
Debug EN = Enable in debug mode
SRS = System Reset Status Register
R = Timer Reload

Figure 24-1. WDOG Operation

The preceding figure shows the operation of the watchdog. The values for N and K are:

- N = 256
- K = 20

The watchdog is a fail safe mechanism that brings the system into a known initial state in case of its failure due to CPU clock stopping or a run away condition in code execution. In its simplest form, the watchdog timer runs continuously off a clock source and expects to be serviced periodically, failing which it resets the system. This ensures that the software is executing correctly and has not run away in an unintended direction. Software can adjust the period of servicing or the time-out value for the watchdog timer to meet the needs of the application.

You can select a windowed mode of operation that expects the servicing to be done only in a particular window of the time-out period. An attempted servicing of the watchdog outside this window results in a reset. By operating in this mode, you can get an indication of whether the code is running faster than expected. The window length is also user programmable.

If a system fails to update/refresh the watchdog due to an unknown and persistent cause, it will be caught in an endless cycle of resets from the watchdog. To analyze the cause of such conditions, you can program the watchdog to first issue an interrupt, followed a little later by a reset. In the interrupt service routine, the software can analyze the system stack to aid debugging.

To enhance the independence of watchdog from the system, it runs off an independent LPO oscillator clock. You can also switch over to an alternate clock source if required, through a control register bit.

24.3.1 Unlocking and Updating the Watchdog

You can unlock the write-once-only control and configuration registers for updating them. As a pre-condition, the ALLOW_UPDATE bit in the watchdog control register must be set. The actual unlock is accomplished by writing 0xC520 followed by 0xD928 within 20 bus clock cycles to a specific unlock register (WDOG_UNLOCK). This opens up an update window equal in length to the watchdog configuration time (WCT) within which you can update the configuration and control register bits. You can not update registers on the bus clock cycle immediately following the write of the unlock sequence, but one cycle later. These register bits can be modified only once after unlocking.

If none of the configuration and control registers is updated within the update window, the watchdog issues a reset (or interrupt-then-reset) to the system. Trying to unlock the watchdog within the WCT time after an initial unlock, has no effect. During the update operation, the watchdog timer is not paused and keeps running in the background. After the update window closes, the watchdog timer restarts and the watchdog functions as per the new configuration.

The update feature is useful for applications that have an initial, non-safety critical part, where the watchdog is kept disabled or with a conveniently long time-out period. This means the application coder does not have to bother with frequently servicing the watchdog. After the critical part of the application begins, the watchdog can be reconfigured as per need.

The watchdog issues a reset (or interrupt-then-reset if enabled) to the system for any of these invalid unlock sequences:

- You write any value other than 0xC520 or 0xD928 to the unlock register.
- ALLOW_UPDATE is set and you allow a gap of more than 20 bus clock cycles between the writing of the unlock sequence values.

Also, an attempted refresh operation between the two writes of the unlock sequence and in the WCT time following a successful unlock, goes undetected. Also, see Watchdog Operation with 8-bit access for guidelines related to 8-bit accesses to the unlock register.

Note

A context switch during unlocking and refreshing may lead to a watchdog reset.

24.3.2 The Watchdog Configuration Time (WCT)

To prevent unintended modification of the watchdog's control and configuration register bits, you are allowed to update them only within a period of 256 bus clock cycles after unlocking. This window period is known as the watchdog configuration time (WCT). In addition, these register bits can be modified only once after unlocking them for editing (even after reset).

You must unlock the registers within WCT time after system reset, failing which the WDOG issues a reset to the system. To be more precise, you must write at least the first word of the unlocking sequence within the WCT time after reset. Once this is done, you get a further 20 bus clock cycles (the maximum allowed gap between the words of the unlock sequence) to complete the unlocking operation. Thereafter, to make sure that you do not forget to configure the watchdog, the watchdog issues a reset if none of the WDOG control and configuration registers is updated in the WCT time after unlock. After the close of this window or after the first write, these register bits are locked out from any further changes.

The watchdog timer keeps running as per its default configuration through unlocking and update operations that can extend up to a maximum total of 2xWCT time + 20 bus clock cycles. Therefore, it must be ensured that the time-out value for the watchdog is always greater than 2xWCT time + 20 bus clock cycles.

Updates in the write–once registers take effect only after the WCT window closes with the following exceptions for which changes take effect immediately:

- the stop, wait, and debug mode enable bits
- the IRQ_RST_EN bit

The operations of refreshing the watchdog goes undetected during the WCT.

24.3.3 Refreshing the Watchdog

A robust refreshing mechanism has been chosen for the watchdog. A valid refresh is a write of 0xA602 followed by 0xB480 within 20 bus clock cycles to watchdog refresh register. If these two values are written more than 20 bus cycles apart or if something other than these two values is written to the register, a watchdog reset (or interrupt-then-reset if enabled) is issued to the system. A valid refresh makes the watchdog timer restart on the next bus clock. Also, an attempted unlock operation, in between the two writes of the refresh sequence goes undetected. See Watchdog Operation with 8-bit access for guidelines related to 8-bit accesses to the refresh register.

24.3.4 Windowed Mode of Operation

In this mode of operation a restriction is placed on the point in time within the time-out period at which the watchdog can be refreshed. The refresh is considered valid only when the watchdog timer increments beyond a certain count as specified by the watchdog window register. This is known as refreshing the watchdog within a window of the total time-out period. If a refresh is attempted before the timer reaches the window value, the watchdog generates a reset (or interrupt-then-reset if enabled). Of course, if there is no refresh at all, the watchdog times out and generates a reset or interrupt-then-reset if enabled.

24.3.5 Watchdog Disabled Mode of Operation

When the watchdog is disabled through the WDOG_EN bit in the watchdog status and control register, the watchdog timer is reset to zero and is disabled from counting until you enable it or it is again enabled by the system reset. In this mode the watchdog timer cannot be refreshed (there is no requirement to do so while the timer is disabled). However, the watchdog still generates a reset (or interrupt-then-reset if enabled) on a non-time-out exception (see Generated Resets and Interrupts). You need to unlock the watchdog before enabling it. A system reset brings the watchdog out of the disabled mode.

24.3.6 Low Power Modes of Operation

- In Wait mode, if the WDOG is enabled (WAIT_EN = 1), it can run on bus clock or low power oscillator clock (CLK_SRC = x) to generate interrupt (IRQ_RST_EN=1) followed by a reset on time-out. After reset the WDOG reset counter increments by one.
- In Stop mode where the bus clock is gated, the WDOG can run only on low power oscillator clock (CLK_SRC=0) if it is enabled in stop (STOP_EN=1). In this case, the WDOG runs to time-out twice, and then generates a reset from its backup circuitry. Therefore, if you program the watchdog to time-out after 100 ms and then enter such a stop mode, the reset will occur after 200 ms. Also, in this case no interrupt will be generated irrespective of the value of IRQ_RST_EN bit. After WDOG reset, the WDOG reset counter will also not increment.
- In Power-down mode, the watchdog is powered off.

24.3.7 Debug Modes of Operation

You can program the watchdog to disable in debug modes (through DBG_EN bit in the watchdog control register). This results in the watchdog timer pausing for the duration of the mode. Register read/writes are still allowed, which means that operations like: refresh, unlock etc. are allowed. On exit from the mode, the timer resumes its operation from the point of pausing.

The entry of the system into the debug mode does not excuse it from compulsorily configuring the watchdog in the WCT time after unlock (unless the system bus clock is gated off, in which case the internal state machine pauses too). Failing to do so still results in a reset (or interrupt-then-reset, if enabled) to the system. Also, all the exception conditions that result in a reset to the system (see Generated Resets and Interrupts) are still valid in this mode. So, if an exception condition occurs and the system bus clock is on, a reset occurs (or interrupt-then-reset, if enabled).

The entry into Debug mode within WCT time after reset is treated differently. The WDOG timer is kept reset to zero and there is no need to unlock and configure it within WCT time. You must not try to refresh or unlock the WDOG in this state or unknown behavior may result. Upon exit from this mode, the WDOG timer restarts and the WDOG has to be unlocked and configured within WCT time.

24.4 Testing the Watchdog

For IEC 60730 and other safety standards, the expectation is that anything that monitors a safety function must be tested and this test is required to be fault tolerant. To test the watchdog, its main timer and its associated compare and reset logic must be tested. Towards this end, two tests are implemented for the watchdog that are described in Quick Test and Byte Test. While there is a control bit provided to put the watchdog into the test mode (functional), there is an overriding test-disable control bit which once set, disables the test mode permanently until reset.

For running a particular test, first select that test. Thereafter, set a certain test mode bit to put the watchdog in the functional test mode. Setting this bit automatically switches the watchdog timer to a fast clock source. The switching of the clock source is done to achieve a faster time-out and hence a faster test. In a successful test, the timer times out after reaching the programmed time-out value and generates a system reset.

Note

> After emerging from a reset due to a watchdog test, you must follow the mandatory steps of unlocking and configuring the watchdog. The refresh and unlock operations and interrupt are not automatically disabled in the test mode.

24.4.1 Quick Test

In this test the time-out value of watchdog timer is programmed to a very low value to achieve quick time-out. The only difference between the quick test and the normal mode of functioning of the watchdog is that the test mode bit is set for the quick test. This allows quick test of the watchdog reset mechanism.

24.4.2 Byte Test

The byte test implements more thorough a test of the watchdog timer. In this test, the timer is split up into its constituent byte-wide stages that are run independently and tested for time-out against the corresponding byte of the time-out value register. The following figure explains the splitting concept:

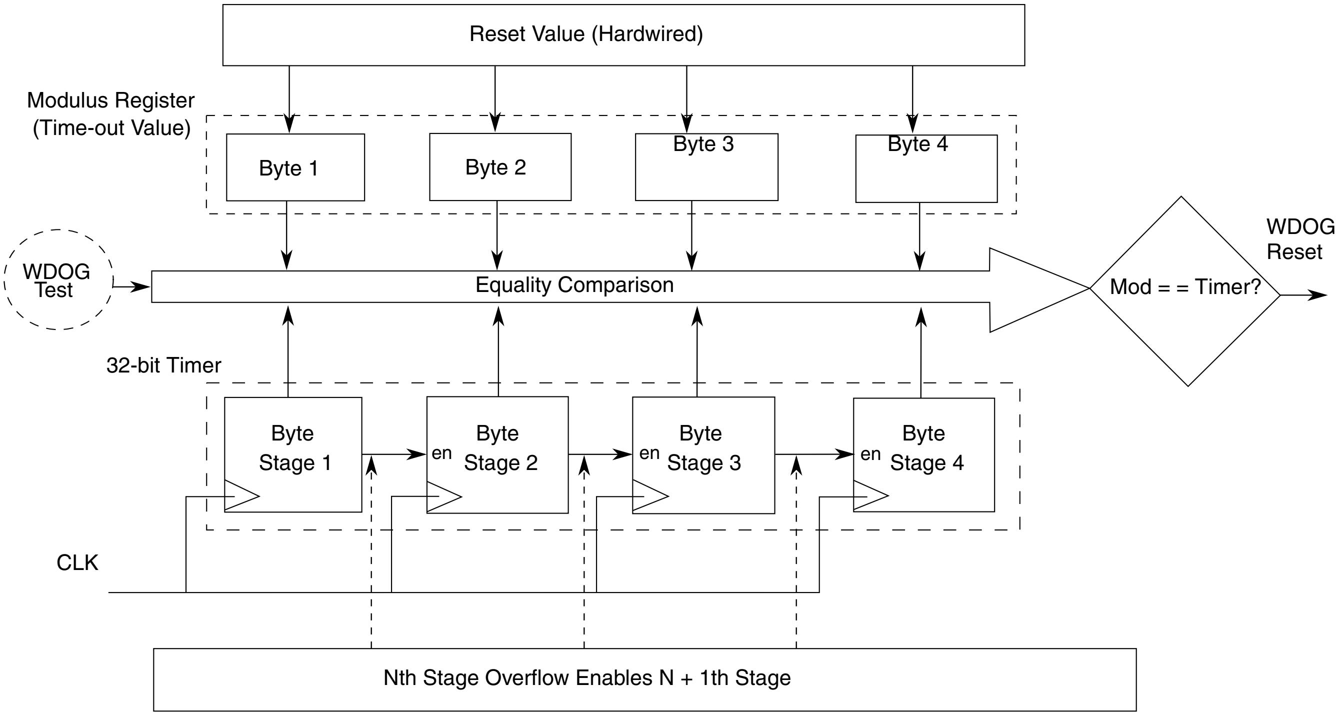


Figure 24-2. Watchdog Timer Byte Splitting

Each stage is an 8-bit synchronous counter followed by combinational logic that generates an overflow signal. The overflow signal acts as an enable to the N + 1th stage.

In the test mode, when an individual byte, N, is tested, byte N – 1 is loaded forcefully with 0xFF, and both these bytes are allowed to run off the clock source. By doing so the overflow signal from stage N – 1 is generated immediately, enabling counter stage N. The Nth stage runs and compares with the Nth byte of the time-out value register. In this way, the byte N is also tested along with the link between it and the preceding stage. No other stages, N – 2, N – 3... and N + 1, N + 2... are enabled for the test on byte N. These disabled stages (except the most significant stage of the counter) are loaded with a value of 0xFF.

These two testing schemes achieve the overall aim of testing the counter functioning and the compare and reset logic.

Note

Do not enable the watchdog interrupt during these tests. If required, you must ensure that the effective time-out value is greater than WCT time. See Generated Resets and Interrupts for more details.

24.5 Backup Reset Generator

The backup reset generator generates the final reset which goes out to the system. It has a backup mechanism which takes care that in case the bus clock stops and prevents the main state machine from generating a reset exception/interrupt, the watchdog timer's time-out is separately routed out as a reset to the system. Two successive timer time-outs without an intervening system reset result in the backup reset generator routing out the time-out signal as a reset to the system.

24.6 Generated Resets and Interrupts

The watchdog generates a reset on the following events (referred to as exceptions at some places in this document):

- A watchdog time-out.
- Failure to unlock the watchdog within WCT time after system reset deassertion.
- No update of the control and configuration registers within the WCT window after unlocking. At least one of the following registers must be written to within the WCT window to avoid reset:
  - WDOG_ST_CTRL_H, WDOG_ST_CTRL_L
  - WDOG_TO_VAL_H, WDOG_TO_VAL_L
  - WDOG_WIN_H, WDOG_WIN_L
  - WDOG_PRESCALER
- A value other than the unlock sequence or the refresh sequence is written to the unlock and/or refresh registers, respectively.

- A gap of more than 20 bus cycles exists between the writes of two values of the unlock sequence.
- A gap of more than 20 bus cycles exists between the writes of two values of the refresh sequence.

The watchdog can also generate an interrupt. If IRQ_RST_EN is set, then on the above mentioned events WDOG_ST_CTRL_L[INT_FLG] is set, generating an interrupt. A watchdog reset is also generated WCT time later to ensure the watchdog is fault tolerant. The interrupt can be cleared by writing 1 to INT_FLG.

The gap of WCT time between interrupt and reset means that the WDOG time-out value must be greater than WCT. Otherwise, if the interrupt was generated due to a time-out, a second consecutive time-out will occur in that WCT gap. This will trigger the backup reset generator to generate a reset to the system, prematurely ending the interrupt service routine execution. Also, the jobs like counting the number of watchdog resets would not be done.

24.7 Memory Map and Register Definition

This section consists of the memory map and register descriptions.

WDOG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_2000	Watchdog Status and Control Register High (WDOG_STCTRLH)	16	R/W	01D3h	24.7.1/ 594
4005_2002	Watchdog Status and Control Register Low (WDOG_STCTRLL)	16	R/W	0001h	24.7.2/ 596
4005_2004	Watchdog Time-out Value Register High (WDOG_TOVALH)	16	R/W	004Ch	24.7.3/ 596
4005_2006	Watchdog Time-out Value Register Low (WDOG_TOVALL)	16	R/W	4B4Ch	24.7.4/ 597
4005_2008	Watchdog Window Register High (WDOG_WINH)	16	R/W	0000h	24.7.5/ 597
4005_200A	Watchdog Window Register Low (WDOG_WINL)	16	R/W	0010h	24.7.6/ 598
4005_200C	Watchdog Refresh Register (WDOG_REFRESH)	16	R/W	B480h	24.7.7/ 598

Table continues on the next page...

WDOG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4005_200E	Watchdog Unlock Register (WDOG_UNLOCK)	16	R/W	D928h	24.7.8/ 599
4005_2010	Watchdog Timer Output Register High (WDOG_TMROUTH)	16	R/W	0000h	24.7.9/ 599
4005_2012	Watchdog Timer Output Register Low (WDOG_TMROUTL)	16	R/W	0000h	24.7.10/ 600
4005_2014	Watchdog Reset Count Register (WDOG_RSTCNT)	16	R/W	0000h	24.7.11/ 600
4005_2016	Watchdog Prescaler Register (WDOG_PRESC)	16	R/W	0400h	24.7.12/ 601

24.7.1 Watchdog Status and Control Register High (WDOG_STCTRLH)

Address: WDOG_STCTRLH is 4005_2000h base + 0h offset = 4005_2000h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	DISTESTWDOG	BYTESEL[1:0]		TESTSEL	TESTWDOG	0	Reserved	WAITEN	STOPEN	DBGEN	ALLOWUPDATE	WINEN	IRQRSTEN	CLKSRC	WDOGEN
Write																
Reset	0	0	0	0	0	0	0	1	1	1	0	1	0	0	1	1

WDOG_STCTRLH field descriptions

Field	Description
15 Reserved	This read-only field is reserved and always has the value zero.
14 DISTESTWDOG	Allows the WDOG’s functional test mode to be disabled permanently. Once set, it can only be cleared by a reset. It cannot be unlocked for editing once it is set. 0 WDOG functional test mode is not disabled. 1 WDOG functional test mode is disabled permanently until reset.
13–12 BYTESEL[1:0]	This 2-bit field select the byte to be tested when the watchdog is in the byte test mode. 00 Byte 0 selected 01 Byte 1 selected 10 Byte 2 selected 11 Byte 3 selected
11 TESTSEL	Selects the test to be run on the watchdog timer. Effective only if TESTWDOG is set.

Table continues on the next page...

WDOG_STCTRLH field descriptions (continued)

Field	Description
	0 Quick test. The timer runs in normal operation. You can load a small time-out value to do a quick test. 1 Byte test. Puts the timer in the byte test mode where individual bytes of the timer are enabled for operation and are compared for time-out against the corresponding byte of the programmed time-out value. Select the byte through BYTESEL[1:0] for testing.
10 TESTWDOG	Puts the watchdog in the functional test mode. In this mode the watchdog timer and the associated compare and reset generation logic is tested for correct operation. The clock for the timer is switched from the main watchdog clock to the fast clock input for watchdog functional test. The TESTSEL bit selects the test to be run.
9 Reserved	This read-only field is reserved and always has the value zero.
8 Reserved	This field is reserved.
7 WAITEN	Enables or disables WDOG in wait mode. 0 WDOG is disabled in CPU wait mode. 1 WDOG is enabled in CPU wait mode.
6 STOPEN	Enables or disables WDOG in stop mode. 0 WDOG is disabled in CPU stop mode. 1 WDOG is enabled in CPU stop mode.
5 DBGEN	Enables or disables WDOG in Debug mode. 0 WDOG is disabled in CPU Debug mode. 1 WDOG is enabled in CPU Debug mode.
4 ALLOWUPDATE	Enables updates to watchdog write once registers, after initial configuration window (WCT) closes, through unlock sequence. 0 No further updates allowed to WDOG write once registers. 1 WDOG write once registers can be unlocked for updating.
3 WINEN	Enable windowing mode. 0 Windowing mode is disabled. 1 Windowing mode is enabled.
2 IRQRSTEN	Used to enable the debug breadcrumbs feature. A change in this bit is updated immediately, as opposed to updating after WCT. 0 WDOG time-out generates reset only. 1 WDOG time-out initially generates an interrupt. After WCT time, it generates a reset.
1 CLKSRC	Selects clock source for the WDOG timer and other internal timing operations. 0 Dedicated clock source selected as WDOG clock (LPO Oscillator). 1 WDOG clock sourced from alternate clock source.
0 WDOGEN	Enables or disables the WDOG's operation. In the disabled state, the watchdog timer is kept in the reset state, but the other exception conditions can still trigger a reset/interrupt. A change in the value of this bit must be held for more than one WDOG_CLK cycle for the WDOG to be enabled or disabled. 0 WDOG is disabled. 1 WDOG is enabled.

24.7.2 Watchdog Status and Control Register Low (WDOG_STCTRLL)

Address: WDOG_STCTRLL is 4005_2000h base + 2h offset = 4005_2002h

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

WDOG_STCTRLL field descriptions

Field	Description
15 INTFLG	Interrupt flag. It is set when an exception occurs. IRQRSTEN = 1 is a precondition to set this flag. INTFLG = 1 results in an interrupt being issued followed by a reset, WCT time later. The interrupt can be cleared by writing 1 to this bit. It also gets cleared on a system reset.
14–0 Reserved	This field is reserved. **NOTE:** Do not modify this bitfield value.

24.7.3 Watchdog Time-out Value Register High (WDOG_TOVALH)

Address: WDOG_TOVALH is 4005_2000h base + 4h offset = 4005_2004h

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

WDOG_TOVALH field descriptions

Field	Description
15–0 TOVALHIGH	Defines the upper 16 bits of the 32-bit time-out value for the watchdog timer. It is defined in terms of cycles of the watchdog clock.

24.7.4 Watchdog Time-out Value Register Low (WDOG_TOVALL)

The time-out value of the watchdog must be set to a minimum of four watchdog clock cycles. This is to take into account the delay in new settings taking effect in the watchdog clock domain.

Address: WDOG_TOVALL is 4005_2000h base + 6h offset = 4005_2006h

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

WDOG_TOVALL field descriptions

Field	Description
15–0 TOVALLOW	Defines the lower 16 bits of the 32-bit time-out value for the watchdog timer. It is defined in terms of cycles of the watchdog clock.

24.7.5 Watchdog Window Register High (WDOG_WINH)

NOTE

You must set the Window Register value lower than the Time-out Value Register.

Address: WDOG_WINH is 4005_2000h base + 8h offset = 4005_2008h

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

WDOG_WINH field descriptions

Field	Description
15–0 WINHIGH	Defines the upper 16 bits of the 32-bit window for the windowed mode of operation of the watchdog. It is defined in terms of cycles of the watchdog clock. In this mode the watchdog can be refreshed only when the timer has reached a value greater than or equal to this window length. A refresh outside this window resets the system or if IRQRSTEN is set, it interrupts and then resets the system.

24.7.6 Watchdog Window Register Low (WDOG_WINL)

NOTE

You must set the Window Register value lower than the Time-out Value Register.

Address: WDOG_WINL is 4005_2000h base + Ah offset = 4005_200Ah

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

WDOG_WINL field descriptions

Field	Description
15–0 WINLOW	Defines the lower 16 bits of the 32-bit window for the windowed mode of operation of the watchdog. It is defined in terms of cycles of the pre-scaled watchdog clock. In this mode, the watchdog can be refreshed only when the timer reaches a value greater than or equal to this window length value. A refresh outside this window resets the system or if IRQRSTEN is set, it interrupts and then resets the system.

24.7.7 Watchdog Refresh Register (WDOG_REFRESH)

Address: WDOG_REFRESH is 4005_2000h base + Ch offset = 4005_200Ch

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

WDOG_REFRESH field descriptions

Field	Description
15–0 WDOGREFRES H	Watchdog refresh register. A sequence of 0xA602 followed by 0xB480 within 20 bus clock cycles when written to this register, refreshes the WDOG and prevents it from resetting the system. Writing a value other than the above mentioned sequence or if the sequence is longer than 20 bus cycles, resets the system or if IRQRSTEN is set, it interrupts and then resets the system).

24.7.8 Watchdog Unlock Register (WDOG_UNLOCK)

Address: WDOG_UNLOCK is 4005_2000h base + Eh offset = 4005_200Eh

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

WDOG_UNLOCK field descriptions

Field	Description
15–0 WDOGUNLOCK	You can write the unlock sequence values to this register to make the watchdog write once registers writable again. The required unlock sequence is 0xC520 followed by 0xD928 within 20 bus clock cycles. A valid unlock sequence opens up a window equal in length to the WCT within which you can update the registers. Writing a value other than the above mentioned sequence or if the sequence is longer than 20 bus cycles, resets the system or if IRQRSTEN is set, it interrupts and then resets the system). The unlock sequence is effective only if ALLOWUPDATE is set.

24.7.9 Watchdog Timer Output Register High (WDOG_TMROUTH)

Address: WDOG_TMROUTH is 4005_2000h base + 10h offset = 4005_2010h

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

WDOG_TMROUTH field descriptions

Field	Description
15–0 TIMEROUTHIGH	Shows the value of the upper 16 bits of the watchdog timer.

24.7.10 Watchdog Timer Output Register Low (WDOG_TMROUTL)

During stop mode, the WDOG_TIMER_OUT will be caught at the pre-stop value of the watchdog timer. After exiting stop mode, a maximum delay of 1 WDOG_CLK cycle + 3 bus clock cycles will occur before the WDOG_TIMER_OUT starts following the watchdog timer.

Address: WDOG_TMROUTL is 4005_2000h base + 12h offset = 4005_2012h

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

WDOG_TMROUTL field descriptions

Field	Description
15–0 TIMEROUTLOW	Shows the value of the lower 16 bits of the watchdog timer.

24.7.11 Watchdog Reset Count Register (WDOG_RSTCNT)

Address: WDOG_RSTCNT is 4005_2000h base + 14h offset = 4005_2014h

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

WDOG_RSTCNT field descriptions

Field	Description
15–0 RSTCNT	Counts the number of times the watchdog resets the system. This register is reset only on a POR. Writing 1 to the bit to be cleared, enables you to clear the contents of this register.

24.7.12 Watchdog Prescaler Register (WDOG_PRESC)

Address: WDOG_PRESC is 4005_2000h base + 16h offset = 4005_2016h

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

WDOG_PRESC field descriptions

Field	Description
15–11 Reserved	This read-only field is reserved and always has the value zero.
10–8 PRESCVAL	3-bit prescaler for the watchdog clock source. A value of zero indicates no division of the input WDOG clock. The watchdog clock is divided by (PRESCVAL + 1) to provide the prescaled WDOG_CLK.
7–0 Reserved	This read-only field is reserved and always has the value zero.

24.8 Watchdog Operation with 8-bit access

This section discusses 8-bit access considerations.

24.8.1 General Guideline

When performing 8-bit accesses to the watchdog's 16-bit registers where the intention is to access both the bytes of a register, you must try to place the two 8-bit accesses one after the other in your code.

24.8.2 Refresh and Unlock operations with 8-bit access

One exception condition that generates a reset to the system, is the write of any value other than those required for a legal refresh/update sequence to the respective refresh and unlock registers.

For an 8-bit access to these registers, writing a correct value requires at least two bus clock cycles that means there is an invalid value in the registers for one cycle. Therefore, the system is reset even if the intention is to write a correct value to the refresh/unlock register. Keeping this in mind the exception condition for 8-bit accesses is slightly modified. Whereas the match for a correct value for a refresh/unlock sequence is as per the original definition, the match for an incorrect value is done byte-wise on the refresh/ unlock rather than for the whole 16-bit value. This means that if the high byte of the refresh/unlock register contains any value other than high bytes of the two values making up the sequence, it is treated as an exception condition, leading to a reset or interrupt-then-reset. The same holds true for the lower byte of the refresh or unlock register. Let us take the refresh operation that expects a write of 0xA602 followed by 0xB480 to the refresh register, as an example.

Table 24-14. Refresh for 8-bit Access

	WDOG_REFRESH[15:8]	WDOG_REFRESH[7:0]	Sequence value1 or value2 match	Mismatch exception
Current Value	0xB4	0x80	Value2 match	No
Write 1	0xB4	0x02	No match	No
Write 2	0xA6	0x02	Value1 match	No
Write 3	0xB4	0x02	No match	No
Write 4	0xB4	0x80	Value2 match. Sequence complete.	No
Write 5	0x02	0x80	No match	Yes

As shown in the preceding table, the refresh register holds its reset value initially. Thereafter, two 8-bit accesses are performed on the register to write the first value of the refresh sequence. No mismatch exception is registered on the intermediate write, Write1. The sequence is completed by performing two more 8-bit accesses, writing in the second value of the sequence for a successful refresh. It must be noted that the match of value2 takes place only when the complete 16-bit value is correctly written, write4. Hence, the requirement of writing value2 of the sequence within 20 bus clock cycles of value1 is checked by measuring the gap between write2 and write4.

It is reiterated that the condition for matching values 1 and 2 of the refresh or unlock sequence remains unchanged. It is just the criterion for detecting a wrong value in these registers which has been relaxed, as explained, for 8-bit accesses. Any 16-bit access still needs to adhere to the original guidelines, mentioned in the sections Refreshing the Watchdog.

24.9 Restrictions on Watchdog Operation

This section mentions some exceptions to the watchdog operation that may not be apparent to you.

- Restriction on unlock / refresh operations—In the period between the closure of the WCT window (after unlock) and the actual reload of the watchdog timer, unlock and refresh operations need not be attempted.
- The update and reload of the watchdog timer happens two to three watchdog clocks after WCT window closes, following a successful configuration on unlock.
- Clock Switching Delay—The watchdog uses glitch free multiplexers at two places – one to choose between the LPO oscillator input and alternate clock input and the other to choose between the watchdog functional clock and fast clock input for watchdog functional test. A maximum time period of ~ 2 clock A cycles plus ~2 clock B cycles elapses from the time a switch is requested to the occurrence of the actual clock switch (clock A and B are the two input clocks to the clock mux).
- For the windowed mode, there is a two to three bus clock latency between the watchdog counter going past the window value and the same registering in the bus clock domain.
- For proper operation of the watchdog, the watchdog clock must be at least five times slower than the system bus clock at all times. An exception is the case when the watchdog clock is synchronous to the bus clock wherein the watchdog clock can be as fast as the bus clock.
- WCT must be equivalent to at least three watchdog clock cycles. If not ensured, this means that even after the close of the WCT window, you have to wait for the synchronized system reset to deassert in the watchdog clock domain, before expecting the configuration updates to take effect.
- The time-out value of the watchdog should be set to a minimum of four watchdog clock cycles. This is to take into account the delay in new settings taking effect in the watchdog clock domain.
- You must take care not only to refresh the watchdog within the watchdog timer's actual time-out period, but also provide enough allowance for the time it takes for the refresh sequence to be detected by the watchdog timer, on the watchdog clock.
- Updates cannot be made in the bus clock cycle immediately following the write of the unlock sequence, but one bus clock cycle later.

- It should be ensured that the time-out value for the watchdog is always greater than 2xWCT time + 20 bus clock cycles.
- An attempted refresh operation, in between the two writes of the unlock sequence and in the WCT time following a successful unlock, will go undetected.
- Trying to unlock the watchdog within the WCT time after an initial unlock has no effect.
- The refresh and unlock operations and interrupt are not automatically disabled in the watchdog functional test mode.
- After emerging from a reset due to a watchdog functional test, you are still expected to go through the mandatory steps of unlocking and configuring the watchdog. The watchdog continues to be in its functional test mode and therefore you should pull the watchdog out of the functional test mode within WCT time of reset.
- After emerging from a reset due to a watchdog functional test, you still need to go through the mandatory steps of unlocking and configuring the watchdog.
- You must ensure that both the clock inputs to the glitchless clock multiplexers are alive during the switching of clocks. Failure to do so results in a loss of clock at their outputs.
- There is a gap of two to three watchdog clock cycles from the point that stop mode is entered to the watchdog timer actually pausing, due to synchronization. The same holds true for an exit from the stop mode, this time resulting in a two to three watchdog clock cycle delay in the timer restarting. In case the duration of the stop mode is less than one watchdog clock cycle, the watchdog timer is not guaranteed to pause.
- Consider the case when the first refresh value is written, following which the system enters stop mode (with system bus clk still on). Now, if the second refresh value is not written within 20 bus cycles of the first value, the system is reset (or interrupt-then-reset if enabled).

Chapter 25
Multipurpose Clock Generator (MCG)

25.1 Introduction

NOTE

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

The multipurpose clock generator (MCG) module provides several clock source choices for the MCU. The module contains a frequency-locked loop (FLL) and a phase-locked loop (PLL). The FLL is controllable by either an internal or an external reference clock. The PLL is controllable by the external reference clock. The module can select either of the FLL or PLL output clocks, or either of the internal or external reference clocks as a source for the MCU system clock. The MCG operates in conjuction with a crystal oscillator, which allows an external crystal, ceramic resonator, or another external clock source to produce the external reference clock.

25.1.1 Features

Key features of the MCG module are:

- Frequency-locked loop (FLL)
  - Digitally-controlled oscillator (DCO)
  - DCO frequency range is programmable for up to four different frequency ranges.
  - Option to program and maximize DCO output frequency for a low frequency external reference clock source.
  - Option to prevent FLL from resetting its current locked frequency when switching clock modes if FLL reference frequency is not changed.

    - Internal or external reference clock can be used as the FLL source.
    - Can be used as a clock source for other on-chip peripherals.
- Phase-locked loop (PLL)
    - Voltage-controlled oscillator (VCO)
    - Selectable External reference clock is used as the PLL source
    - Modulo VCO frequency divider
    - Phase/Frequency detector
    - Integrated loop filter
    - Can be used as a clock source for other on-chip peripherals.
- Internal reference clock generator
    - Slow clock with nine trim bits for accuracy
    - Fast clock with four trim bits
    - Can be used as source clock for the FLL. In FEI mode, only the slow Internal Reference Clock (IRC) can be used as the FLL source.
    - Either the slow or the fast clock can be selected as the clock source for the MCU
    - Can be used as a clock source for other on-chip peripherals
- Control signals for both of "the MCG external reference low power oscillator clock generators are provided:
    - HGO0, RANGE0, EREFS0

      HGO1, RANGE1, EREFS1
- External clock from the Crystal Oscillator (OSC0)
    - Can be used as a source for the FLL and/or the PLL.
    - Can be selected as the clock source for the MCU
- External clock from the Real Time Counter (RTC)
    - Can only be used as a source for the FLL.
    - Can be selected as the clock source for the MCU
- External clock from the Crystal Oscillator (OSC1)
    - Can only be used as a source for the PLL.

- External clock monitor with reset and interrupt request capability to check for external clock failure when running in FBE, PEE, BLPE, or FEE modes
- Lock detector with interrupt request capability for use with the PLL
- Internal Reference Clocks Auto Trim Machine (ATM) capability using an external clock as a reference
- Reference dividers for both the FLL and PLL are provided
- Reference dividers for the Fast Internal Reference Clock are provided
- MCG PLL0 Clock (MCGPLL0CLK) is provided as a clock source for other on-chip peripherals
- MCG PLL1 Clock (MCGPLL1CLK) is provided as a clock source for other on-chip peripherals
- MCG PLL0 2X Clock (MCGPLL0CLK2X) is provided as a clock source for other on-chip peripherals
- MCG PLL1 2X Clock (MCGPLL1CLK2X) is provided as a clock source for other on-chip peripherals
- MCG FLL Clock (MCGFLLCLK) is provided as a clock source for other on-chip peripherals
- MCG Fixed Frequency Clock (MCGFFCLK) is provided as a clock source for other on-chip peripherals
- MCG Internal Reference Clock (MCGIRCLK) is provided as a clock source for other on-chip peripherals.

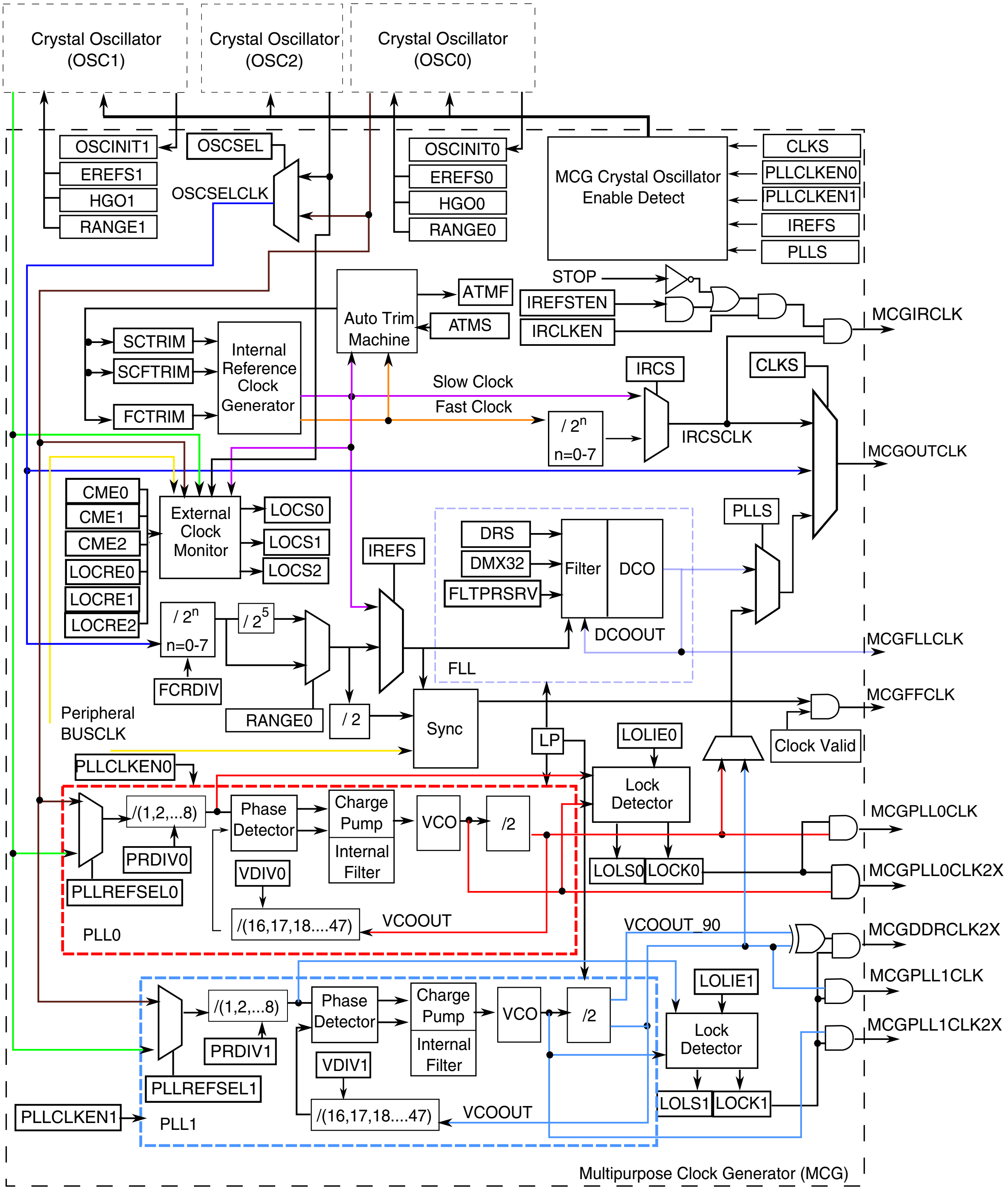


Figure 25-1. Multipurpose Clock Generator (MCG) Block Diagram

NOTE

Refer to the chip configuration chapter to identify the oscillator used in this MCU.

25.1.2 Modes of Operation

There are nine modes of operation for the MCG: FEI, FEE, FBI, FBE, PBE, PEE, BLPI, BLPE, and Stop. For details, see MCG Modes of Operation.

25.2 External Signal Description

There are no MCG signals that connect off chip.

25.3 Memory Map/Register Definition

This section includes the memory map and register definition.

The MCG registers can only be written to when in supervisor mode. Write accesses when in user mode will result in a bus error. Read accesses may be performed in both supervisor and user modes.

MCG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_4000	MCG Control 1 Register (MCG_C1)	8	R/W	04h	25.3.1/ 610
4006_4001	MCG Control 2 Register (MCG_C2)	8	R/W	80h	25.3.2/ 611
4006_4002	MCG Control 3 Register (MCG_C3)	8	R/W	Undefined	25.3.3/ 613
4006_4003	MCG Control 4 Register (MCG_C4)	8	R/W	Undefined	25.3.4/ 613
4006_4004	MCG Control 5 Register (MCG_C5)	8	R/W	00h	25.3.5/ 615
4006_4005	MCG Control 6 Register (MCG_C6)	8	R/W	00h	25.3.6/ 616
4006_4006	MCG Status Register (MCG_S)	8	R	10h	25.3.7/ 617
4006_4008	MCG Status and Control Register (MCG_SC)	8	R/W	02h	25.3.8/ 619

Table continues on the next page...

MCG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_400A	MCG Auto Trim Compare Value High Register (MCG_ATCVH)	8	R/W	00h	25.3.9/ 620
4006_400B	MCG Auto Trim Compare Value Low Register (MCG_ATCVL)	8	R/W	00h	25.3.10/ 621
4006_400C	MCG Control 7 Register (MCG_C7)	8	R/W	00h	25.3.11/ 621
4006_400D	MCG Control 8 Register (MCG_C8)	8	R/W	See section	25.3.12/ 622
4006_400F	MCG Control 10 Register (MCG_C10)	8	R/W	80h	25.3.13/ 623
4006_4010	MCG Control 11 Register (MCG_C11)	8	R/W	00h	25.3.14/ 624
4006_4011	MCG Control 12 Register (MCG_C12)	8	R/W	00h	25.3.15/ 625
4006_4012	MCG Status 2 Register (MCG_S2)	8	R	00h	25.3.16/ 626

25.3.1 MCG Control 1 Register (MCG_C1)

Address: MCG_C1 is 4006_4000h base + 0h offset = 4006_4000h

Bit	7	6	5	4	3	2	1	0
Read Write	CLKS		FRDIV			IREFS	IRCLKEN	IREFSTEN
Reset	0	0	0	0	0	1	0	0

MCG_C1 field descriptions

Field	Description
7–6 CLKS	Clock Source Select Selects the clock source for MCGOUTCLK . 00 Encoding 0 — Output of FLL or PLLCS is selected (depends on PLLS control bit). 01 Encoding 1 — Internal reference clock is selected. 10 Encoding 2 — External reference clock is selected. 11 Encoding 3 — Reserved.
5–3 FRDIV	FLL External Reference Divider Selects the amount to divide down the external reference clock for the FLL. The resulting frequency must be in the range 31.25 kHz to 39.0625 kHz (This is required when FLL/DCO is the clock source for MCGOUTCLK . In FBE mode, it is not required to meet this range, but it is recommended in the cases when trying to enter a FLL mode from FBE).

Table continues on the next page...

MCG_C1 field descriptions (continued)

Field	Description
	000 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 1; for all other RANGE 0 values, Divide Factor is 32. 001 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 2; for all other RANGE 0 values, Divide Factor is 64. 010 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 4; for all other RANGE 0 values, Divide Factor is 128. 011 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 8; for all other RANGE 0 values, Divide Factor is 256. 100 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 16; for all other RANGE 0 values, Divide Factor is 512. 101 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 32; for all other RANGE 0 values, Divide Factor is 1024. 110 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 64; for all other RANGE 0 values, Divide Factor is Reserved . 111 If RANGE 0 = 0 or OSCSEL=1 , Divide Factor is 128; for all other RANGE 0 values, Divide Factor is Reserved .
2 IREFS	Internal Reference Select Selects the reference clock source for the FLL. 0 External reference clock is selected. 1 The slow internal reference clock is selected.
1 IRCLKEN	Internal Reference Clock Enable Enables the internal reference clock for use as MCGIRCLK. 0 MCGIRCLK inactive. 1 MCGIRCLK active.
0 IREFSTEN	Internal Reference Stop Enable Controls whether or not the internal reference clock remains enabled when the MCG enters Stop mode. 0 Internal reference clock is disabled in Stop mode. 1 Internal reference clock is enabled in Stop mode if IRCLKEN is set or if MCG is in FEI, FBI, or BLPI modes before entering Stop mode.

25.3.2 MCG Control 2 Register (MCG_C2)

Address: MCG_C2 is 4006_4000h base + 1h offset = 4006_4001h

Bit	7	6	5	4	3	2	1	0
Read	LOCRE0	0	RANGE0		HGO0	EREFS0	LP	IRCS
Write		▨						
Reset	1	0	0	0	0	0	0	0

MCG_C2 field descriptions

Field	Description
7 LOCRE0	Loss of Clock Reset Enable Determines if a interrupt or a reset request is made following a loss of OSC0 external reference clock. The LOCRE0 only has an affect when CME0 is set. 0 Interrupt request is generated on a loss of OSC0 external reference clock. 1 Generate a reset request on a loss of OSC0 external reference clock
6 Reserved	This read-only field is reserved and always has the value zero.
5–4 RANGE0	Frequency Range Select Selects the frequency range for the crystal oscillator or external clock source. Refer to the Oscillator (OSC) chapter for more details and the device data sheet for the frequency ranges used. 00 Encoding 0 — Low frequency range selected for the crystal oscillator . 01 Encoding 1 — High frequency range selected for the crystal oscillator . 1X Encoding 2 — Very high frequency range selected for the crystal oscillator .
3 HGO0	High Gain Oscillator Select Controls the crystal oscillator mode of operation. Refer to the Oscillator (OSC) chapter for more details. 0 Configure crystal oscillator for low-power operation. 1 Configure crystal oscillator for high-gain operation.
2 EREFS0	External Reference Select Selects the source for the external reference clock. Refer to the Oscillator (OSC) chapter for more details. 0 External reference clock requested. 1 Oscillator requested.
1 LP	Low Power Select Controls whether the FLL (or PLL) is disabled in BLPI and BLPE modes. In FBE or PBE modes, setting this bit to 1 will transition the MCG into BLPE mode; in FBI mode, setting this bit to 1 will transition the MCG into BLPI mode. In any other MCG mode, LP bit has no affect. 0 FLL (or PLL) is not disabled in bypass modes. 1 FLL (or PLL) is disabled in bypass modes (lower power)
0 IRCS	Internal Reference Clock Select Selects between the fast or slow internal reference clock source. 0 Slow internal reference clock selected. 1 Fast internal reference clock selected.

25.3.3 MCG Control 3 Register (MCG_C3)

Address: MCG_C3 is 4006_4000h base + 2h offset = 4006_4002h

Bit	7	6	5	4	3	2	1	0
Read Write	SCTRIM							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

MCG_C3 field descriptions

Field	Description
7–0 SCTRIM	Slow Internal Reference Clock Trim Setting SCTRIM[1] controls the slow internal reference clock frequency by controlling the slow internal reference clock period. The SCTRIM bits are binary weighted (that is, bit 1 adjusts twice as much as bit 0). Increasing the binary value increases the period, and decreasing the value decreases the period. An additional fine trim bit is available in C4 register as the SCFTRIM bit. Upon reset this value is loaded with a factory trim value. If an SCTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.

[1] A value for SCTRIM is loaded during reset from a factory programmed location .

25.3.4 MCG Control 4 Register (MCG_C4)

NOTE

Reset values for DRST and DMX32 bits are 0.

Address: MCG_C4 is 4006_4000h base + 3h offset = 4006_4003h

Bit	7	6	5	4	3	2	1	0
Read Write	DMX32	DRST_DRS		FCTRIM				SCFTRIM
Reset	0	0	0	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.
- A value for FCTRIM is loaded during reset from a factory programmed location . x = Undefined at reset.

MCG_C4 field descriptions

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>7
DMX32</td><td>DCO Maximum Frequency with 32.768 kHz Reference

The DMX32 bit controls whether or not the DCO frequency range is narrowed to its maximum frequency with a 32.768 kHz reference.

The following table identifies settings for the DCO frequency range.

<b>NOTE:</b> The system clocks derived from this source should not exceed their specified maximums.

<table>
<tr><td>DRST_DRS</td><td>DMX32</td><td>Reference Range</td><td>FLL Factor</td><td>DCO Range</td></tr>
<tr><td>00</td><td>0</td><td>31.25-39.0625 kHz</td><td>640</td><td>20-25 MHz</td></tr>
<tr><td></td><td>1</td><td>32.768 kHz</td><td>732</td><td>24 MHz</td></tr>
<tr><td>01</td><td>0</td><td>31.25-39.0625 kHz</td><td>1280</td><td>40-50 MHz</td></tr>
<tr><td></td><td>1</td><td>32.768 kHz</td><td>1464</td><td>48 MHz</td></tr>
<tr><td>10</td><td>0</td><td>31.25-39.0625 kHz</td><td>1920</td><td>60-75 MHz</td></tr>
<tr><td></td><td>1</td><td>32.768 kHz</td><td>2197</td><td>72 MHz</td></tr>
<tr><td>11</td><td>0</td><td>31.25-39.0625 kHz</td><td>2560</td><td>80-100 MHz</td></tr>
<tr><td></td><td>1</td><td>32.768 kHz</td><td>2929</td><td>96 MHz</td></tr>
</table>

0 DCO has a default range of 25%.
1 DCO is fine-tuned for maximum frequency with 32.768 kHz reference.</td></tr>
<tr><td>6–5
DRST_DRS</td><td>DCO Range Select

The DRS bits select the frequency range for the FLL output, DCOOUT. When the LP bit is set, writes to the DRS bits are ignored. The DRST read field indicates the current frequency range for DCOOUT. The DRST field does not update immediately after a write to the DRS field due to internal synchronization between clock domains. Refer to DCO Frequency Range table for more details.

00 Encoding 0 — Low range (reset default).
01 Encoding 1 — Mid range.
10 Encoding 2 — Mid-high range.
11 Encoding 3 — High range.</td></tr>
<tr><td>4–1
FCTRIM</td><td>Fast Internal Reference Clock Trim Setting

FCTRIM [1] controls the fast internal reference clock frequency by controlling the fast internal reference clock period. The FCTRIM bits are binary weighted (that is, bit 1 adjusts twice as much as bit 0). Increasing the binary value increases the period, and decreasing the value decreases the period.

If an FCTRIM[3:0] value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.</td></tr>
<tr><td>0
SCFTRIM</td><td>Slow Internal Reference Clock Fine Trim

SCFTRIM [2] controls the smallest adjustment of the slow internal reference clock frequency. Setting SCFTRIM increases the period and clearing SCFTRIM decreases the period by the smallest amount possible.

If an SCFTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this bit.</td></tr>
</table>

[1] A value for FCTRIM is loaded during reset from a factory programmed location .

2. A value for SCFTRIM is loaded during reset from a factory programmed location .

25.3.5 MCG Control 5 Register (MCG_C5)

Address: MCG_C5 is 4006_4000h base + 4h offset = 4006_4004h

Bit	7	6	5	4	3	2	1	0
Read	PLLREFSEL0	PLLCLKEN0	PLLSTEN0	0		PRDIV0		
Write				▨				
Reset	0	0	0	0	0	0	0	0

MCG_C5 field descriptions

Field	Description
7 PLLREFSEL0	PLL0 External Reference Select Selects PLL0 external reference clock source. 0 Selects OSC0 clock source as its external reference clock. 1 Selects OSC1 clock source as its external reference clock.
6 PLLCLKEN0	PLL Clock Enable Enables PLL0 independent of PLLS and enables the PLL0 clock for use as MCGPLL0CLK and MCGPLL0CLK2X. (PRDIV0 needs to be programmed to the correct divider to generate a PLL1 reference clock in a valid reference range prior to setting the PLLCLKEN0 bit). Setting PLLCLKEN0 will enable the external oscillator selected by REFSEL if not already enabled. Whenever the PLL0 is being enabled by means of the PLLCLKEN0 bit, and the external oscillator is being used as the reference clock, the OSCINIT 0 bit should be checked to make sure it is set. 0 MCGPLL0CLK and MCGPLL0CLK2X are inactive. 1 MCGPLL0CLK and MCGPLL0CLK2X are active.
5 PLLSTEN0	PLL0 Stop Enable Enables the PLL0 Clock during Normal Stop (In Low Power Stop mode, the PLL0 clock gets disabled even if PLLSTEN0=1). All other power modes, PLLSTEN0 bit has no affect and does not enable the PLL0 Clock to run if it is written to 1. 0 MCGPLL0CLK and MCGPLL0CLK2X are disabled in any of the Stop modes. 1 MCGPLL0CLK and MCGPLL0CLK2X are enabled if system is in Normal Stop mode.
4–3 Reserved	Reserved This read-only field is reserved and always has the value zero.
2–0 PRDIV0	PLL0 External Reference Divider

Table continues on the next page...

MCG_C5 field descriptions (continued)

Field	Description
	Selects the amount to divide down the external reference clock for the PLL0. The resulting frequency must be in a valid reference range. After the PLL0 is enabled (by setting either PLLCLKEN0 or PLLS), the PRDIV0 value must not be changed when LOCK0 is zero.

Table 25-7. PLL0 External Reference Divide Factor

PRDIV0	Divide Factor
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

25.3.6 MCG Control 6 Register (MCG_C6)

Address: MCG_C6 is 4006_4000h base + 5h offset = 4006_4005h

Bit	7	6	5	4	3	2	1	0
Read Write	LOLIE0	PLLS	CME0	VDIV0				
Reset	0	0	0	0	0	0	0	0

MCG_C6 field descriptions

Field	Description
7 LOLIE0	Loss of Lock Interrrupt Enable Determines if an interrupt request is made following a loss of lock indication. This bit only has an effect when LOLS 0 is set. 0 No interrupt request is generated on loss of lock. 1 Generate an interrupt request on loss of lock.
6 PLLS	PLL Select Controls whether the PLLCS or FLL output is selected as the MCG source when CLKS[1:0]=00. If the PLLS bit is cleared and PLLCLKEN0 and PLLCLKEN1 is not set, the PLLCS output clock is disabled in all modes. If the PLLS is set, the FLL is disabled in all modes.

Table continues on the next page...

MCG_C6 field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>0 FLL is selected.
1 PLLCS output clock is selected (PRDIV0 bits of PLL in control need to be programmed to the correct divider to generate a PLL reference clock in the range of 1 - 32 MHz prior to setting the PLLS bit).</td></tr>
<tr><td>5
CME0</td><td>Clock Monitor Enable

Enables the loss of clock monitoring circuit for the OSC0 external reference mux select. The LOCRE0 bit will determine if a interrupt or a reset request is generated following a loss of OSC0 indication. The CME0 bit should only be set to a logic 1 when the MCG is in an operational mode that uses the external clock (FEE, FBE, PEE, PBE, or BLPE) . Whenever the CME0 bit is set to a logic 1, the value of the RANGE0 bits in the C2 register should not be changed. CME0 bit should be set to a logic 0 before the MCG enters any Stop mode. Otherwise, a reset request may occur while in Stop mode. CME0 should also be set to a logic 0 before entering VLPR or VLPW power modes if the MCG is in BLPE mode.

0 External clock monitor is disabled for OSC0.
1 External clock monitor is enabled for OSC0.</td></tr>
<tr><td>4–0
VDIV0</td><td>VCO0 Divider

Selects the amount to divide the VCO output of the PLL0. The VDIV0 bits establish the multiplication factor (M) applied to the reference clock frequency. After the PLL0 is enabled (by setting either PLLCLKEN0 or PLLS), the VDIV0 value must not be changed when LOCK0 is zero.

<b>Table 25-9. PLL0 VCO Divide Factor</b>
<table>
<tr><th>VDIV0</th><th>Multiply Factor</th><th></th><th>VDIV0</th><th>Multiply Factor</th><th></th><th>VDIV0</th><th>Multiply Factor</th><th></th><th>VDIV0</th><th>Multiply Factor</th></tr>
<tr><td>00000</td><td>16</td><td></td><td>01000</td><td>24</td><td></td><td>10000</td><td>32</td><td></td><td>11000</td><td>40</td></tr>
<tr><td>00001</td><td>17</td><td></td><td>01001</td><td>25</td><td></td><td>10001</td><td>33</td><td></td><td>11001</td><td>41</td></tr>
<tr><td>00010</td><td>18</td><td></td><td>01010</td><td>26</td><td></td><td>10010</td><td>34</td><td></td><td>11010</td><td>42</td></tr>
<tr><td>00011</td><td>19</td><td></td><td>01011</td><td>27</td><td></td><td>10011</td><td>35</td><td></td><td>11011</td><td>43</td></tr>
<tr><td>00100</td><td>20</td><td></td><td>01100</td><td>28</td><td></td><td>10100</td><td>36</td><td></td><td>11100</td><td>44</td></tr>
<tr><td>00101</td><td>21</td><td></td><td>01101</td><td>29</td><td></td><td>10101</td><td>37</td><td></td><td>11101</td><td>45</td></tr>
<tr><td>00110</td><td>22</td><td></td><td>01110</td><td>30</td><td></td><td>10110</td><td>38</td><td></td><td>11110</td><td>46</td></tr>
<tr><td>00111</td><td>23</td><td></td><td>01111</td><td>31</td><td></td><td>10111</td><td>39</td><td></td><td>11111</td><td>47</td></tr>
</table>
</td></tr>
</table>

25.3.7 MCG Status Register (MCG_S)

Address: MCG_S is 4006_4000h base + 6h offset = 4006_4006h

Bit	7	6	5	4	3	2	1	0
Read	LOLS0	LOCK0	PLLST	IREFST	CLKST		OSCINIT0	IRCST
Write								
Reset	0	0	0	1	0	0	0	0

MCG_S field descriptions

Field	Description
7 LOLS0	Loss of Lock Status This bit is a sticky bit indicating the lock status for the PLL. LOLS 0 is set if after acquiring lock, the PLL output frequency has fallen outside the lock exit frequency tolerance, D $_{unl}$. LOLIE 0 determines whether an interrupt request is made when LOLS 0 is set. This bit is cleared by reset or by writing a logic 1 to it when set. Writing a logic 0 to this bit has no effect. 0 PLL has not lost lock since LOLS 0 was last cleared. 1 PLL has lost lock since LOLS 0 was last cleared.
6 LOCK0	Lock Status This bit indicates whether the PLL0 has acquired lock. Lock detection is disabled when not operating in either PBE or PEE mode unless PLLCLKEN0=1 and the MCG is not configured in BLPI or BLPE mode. While the PLL0 clock is locking to the desired frequency, MCGPLL0CLK and MCGPLL0CLK2X will be gated off until the LOCK0 bit gets asserted. If the lock status bit is set, changing the value of the PRDIV0[2:0] bits in the C5 register or the VDIV0[4:0] bits in the C6 register causes the lock status bit to clear and stay cleared until the PLL0 has reacquired lock. Loss of PLL0 reference clock will also cause the LOCK0 bit to clear until PLL0 has a Entry into LLS, VLPS, or regular Stop with PLLSTEN0=0 also causes the lock status bit to clear and stay cleared until the Stop mode is exited and the PLL0 has reacquired lock. Any time the PLL0 is enabled and the LOCK0 bit is cleared, the MCGPLL0CLK and MCGPLL0CLK2X will be gated off until the LOCK0 bit is asserted again. 0 PLL is currently unlocked. 1 PLL is currently locked.
5 PLLST	PLL Select Status This bit indicates the clock source selected by PLLS . The PLLST bit does not update immediately after a write to the PLLS bit due to internal synchronization between clock domains. 0 Source of PLLS clock is FLL clock. 1 Source of PLLS clock is PLLCS output clock.
4 IREFST	Internal Reference Status This bit indicates the current source for the FLL reference clock. The IREFST bit does not update immediately after a write to the IREFS bit due to internal synchronization between clock domains. 0 Source of FLL reference clock is the external reference clock. 1 Source of FLL reference clock is the internal reference clock.
3–2 CLKST	Clock Mode Status These bits indicate the current clock mode. The CLKST bits do not update immediately after a write to the CLKS bits due to internal synchronization between clock domains. 00 Encoding 0 — Output of the FLL is selected (reset default). 01 Encoding 1 — Internal reference clock is selected. 10 Encoding 2 — External reference clock is selected. 11 Encoding 3 — Output of the PLL is selected.
1 OSCINIT0	OSC Initialization

Table continues on the next page...

MCG_S field descriptions (continued)

Field	Description
	This bit, which resets to 0, is set to 1 after the initialization cycles of the crystal oscillator clock have completed. After being set, the bit is cleared to 0 if the OSC is subsequently disabled. Refer to the OSC module's detailed description for more information.
0 IRCST	Internal Reference Clock Status The IRCST bit indicates the current source for the internal reference clock select clock (IRCSCLK). The IRCST bit does not update immediately after a write to the IRCS bit due to internal synchronization between clock domains. The IRCST bit will only be updated if the internal reference clock is enabled, either by the MCG being in a mode that uses the IRC or by setting the C1[IRCLKEN] bit . 0 Source of internal reference clock is the slow clock (32 kHz IRC). 1 Source of internal reference clock is the fast clock (2 MHz IRC).

25.3.8 MCG Status and Control Register (MCG_SC)

Address: MCG_SC is 4006_4000h base + 8h offset = 4006_4008h

Bit	7	6	5	4	3	2	1	0
Read	ATME	ATMS	ATMF	FLTPRSRV	FCRDIV			LOCS0
Write			▨					▨
Reset	0	0	0	0	0	0	1	0

MCG_SC field descriptions

Field	Description
7 ATME	Automatic Trim Machine Enable Enables the Auto Trim Machine to start automatically trimming the selected Internal Reference Clock. **NOTE:** ATME deasserts after the Auto Trim Machine has completed trimming all trim bits of the IRCS clock selected by the ATMS bit. Writing to C1, C3, C4, and SC registers or entering Stop mode aborts the auto trim operation and clears this bit. 0 Auto Trim Machine disabled. 1 Auto Trim Machine enabled.
6 ATMS	Automatic Trim Machine Select Selects the IRCS clock for Auto Trim Test. 0 32 kHz Internal Reference Clock selected. 1 4 MHz Internal Reference Clock selected.
5 ATMF	Automatic Trim machine Fail Flag Fail flag for the Automatic Trim Machine (ATM). This bit asserts when the Automatic Trim Machine is enabled (ATME=1) and a write to the C1, C3, C4, and SC registers is detected or the MCG enters into any Stop mode. A write to ATMF clears the flag.

Table continues on the next page...

MCG_SC field descriptions (continued)

Field	Description
	0 Automatic Trim Machine completed normally. 1 Automatic Trim Machine failed.
4 FLTPRSRV	FLL Filter Preserve Enable This bit will prevent the FLL filter values from resetting allowing the FLL output frequency to remain the same during clock mode changes where the FLL/DCO output is still valid. (Note: This requires that the FLL reference frequency to remain the same as what it was prior to the new clock mode switch. Otherwise FLL filter and frequency values will change.) 0 FLL filter and FLL frequency will reset on changes to currect clock mode. 1 Fll filter and FLL frequency retain their previous values during new clock mode change.
3–1 FCRDIV	Fast Clock Internal Reference Divider Selects the amount to divide down the fast internal reference clock. The resulting frequency will be in the range 31.25 kHz to 4 MHz (Note: Changing the divider when the Fast IRC is enabled is not supported). 000 Divide Factor is 1 001 Divide Factor is 2. 010 Divide Factor is 4. 011 Divide Factor is 8. 100 Divide Factor is 16 101 Divide Factor is 32 110 Divide Factor is 64 111 Divide Factor is 128.
0 LOCS0	OSC0 Loss of Clock Status The LOCS0 indicates when a loss of OSC0 reference clock has occurred. The LOCS0 bit only has an effect when CME0 is set. This bit is cleared by writing a logic 1 to it when set. 0 Loss of OSC0 has not occurred. 1 Loss of OSC0 has occurred.

25.3.9 MCG Auto Trim Compare Value High Register (MCG_ATCVH)

Address: MCG_ATCVH is 4006_4000h base + Ah offset = 4006_400Ah

Bit	7	6	5	4	3	2	1	0
Read Write	ATCVH							
Reset	0	0	0	0	0	0	0	0

MCG_ATCVH field descriptions

Field	Description
7–0 ATCVH	ATM Compare Value High Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion.

25.3.10 MCG Auto Trim Compare Value Low Register (MCG_ATCVL)

Address: MCG_ATCVL is 4006_4000h base + Bh offset = 4006_400Bh

Bit	7	6	5	4	3	2	1	0
Read Write	ATCVL							
Reset	0	0	0	0	0	0	0	0

MCG_ATCVL field descriptions

Field	Description
7–0 ATCVL	ATM Compare Value Low Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion.

25.3.11 MCG Control 7 Register (MCG_C7)

Address: MCG_C7 is 4006_4000h base + Ch offset = 4006_400Ch

Bit	7	6	5	4	3	2	1	0
Read	0							OSCSEL
Write								
Reset	0	0	0	0	0	0	0	0

MCG_C7 field descriptions

Field	Description
7–1 Reserved	This read-only field is reserved and always has the value zero.
0 OSCSEL	MCG OSC Clock Select Selects the MCG FLL external reference clock 0 Selects System Oscillator (OSCCLK). 1 Selects 32 kHz RTC Oscillator.

25.3.12 MCG Control 8 Register (MCG_C8)

Address: MCG_C8 is 4006_4000h base + Dh offset = 4006_400Dh

Bit	7	6	5	4	3	2	1	0
Read	LOCRE1	0	CME1	0				LOCS1
Write								
Reset	1	0	0	0	0	0	0	0

MCG_C8 field descriptions

Field	Description
7 LOCRE1	Loss of Clock Reset Enable Determines if a interrupt or a reset request is made following a loss of RTC external reference clock. The LOCRE1 only has an affect when CME1 is set. 0 Interrupt request is generated on a loss of RTC external reference clock. 1 Generate a reset request on a loss of RTC external reference clock
6 Reserved	This read-only field is reserved and always has the value zero.
5 CME1	Clock Monitor Enable1 Enables the loss of clock monitoring circuit for the output of the RTC external reference clock. The LOCRE1 bit will determine if a interrupt or a reset request is generated following a loss of RTC clock indication. The CME1 bit should only be set to a logic 1 when the MCG is in an operational mode that uses the external clock (FEE, FBE, PEE, PBE, or BLPE). CME1 bit must be set to a logic 0 before the MCG enters any Stop mode. Otherwise, a reset request may occur while in Stop mode. CME1 should also be set to a logic 0 before entering VLPR or VLPW power modes if the MCG is in BLPE mode. 0 External clock monitor is disabled for RTC clock. 1 External clock monitor is enabled for RTC clock.
4–1 Reserved	This read-only field is reserved and always has the value zero.
0 LOCS1	RTC Loss of Clock Status This bit indicates when a loss of clock has occurred. This bit is cleared by writing a logic 1 to it when set. 0 Loss of RTC has not occur. 1 Loss of RTC has occur

25.3.13 MCG Control 10 Register (MCG_C10)

Address: MCG_C10 is 4006_4000h base + Fh offset = 4006_400Fh

Bit	7	6	5	4	3	2	1	0
Read	LOCRE2	0	RANGE1		HGO1	EREFS1	0	
Write								
Reset	1	0	0	0	0	0	0	0

MCG_C10 field descriptions

Field	Description
7 LOCRE2	OSC1 Loss of Clock Reset Enable Determines if a interrupt or reset request is made following a loss of OSC1 external reference clock. The LOCRE2 only has an affect when LOCS2 is set. 0 Interrupt request is generated on a loss of OSC1 external reference clock. 1 Reset request is generated on a loss of OSC1 external reference clock.
6 Reserved	This read-only field is reserved and always has the value zero.
5–4 RANGE1	Frequency Range1 Select Selects the frequency range for the OSC1 crystal oscillator or external clock source. Refer to the Oscillator chapter for more details and the device data sheet for the frequency ranges used. 00 Encoding 0 — Low frequency range selected for the crystal oscillator . 01 Encoding 1 — High frequency range selected for the crystal oscillator . 1X Encoding 2 — Very high frequency range selected for the crystal oscillator .
3 HGO1	High Gain Oscillator1 Select Controls the OSC1 crystal oscillator mode of operation Refer to the Oscillator chapter for more details. 0 Configure crystal oscillator for low-power operation. 1 Configure crystal oscillator for high-gain operation.
2 EREFS1	External Reference Select Selects the source for the OSC1 external reference clock. Refer to the Oscillator chapter for more details. 0 External reference clock requested. 1 Oscillator requested.
1–0 Reserved	This read-only field is reserved and always has the value zero.

25.3.14 MCG Control 11 Register (MCG_C11)

Address: MCG_C11 is 4006_4000h base + 10h offset = 4006_4010h

Bit	7	6	5	4	3	2	1	0
Read	PLLREFSEL1	PLLCLKEN1	PLLSTEN1	PLLCS	0	PRDIV1		
Write								
Reset	0	0	0	0	0	0	0	0

MCG_C11 field descriptions

Field	Description
7 PLLREFSEL1	PLL1 External Reference Select Selects the PLL1 external reference clock source. 0 Selects OSC0 clock source as its external reference clock. 1 Selects OSC1 clock source as its external reference clock.
6 PLLCLKEN1	PLL1 Clock Enable Enables the PLL1 independent of PLLS and enables the PLL clocks for use as MCGPLL1CLK, MCGPLL1CLK2X, and MCGDDRCLK2X. (PRDIV1 needs to be programmed to the correct divider to generate a PLL1 reference clock in a valid reference range prior to setting the PLLCLKEN1 bit.) Setting PLLCLKEN1 will enable the PLL1 selected external oscillator if not already enabled. Whenever the PLL1 is being enabled by means of the PLLCLKEN1 bit, and the external oscillator is being used as the reference clock, the OSCINIT1 bit should be checked to make sure it is set. 0 MCGPLL1CLK, MCGPLL1CLK2X, and MCGDDRCLK2X are inactive 1 MCGPLL1CLK, MCGPLL1CLK2X, and MCGDDRCLK2X are active unless MCG is in a bypass mode with LP=1 (BLPI or BLPE).
5 PLLSTEN1	PLL1 Stop Enable Enables the PLL1 Clock during Normal Stop (In Low Power Stop modes, the PLL1 clock gets disabled even if PLLSTEN1=1. All other power modes, PLLSTEN1 bit has no affect and does not enable the PLL1 Clock to run if it is written to 1. 0 PLL1 clocks (MCGPLL1CLK, MCGPLL1CLK2X, and MCGDDRCLK2X) are disabled in any of the Stop modes. 1 PLL1 and its clocks (MCGPLL1CLK, MCGPLL1CLK2X, and MCGDDRCLK2X) are enabled if system is in Normal Stop mode.
4 PLLCS	PLL Clock Select Controls whether the PLL0 or PLL1 output is selected as the MCG source when CLKS are programmed in PLL Engaged External (PEE) mode (CLKS[1:0]=00 and IREFS=0 and PLLS=1). 0 PLL0 output clock is selected. 1 PLL1 output clock is selected.
3 Reserved	This read-only field is reserved and always has the value zero.
2–0 PRDIV1	PLL1 External Reference Divider

Table continues on the next page...

MCG_C11 field descriptions (continued)

Field	Description
	Selects the amount to divide down the external reference clock selected by REFSEL2 for PLL1. The resulting frequency must be in a valid reference range. After the PLL1 is enabled (by setting either PLLCLKEN1 or PLLS), the PRDIV1 value must not be changed when LOCK1 is zero.

Table 25-18. PLL1 External Reference Divide Factor

PRDIV1	Divide Factor
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

25.3.15 MCG Control 12 Register (MCG_C12)

Address: MCG_C12 is 4006_4000h base + 11h offset = 4006_4011h

Bit	7	6	5	4	3	2	1	0
Read	LOLIE1	0	CME2	VDIV1				
Write		▨						
Reset	0	0	0	0	0	0	0	0

MCG_C12 field descriptions

Field	Description
7 LOLIE1	PLL1 Loss of Lock Interrupt Enable Determines if an interrupt request is made following a loss of lock indication for PLL1. This bit only has an affect when LOLS1 is set. 0 No interrupt request is generated on loss of lock on PLL1. 1 Generate an interrupt request on loss of lock on PLL1.
6 Reserved	Reserved This read-only field is reserved and always has the value zero.
5 CME2	Clock Monitor Enable2 Enables the loss of clock monitor for the OSC1 external reference clock. LOCRE2 will determine if a reset or interrupt request is geneated following a loss of OSC1 external reference clock. The CME2 bit should only be set to a logic 1 when the MCG is in an operational mode that uses the external clock (PEE or

Table continues on the next page...

MCG_C12 field descriptions (continued)

Field	Description
	PBE) . Whenever the CME2 bit is set to a logic 1, the value of the RANGE1 bits in the C10 register should not be changed. CME2 bit should be set to a logic 0 before the MCG enters any Stop mode. Otherwise, a reset request may occur while in Stop mode. 0 External clock monitor for OSC1 is disabled. 1 Generate a reset request on loss of external clock on OSC1.
4–0 VDIV1	VCO1 Divider Selects the amount to divide the VCO output of the PLL1. The VDIV1 bits establish the multiplication factor (M) applied to the reference clock frequency. After the PLL1 is enabled (by setting either PLLCLKEN1 or PLLS), the VDIV1 value must not be changed when LOCK1 is zero.

Table 25-20. PLL1 VCO Divide Factor

VDIV1	Multiply Factor		VDIV1	Multiply Factor		VDIV1	Multiply Factor		VDIV1	Multiply Factor
00000	16		01000	24		10000	32		11000	40
00001	17		01001	25		10001	33		11001	41
00010	18		01010	26		10010	34		11010	42
00011	19		01011	27		10011	35		11011	43
00100	20		01100	28		10100	36		11100	44
00101	21		01101	29		10101	37		11101	45
00110	22		01110	30		10110	38		11110	46
00111	23		01111	31		10111	39		11111	47

25.3.16 MCG Status 2 Register (MCG_S2)

Address: MCG_S2 is 4006_4000h base + 12h offset = 4006_4012h

Bit	7	6	5	4	3	2	1	0
Read	LOLS1	LOCK1	0	PLLCST	0		OSCINIT1	LOCS2
Write								
Reset	0	0	0	0	0	0	0	0

MCG_S2 field descriptions

Field	Description
7 LOLS1	Loss of Lock2 Status This bit is a sticky bit indicating the lock status for the PLL1. LOLS1 is set if after acquiring lock, the PLL1 output frequency has fallen outside the lock exit frequency tolerance, D $_{unl}$. LOLIE1 determines whether an interrupt request is made when LOLS1 is set. This bit is cleared by reset or by writing a logic 1 to it when set. Writing a logic 0 to this bit has no effect.

Table continues on the next page...

MCG_S2 field descriptions (continued)

Field	Description
	0 PLL1 has not lost lock since LOLS1 was last cleared. 1 PLL1 has lost lock since LOLS1 was last cleared.
6 LOCK1	Lock1 Status This bit indicates whether PLL1 has acquired lock. PLL1 Lock detection is disabled when not operating in either PBE or PEE mode unless PLLCLKEN1=1 and the MCG is not configured in BLPI or BLPE mode. While the PLL1 clock is locking to the desired frequncy, MCGPLL1CLK, MCGPLL1CLK2X, and MCGDDRCLK2X will be gated off until the LOCK1 bit gets asserted. If the lock status bit is set, changing the value of the PRDIV1[2:0] bits in the C8 register or the VDIV2[4:0] bits in the C9 register causes the lock status bit to clear and stay cleared until the PLL1 has reacquired lock. Loss of PLL1 reference clock will also cause the LOCK1 bit to clear until PLL1 has reacquired lock. Entry into LLS, VLPS, or regular Stop with PLLSTEN1=0 also causes the lock status bit to clear and stay cleared until Stop mode is exited and the PLL1 has reacquired lock. Any time the PLL1 is enabled and the LOCK1 bit is cleared, the MCGPLL1CLK, MCGPLL1CLK2X, and MCGDDRCLK2X will be gated off until the LOCK1 bit is asserted again. 0 PLL1 is currently unlocked. 1 PLL1 is currently locked.
5 Reserved	This read-only field is reserved and always has the value zero.
4 PLLCST	PLL Clock Select Status The PLLCST indicates the PLL clock selected by PLLCS. The PLLCST bit does not updated immediately after a write to the PLLCS bit due internal synchronization between clock domains. 0 Source of PLLCS is PLL0 clock. 1 Source of PLLCS is PLL1 clock.
3–2 Reserved	This read-only field is reserved and always has the value zero.
1 OSCINIT1	OSC1 Initialization This bit is set after the initialization cycles of the 2nd crystal oscillator clock have completed. Refer to the Oscillator block guide for more details.
0 LOCS2	OSC1 Loss of Clock Status This bit indicates when a loss of OSC1 external reference clock has occurred. LOCRE2 determines if a reset or interrupt is generated when LOCS2 is set. This bit is cleared by writing a logic 1 to it when set. 0 No loss of OSC1 external reference clock has occurred. 1 Loss of OSC1 external reference clock has occurred.

25.4 Functional Description

25.4.1 MCG Mode State Diagram

The nine states of the MCG are shown in the following figure and are described in Table 25-22. The arrows indicate the permitted MCG mode transitions.

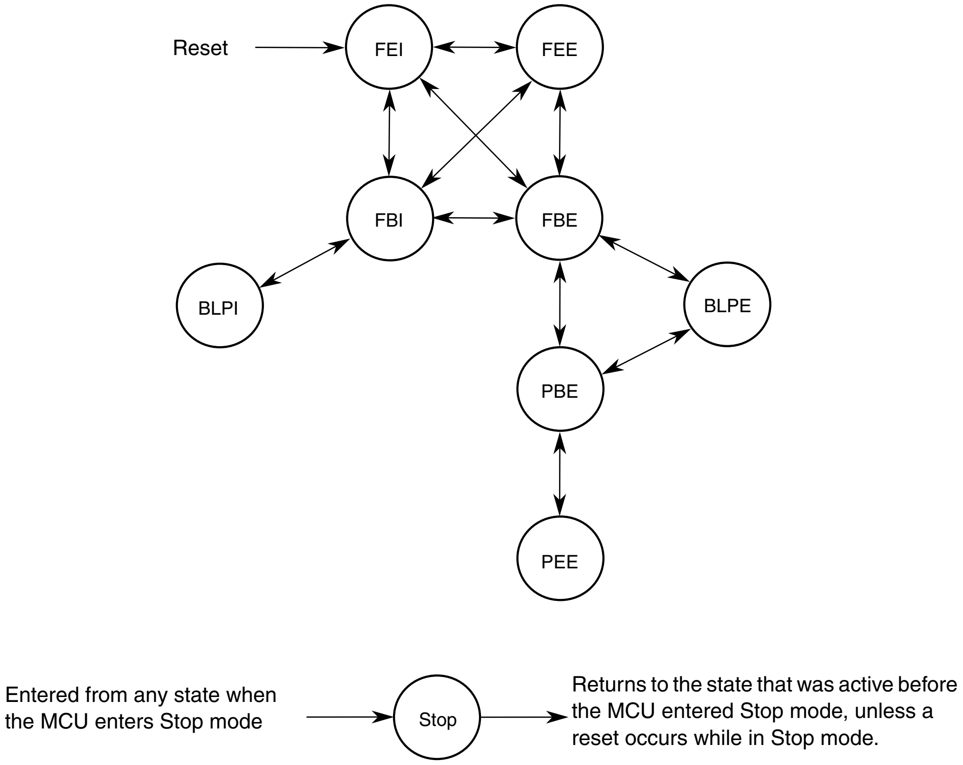

Reset
FEI
FEE
FBI
FBE
BLPI
BLPE
PBE
PEE
Entered from any state when the MCU enters Stop mode
Stop
Returns to the state that was active before the MCU entered Stop mode, unless a reset occurs while in Stop mode.

Figure 25-18. MCG Mode State Diagram

NOTE

- During exits from LLS or VLPS when the MCG is in PEE mode, the MCG will reset to PBE clock mode and the C1[CLKS] and S[CLKST] will automatically be set to 2’b10.
- If entering Normal Stop mode when the MCG is in PEE mode with C5[PLLSTEN]=0, the MCG will reset to PBE clock mode and C1[CLKS] and S[CLKST] will automatically be set to 2’b10.

25.4.1.1 MCG Modes of Operation

The MCG operates in one of the following modes.

Note

> The MCG restricts transitions between modes. For the permitted transitions, see Figure 25-18.

Table 25-22. MCG Modes of Operation

Mode	Description
FLL Engaged Internal (FEI)	FLL engaged internal (FEI) is the default mode of operation and is entered when all the following condtions occur: • C1[CLKS] bits are written to 00 • C1[IREFS] bit is written to 1 • C6[PLLS] bit is written to 0 In FEI mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the 32 kHz Internal Reference Clock (IRC). The FLL loop will lock the DCO frequency to the FLL factor, as selected by the C4[DRST_DRS] and C4[DMX32] bits, times the internal reference frequency . Refer to the C4[DMX32] bit description for more details. In FEI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.
FLL Engaged External (FEE)	FLL engaged external (FEE) mode is entered when all the following conditions occur: • C1[CLKS] bits are written to 00 • C1[IREFS] bit is written to 0 • C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz • C6[PLLS] bit is written to 0 In FEE mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the external reference clock. The FLL loop will lock the DCO frequency to the FLL factor, as selected by C4[DRST_DRS] and C4[DMX32] bits, times the external reference frequency, as specified by the C1[FRDIV] and C2[RANGE0]. Refer to the C4[DMX32] bit description for more details. In FEE mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.
FLL Bypassed Internal (FBI)	FLL bypassed internal (FBI) mode is entered when all the following conditions occur: • C1[CLKS] bits are written to 01 • C1[IREFS] bit is written to 1 • C6[PLLS] is written to 0 • C2[LP] is written to 0 In FBI mode, the MCGOUTCLK is derived either from the slow (32 kHz IRC) or fast (4 MHz IRC) internal reference clock, as selected by the C2[IRCS] bit. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the C2[IRCS] selected internal reference clock. The FLL clock (DCOCLK) is controlled by the slow internal reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by the C4[DRST_DRS] and C4[DMX32] bits, times the internal reference frequency. Refer to the C4[DMX32] bit description for more details. In FBI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.

Table continues on the next page...

Table 25-22. MCG Modes of Operation (continued)

Mode	Description
FLL Bypassed External (FBE)	FLL bypassed external (FBE) mode is entered when all the following conditions occur: • C1[CLKS] bits are written to 10 • C1[IREFS] bit is written to 0 • C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz. • C6[PLLS] bit is written to 0 • C2[LP] is written to 0 In FBE mode, the MCGOUTCLK is derived from the OSCSEL external reference clock. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the external reference clock. The FLL clock (DCOCLK) is controlled by the external reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by the C4[DRST_DRS] and C4[DMX32] bits, times the divided external reference frequency. Refer to the C4[DMX32] bit description for more details. In FBI mode the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.
PLL Engaged External (PEE)	PLL Engaged External (PEE) mode is entered when all the following conditions occur: • C1[CLKS] bits are written to 00 • C1[IREFS] bit is written to 0 • C6[PLLS] bit is written to 1 In PEE mode, the MCGOUTCLK is derived from either the output of PLL0 or PLL1 (depending on the C11[PLLCS] bit) which are controlled by a selectable PLL reference clock. The selected PLL clock frequency locks to a multiplication factor, as specified by its corresponding VDIV, times the selected PLL reference frequency, as specified by its corresponding PRDIV. The PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.
PLL Bypassed External (PBE)	PLL Bypassed External (PBE) mode is entered when all the following conditions occur: • C1[CLKS] bits are written to 10 • C1[IREFS] bit is written to 0 • C6[PLLS] bit is written to 1 • C2[LP] bit is written to 0 In PBE mode, MCGOUTCLK is derived from the OSCSEL external reference clock; the PLL selected by C11[PLLCS] is operational, but its output clock is not used. This mode is useful to allow the PLLCS selected PLL to acquire its target frequency while MCGOUTCLK is driven from the external reference clock. The C11[PLLCS] selected PLL clock frequency locks to a multiplication factor, as specified by its [VDIV], times the PLL reference frequency, as specified by its [PRDIV]. In preparation for transition to PEE, the PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.

Table continues on the next page...

Table 25-22. MCG Modes of Operation (continued)

Mode	Description
Bypassed Low Power Internal (BLPI)[1]	Bypassed Low Power Internal (BLPI) mode is entered when all the following conditions occur: • C1[CLKS] bits are written to 01 • C1[IREFS] bit is written to 1 • C6[PLLS] bit is written to 0 • C2[LP] bit is written to 1 In BLPI mode, MCGOUTCLK is derived from the internal reference clock. The FLL is disabled and PLL is disabled even if the C5[PLLCLKEN0] is set to 1.
Bypassed Low Power External (BLPE)	Bypassed Low Power External (BLPE) mode is entered when all the following conditions occur: • C1[CLKS] bits are written to 10 • C1[IREFS] bit is written to 0 • C2[LP] bit is written to 1 In BLPE mode, MCGOUTCLK is derived from the OSCSEL external reference clock. The FLL is disabled and PLL is disabled even if the C5[PLLCLKEN0] is set to 1.
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 MCG behavior during Stop recovery. Entering Stop mode, the FLL is disabled, and all MCG clock signals are static except in the following case: MCGPLLCLK is active in Normal Stop mode when PLLSTEN=1 MCGIRCLK is active in Stop mode when all the following conditions become true: • C1[IRCLKEN] = 1 • C1[IREFSTEN] = 1 **NOTE:** • When entering Low Power Stop modes (LLS or VLPS) from PEE mode, on exit the MCG clock mode is forced to PBE clock mode, the C1[CLKS] and S[CLKST] will be configured to 2'b10 and S[LOCK0] bit will be cleared without setting S[LOLS0]. • When entering Normal Stop mode from PEE mode and if C5[PLLSTEN0]=0, on exit the MCG clock mode is forced to PBE mode, the C1[CLKS] and S[CLKST] will be configured to 2'b10 and S[LOCK0] bit will clear without setting S[LOLS0]. If C5[PLLSTEN0]=1, the S[LOCK0] bit will not get cleared and on exit the MCG will continue to run in PEE mode.

[1] If entering VLPR mode, MCG has to be configured and enter BLPE mode or BLPI mode with the 4 MHz IRC clock selected (C2[IRCS]=1). Once in VLPR mode, writes to any of the MCG control registers that can cause a MCG clock mode switch to a non low power clock mode must be avoided.

NOTE

For the chip-specific modes of operation, refer to the power management chapter of this MCU.

25.4.1.2 MCG Mode Switching

The C1[IREFS] bit can be changed at any time, but the actual switch to the newly selected reference clocks is shown by the S[IREFST] bit. When switching between engaged internal and engaged external modes, the FLL will begin locking again after the switch is completed.

The C1[CLKS] bits can also be changed at anytime, but the actual switch to the newly selected clock is shown by the S[CLKST] bits. If the newly selected clock is not available, the previous clock will remain selected.

The C4[DRST_DRS] write bits can be changed at anytime except when C2[LP] bit is 1. If the C4[DRST_DRS] write bits are changed while in FLL engaged internal (FEI) or FLL engaged external (FEE), the MCGOUTCLK will switch to the new selected DCO range within three clocks of the selected DCO clock. After switching to the new DCO, the FLL remains unlocked for several reference cycles. DCO startup time is equal to the FLL acquisition time. After the selected DCO startup time is over, the FLL is locked. The completion of the switch is shown by the C4[DRST_DRS] read bits.

25.4.2 Low Power Bit Usage

The C2[LP] bit is provided to allow the FLL or PLL to be disabled and thus conserve power when these systems are not being used. The C4[DRST_DRS] can not be written while C2[LP] bit is 1. However, in some applications, it may be desirable to enable the FLL or PLL and allow it to lock for maximum accuracy before switching to an engaged mode. Do this by writing C2[LP] to 0.

25.4.3 MCG Internal Reference Clocks

This module supports two internal reference clocks with nominal frequencies of 32 kHz (slow IRC) and 4 MHz (fast IRC). The fast IRC frequency can be divided down by programming of the FCRDIV to produce a frequency range of 32 kHz to 4 MHz.

25.4.3.1 MCG Internal Reference Clock

The MCG Internal Reference Clock (MCGIRCLK) provides a clock source for other on-chip peripherals and is enabled when C1[IRCLKEN]=1. When enabled, MCGIRCLK is driven by either the fast internal reference clock (4 MHz IRC which can be divided down by the FRDIV factors) or the slow internal reference clock (32 kHz IRC). The IRCS clock frequency can be re-targeted by trimming the period of its IRCS selected internal reference clock. This can be done by writing a new trim value to the C3[SCTRIM]:C4[SCFTRIM] bits when the slow IRC clock is selected or by writing a new trim value to the C4[FCTRIM] bits when the fast IRC clock is selected. The internal reference clock period is proportional to the trim value written. C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) and C4[FCTRIM] (if C2[IRCS]=1) bits affect the MCGOUTCLK frequency if the MCG is in FBI or BLPI modes. C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) bits also affect the MCGOUTCLK frequency if the MCG is in FEI mode.

Additionally, this clock can be enabled in Stop mode by setting C1[IRCLKEN] and C1[IREFSTEN], otherwise this clock is disabled in Stop mode.

25.4.4 External Reference Clock

The MCG module can support an external reference clock in all modes. Refer to the device datasheet for external reference frequency range. When C1[IREFS] is set, the external reference clock will not be used by the FLL or PLL. In these modes, the frequency can be equal to the maximum frequency the chip-level timing specifications will support.

If any of the CME bits are asserted the slow internal reference clock is enabled along with the enabled external clock monitor. For the case when C6[CME0]=1, a loss of clock is detected if the OSC0 external reference falls below a minimum frequency (f_{loc_high} or f_{loc_low} depending on C2[RANGE0]). For the case when C12[CME2]=1, a loss of clock is detected if the OSC1 external reference falls below a minimum frequency (f_{loc_high} or f_{loc_low} depending on C10[RANGE1]).For the case when C8[CME1]=1, a loss of clock is detected if the RTC external reference falls below a minimum frequency (f_{loc_low}).

Upon detect of a loss of clock event, the MCU generates a system reset if the respective LOCRE bit is set. Otherwise the MCG sets the respective LOCS bit and the MCG generates a LOCS interrupt request. In the case where a OSC0 or OSC1 loss of clock is detected the respective PLL's LOCK status bit is cleared if the OSC clock that is lost was selected as the PLL reference clock.

25.4.5 MCG Fixed Frequency Clock

The MCG Fixed Frequency Clock (MCGFFCLK) provides a fixed frequency clock source for other on-chip peripherals. This clock is driven by either the slow clock from the internal reference clock generator or the external reference clock from the Crystal Oscillator, divided by the FLL reference clock divider. The source of MCGFFCLK is selected by C1[IREFS]. Additionally, this clock is divided by two.

This clock is synchronized to the peripheral bus clock and is only valid when it's frequency is not more than 1/8 of the MCGOUTCLK frequency. When it is not valid, it is disabled and held high. The MCGFFCLK is not available when the MCG is in BLPI mode. This clock is also disabled in Stop mode. The FLL reference clock must be set within the valid frequency range for the MCGFFCLK.

25.4.6 MCG PLL Clock

The MCG PLL Clocks (MCGPLL0CLK, MCGPLL0CLK2X, MCGDDRCLK, MCGPLL1CLK, MCGPLL1CLK2X) are available depending on the device's configuration of the MCG module and DDR interface. For more details, refer to the clock distribution chapter of this MCU. The MCGPLLCLK is prevented from coming out of the MCG until it is enabled and S[LOCK0] is set.

25.4.7 MCG Auto TRIM (ATM)

The MCG Auto Trim (ATM) is a MCG feature that when enabled, it configures the MCG hardware to automatically trim the MCG Internal Reference Clocks using an external clock as a reference. The selection between which MCG IRC clock gets tested and enabled is controlled by the ATC[ATMS] control bit (ATC[ATMS]=0 selects the 32 kHz IRC and ATC[ATMS]=1 selects the 4 MHz IRC). If 4 MHz IRC is selected for the ATM, a divide by 128 is enabled to divide down the 4 MHz IRC to a range of 31.250 kHz.

When MCG ATM is enabled by writing ATC[ATME] bit to 1, The ATM machine will start auto trimming the selected IRC clock. During the autotrim process, ATC[ATME] will remain asserted and will deassert after ATM is completed or an abort occurs. The MCG ATM is aborted if a write to any of the following control registers is detected including: C1, C3, C4, or ATC or if Stop mode is entered. If an abort occurs, ATC[ATMF] fail flag is asserted.

The ATM machine uses the bus clock as the external reference clock to perform the IRC auto-trim. Therefore, it is required that the MCG is configured in a clock mode where the reference clock used to generate the system clock is the external reference clock such as FBE clock mode. The MCG must not be configured in a clock mode where selected IRC ATM clock is used to generate the system clock. The bus clock is also required to be running with in the range of 8 - 16 MHz.

To perform the ATM on the selected IRC, the ATM machine uses the successive approximation technique to adjust the IRC trim bits to generate the desired IRC trimmed frequency. The ATM SARs each of the ATM IRC trim bits starting with the MSB. For each trim bit test, the ATM uses a pulse that is generated by the ATM selected IRC clock to enable a counter that counts number of ATM external clocks. At end of each trim bit, the ATM external counter value is compared to the ATCV[15:0] register value. Based on the comparison result, the ATM trim bit under test will get cleared or stay asserted. This is done until all trim bits have been tested by ATM SAR machine.

Before the ATM can be enabled, the ATM expected count needs to get derived and stored into the ATCV register. The ATCV expected count is derived based on the required target Internal Reference Clock (IRC) frequency, the frequency of the external reference clock, and using the following formula:

$$\text{ATCV Expected Count Value} = 21 * (Fe / Fr)$$

- Fr = Target Internal Reference Clock (IRC) Trimmed Frequency
- Fe = External Clock Frequency

If the auto trim is being performed on the 4 MHz IRC, the calculated expected count value must be multiplied by 128 before storing it in the ATCV register. Therefore, the ATCV Expected Count Value for trimming the 4 MHz IRC is calculated using the following formula.

$$\text{Expected Count Value} = (Fe / Fr) * 21 * (128)$$

25.5 Initialization / Application Information

This section describes how to initialize and configure the MCG module in an application. The following sections include examples on how to initialize the MCG and properly switch between the various available modes.

25.5.1 MCG Module Initialization Sequence

The MCG comes out of reset configured for FEI mode. The internal reference will stabilize in $t_{irefsts}$ microseconds before the FLL can acquire lock. As soon as the internal reference is stable, the FLL will acquire lock in $t_{fll\_acquire}$ milliseconds.

25.5.1.1 Initializing the MCG

Because the MCG comes out of reset in FEI mode, the only MCG modes that can be directly switched to upon reset are FEE, FBE, and FBI modes (see Figure 25-18). Reaching any of the other modes requires first configuring the MCG for one of these three intermediate modes. Care must be taken to check relevant status bits in the MCG status register reflecting all configuration changes within each mode.

To change from FEI mode to FEE or FBE modes, follow this procedure:

1. Enable the external clock source by setting the appropriate bits in C2 register.
2. Write to C1 register to select the clock mode.
    - If entering FEE mode, set C1[FRDIV] appropriately, clear the C1[IREFS] bit to switch to the external reference, and leave the C1[CLKS] bits at 2'b00 so that the output of the FLL is selected as the system clock source.
    - If entering FBE, clear the C1[IREFS] bit to switch to the external reference and change the C1[CLKS] bits to 2'b10 so that the external reference clock is selected as the system clock source. The C1[FRDIV] bits should also be set appropriately here according to the external reference frequency to keep the FLL reference clock in the range of 31.25 kHz to 39.0625 kHz. Although the FLL is bypassed, it is still on in FBE mode.
    - The internal reference can optionally be kept running by setting the C1[IRCLKEN] bit. This is useful if the application will switch back and forth between internal and external modes. For minimum power consumption, leave the internal reference disabled while in an external clock mode.
3. Once the proper configuration bits have been set, wait for the affected bits in the MCG status register to be changed appropriately, reflecting that the MCG has moved into the proper mode.
    - If the MCG is in FEE, FBE, PEE, PBE, or BLPE mode, and C2[EREFS0] was also set in step 1, wait here for S[OSCINIT0] bit to become set indicating that the external clock source has finished its initialization cycles and stabilized.
    - If in FEE mode, check to make sure the S[IREFST] bit is cleared before moving on.
    - If in FBE mode, check to make sure the S[IREFST] bit is cleared and S[CLKST] bits have changed to 2'b10 indicating the external reference clock has been appropriately selected. Although the FLL is bypassed, it is still on in FBE mode.
4. Write to the C4 register to determine the DCO output (MCGFLLCLK) frequency range.

- By default, with C4[DMX32] cleared to 0, the FLL multiplier for the DCO output is 640. For greater flexibility, if a mid-low-range FLL multiplier of 1280 is desired instead, set C4[DRST_DRS] bits to 2'b01 for a DCO output frequency of 40 MHz. If a mid high-range FLL multiplier of 1920 is desired instead, set the C4[DRST_DRS] bits to 2'b10 for a DCO output frequency of 60 MHz. If a high-range FLL multiplier of 2560 is desired instead, set the C4[DRST_DRS] bits to 2'b11 for a DCO output frequency of 80 MHz.
- When using a 32.768 kHz external reference, if the maximum low-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b00 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 732 will be 24 MHz.
- When using a 32.768 kHz external reference, if the maximum mid-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b01 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 1464 will be 48 MHz.
- When using a 32.768 kHz external reference, if the maximum mid high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b10 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2197 will be 72 MHz.
- When using a 32.768 kHz external reference, if the maximum high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST_DRS] bits to 2'b11 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2929 will be 96 MHz.

5. Wait for the FLL lock time to guarantee FLL is running at new C4[DRST_DRS] and C4[DMX32] programmed frequency.

To change from FEI clock mode to FBI clock mode, follow this procedure:

1. Change C1[CLKS] bits in C1 register to 2'b01 so that the internal reference clock is selected as the system clock source.
2. Wait for S[CLKST] bits in the MCG status register to change to 2'b01, indicating that the internal reference clock has been appropriately selected.
3. Write to the C2 register to determine the IRCS output (IRCSCLK) frequency range.

- By default, with C2[IRCS] cleared to 0, the IRCS selected output clock is the slow internal reference clock (32 kHz IRC). If the faster IRC is desired, set C2[IRCS] bit to 1 for a IRCS clock derived from the 4 MHz IRC source.

25.5.2 Using a 32.768 kHz Reference

In FEE and FBE modes, if using a 32.768 kHz external reference, at the default FLL multiplication factor of 640, the DCO output (MCGFLLCLK) frequency is 20.97 MHz at low-range. If C4[DRST_DRS] bits are set to 2'b01, the multiplication factor is doubled to 1280, and the resulting DCO output frequency is 41.94 Mhz at mid-low-range. If C4[DRST_DRS] bits are set to 2'b10, the multiplication factor is set to 1920, and the resulting DCO output frequency is 62.91 MHz at mid high-range. If C4[DRST_DRS] bits are set to 2'b11, the multiplication factor is set to 2560, and the resulting DCO output frequency is 83.89 MHz at high-range.

In FBI and FEI modes, setting C4[DMX32] bit is not recommended. If the internal reference is trimmed to a frequency above 32.768 kHz, the greater FLL multiplication factor could potentially push the microcontroller system clock out of specification and damage the part.

25.5.3 MCG Mode Switching

When switching between operational modes of the MCG, certain configuration bits must be changed in order to properly move from one mode to another. Each time any of these bits are changed (C6[PLLS], C1[IREFS], C1[CLKS], C2[IRCS], or C2[EREFS0]), the corresponding bits in the MCG status register (PLLST, IREFST, CLKST, IRCST, or OSCINIT) must be checked before moving on in the application software.

Additionally, care must be taken to ensure that the reference clock divider (C1[FRDIV] and C5[PRDIV0]) is set properly for the mode being switched to. For instance, in PEE mode, if using a 16 MHz crystal, C5[PRDIV0] must be set to 3'b000 (divide-by-1) or 3'b001 (divide -by-2) in order to divide the external reference down to the required frequency between 8 and 16 MHz.

In FBE, FEE, FBI, and FEI modes, at any time, the application can switch the FLL multiplication factor between 640, 1280, 1920, and 2560 with C4[DRST_DRS] bits. Writes to C4[DRST_DRS] bits will be ignored if C2[LP]=1.

The table below shows MCGOUTCLK frequency calculations using C1[FRDIV], C5[PRDIV0], and C6[VDIV0] settings for each clock mode.

Table 25-23. MCGOUTCLK Frequency Calculation Options

Clock Mode	$f_{MCGOUTCLK}$[1]	Note
FEI (FLL engaged internal)	(f_{int} * F)	Typical $f_{MCGOUTCLK}$ = 20 MHz immediately after reset.
FEE (FLL engaged external)	(f_{ext} / FLL_R) *F	f_{ext} / FLL_R must be in the range of 31.25 kHz to 39.0625 kHz
FBE (FLL bypassed external)	f_{ext}	f_{ext} / FLL_R must be in the range of 31.25 kHz to 39.0625 kHz
FBI (FLL bypassed internal)	f_{int}	Typical f_{int} = 32 kHz
PEE (PLL engaged external)	(f_{ext} / PLL_R) * M	f_{ext} / PLL_R must be in the range of 8 – 16 MHz
PBE (PLL bypassed external)	f_{ext}	f_{ext} / PLL_R must be in the range of 8 – 16 MHz
BLPI (Bypassed low power internal)	f_{int}	
BLPE (Bypassed low power external)	f_{ext}	

1. FLL_R is the reference divider selected by the C1[FRDIV] bits, PLL_R is the reference divider selected by C5[PRDIV0] bits, F is the FLL factor selected by C4[DRST_DRS] and C4[DMX32] bits, and M is the multiplier selected by C6[VDIV0] bits.

This section will include 3 mode switching examples using an 16 MHz external crystal. If using an external clock source less than 8 MHz, the MCG should not be configured for any of the PLL modes (PEE and PBE).

25.5.3.1 Example 1: Moving from FEI to PEE Mode with OSC0 as the source for the external crystal clock: External Crystal = 16 MHz, MCGOUTCLK Frequency = 128 MHz

In this example, the MCG will move through the proper operational modes from FEI to PEE to achieve 128 MHz MCGOUTCLK frequency from 16 MHz external crystal reference. First, the code sequence will be described. Then a flowchart will be included which illustrates the sequence.

1. First, FEI must transition to FBE mode:

   a. C2 = 0x1C

      - C2[RANGE0] set to 2'b10 because the frequency of 16 MHz is within the high frequency range
      - C2[HGO0] set to 1 to configure the crystal oscillator for high gain operation
      - C2[EREFS0] set to 1, because a crystal is being used

   b. C1 = 0xA0

- C1[CLKS] set to 2'b10 in order to select external reference clock as system clock source
- C1[FRDIV] set to 3'b100, or divide-by-512 because 8 MHz / 512 = 31.25 kHz which is in the 31.25 kHz to 39.0625 kHz range required by the FLL
- C1[IREFS] cleared to 0, selecting the external reference clock and enabling the external oscillator.

c. Loop until S[OSCINIT0] is 1, indicating the crystal selected by C2[EREFS0] has been initialized..

d. Loop until S[IREFST] is 0, indicating the external reference is the current source for the reference clock

e. Loop until S[CLKST] is 2'b10, indicating that the external reference clock is selected to feed MCGOUTCLK

2. Then configure C5[PRDIV0] to generate correct PLL reference frequency.

   a. C5 = 0x01

      - C5[PRDIV0] set to 3'b001, or divide-by-2 resulting in a pll reference frequency of 16 MHz/2 = 8 MHz.
      - C5[PLLREFSEL0] set to 1'b0 to select the external reference clock from OSC0 as the reference clock to the PLL.

3. Then, FBE must transition either directly to PBE mode or first through BLPE mode and then to PBE mode:

   a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1.

   b. BLPE/PBE: C6 = 0x40

      - C6[PLLS] set to 1, selects the PLL. At this time, with a C1[PRDIV] value of 2'b001, the PLL reference divider is 2 (see PLL External Reference Divide Factor table), resulting in a reference frequency of 16 MHz/ 2 = 8 MHz. In BLPE mode,changing the C6[PLLS] bit only prepares the MCG for PLL usage in PBE mode.
      - C6[VDIV0] set to 5'b0000, or multiply-by-16 because 8 MHz reference * 16 = 128 MHz. In BLPE mode, the configuration of the VDIV bits does not matter because the PLL is disabled. Changing them only sets up the multiply value for PLL usage in PBE mode.

   c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to PBE mode.

- d. PBE: Loop until S[PLLST] is set, indicating that the current source for the PLLS clock is the PLL.
- e. PBE: Then loop until S[LOCK0] is set, indicating that the PLL has acquired lock.

4. Lastly, PBE mode transitions into PEE mode:
   - a. C1 = 0x20
     - C1[CLKS] set to 2'b00 in order to select the output of the PLL as the system clock source.
   - b. Loop until S[CLKST] are 2'b11, indicating that the PLL output is selected to feed MCGOUTCLK in the current clock mode.
     - Now, With PRDIV0 of divide-by-2, and C6[VDIV0] of multiply-by-16, MCGOUTCLK = [(16 MHz / 2) * 16] = 128 MHz.

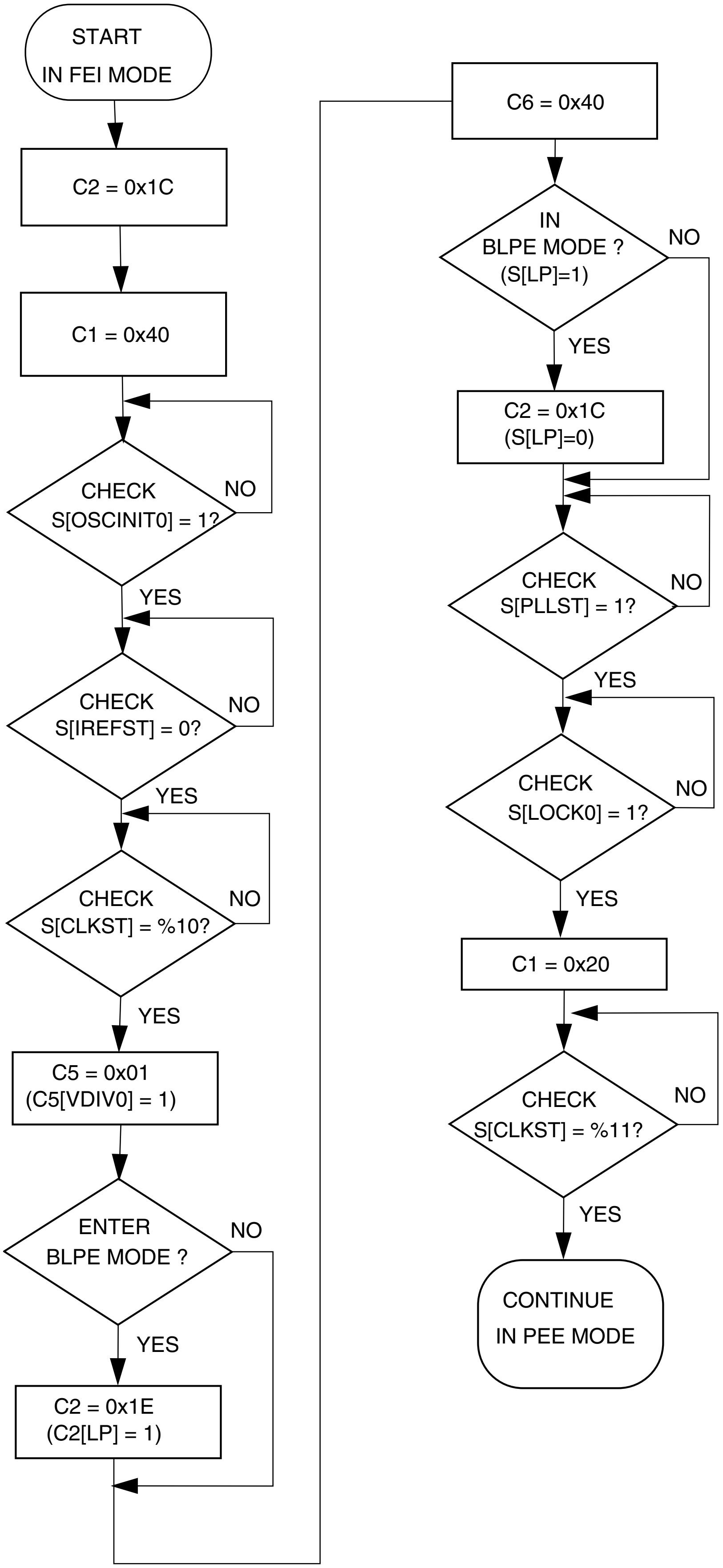


Figure 25-19. Flowchart of FEI to PEE Mode Transition using an 16 MHz crystal

25.5.3.2 Example 2: Moving from PEE to BLPI Mode: MCGOUTCLK Frequency =32 kHz

In this example, the MCG will move through the proper operational modes from PEE mode with a 16 MHz crystal configured for a 128 MHz MCGOUTCLK frequency (see previous example) to BLPI mode with a 32 kHz MCGOUTCLK frequency.First, the code sequence will be described. Then a flowchart will be included which illustrates the sequence.

1. First, PEE must transition to PBE mode:
    a. C1 = 0x40
        - C1[CLKS] set to 2'b10 in order to switch the system clock source to the external reference clock.

    b. Loop until S[CLKST] are 2'b10, indicating that the external reference clock is selected to feed MCGOUTCLK.
2. Then, PBE must transition either directly to FBE mode or first through BLPE mode and then to FBE mode:
    a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1

    b. BLPE/FBE: C6 = 0x00
        - C6[PLLS] clear to 0 to select the FLL. At this time, with C1[FRDIV] value of 3'b100, the FLL divider is set to 512, resulting in a reference frequency of 16 MHz / 512 = 31.25 kHz. If C1[FRDIV] was not previously set to 3'b100 (necessary to achieve required 31.25-39.06 kHz FLL reference frequency with an 16 MHz external source frequency), it must be changed prior to clearing C6[PLLS] bit. In BLPE mode,changing this bit only prepares the MCG for FLL usage in FBE mode. With C6[PLLS] = 0, the C6[VDIV0] value does not matter.

    c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to FBE mode.

    d. FBE: Loop until S[PLLST] is cleared, indicating that the current source for the PLLS clock is the FLL.
3. Next, FBE mode transitions into FBI mode:
    a. C1 = 0x64

- C1[CLKS] set to 2'b01 in order to switch the system clock to the internal reference clock.
- C1[IREFS] set to 1 to select the internal reference clock as the reference clock source.
- C1[FRDIV] remain unchanged because the reference divider does not affect the internal reference.

b. Loop until S[IREFST] is 1, indicating the internal reference clock has been selected as the reference clock source.

c. Loop until S[CLKST] are 2'b01, indicating that the internal reference clock is selected to feed MCGOUTCLK.

4. Lastly, FBI transitions into BLPI mode.

a. C2 = 0x22

- C2[LP] is 1
- C2[RANGE0], C2[HGO0], C2[EREFS0], C1[IRCLKEN], and C1[IREFSTEN] bits are ignored when the C1[IREFS] bit is set. They can remain set, or be cleared at this point.

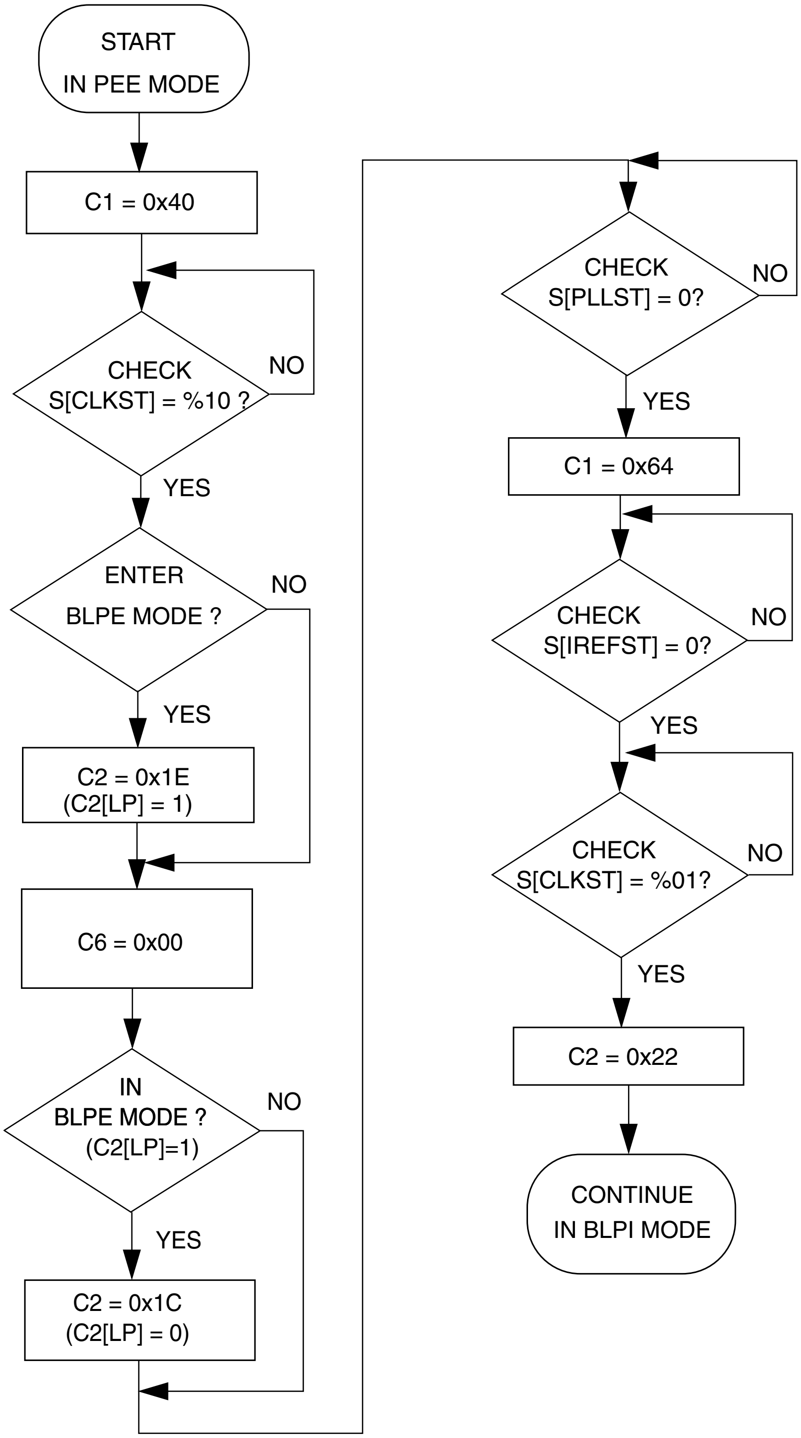


Figure 25-20. Flowchart of PEE to BLPI Mode Transition using an 16 MHz crystal

25.5.3.3 Example 3: Moving from BLPI to FEE Mode

In this example, the MCG will move through the proper operational modes from BLPI mode at a 32 kHz MCGOUTCLK frequency running off the internal reference clock (see previous example) to FEE mode using a 16 MHz crystal configured for a 20 MHz MCGOUTCLK frequency. First, the code sequence will be described. Then a flowchart will be included which illustrates the sequence.

1. First, BLPI must transition to FBI mode.
   a. C2 = 0x00
      - C2[LP] is 0
2. Next, FBI will transition to FEE mode.
   a. C2 = 0x2C
      - C2[RANGE0] set to 2'b10 because the frequency of 16 MHz is within the high frequency range.
      - C2[HGO0] set to 1 to configure the crystal oscillator for high gain operation.
      - C2[EREFS0] set to 1, because a crystal is being used.

   b. C1 = 0x20
      - C1[CLKS] set to 2'b00 in order to select the output of the FLL as system clock source.
      - C1[FRDIV] remain at 3'b100, or divide-by-512 for a reference of 16 MHz / 512 = 31.25 kHz.
      - C1[IREFS] cleared to 0, selecting the external reference clock.

   c. Loop until S[OSCINIT0] is 1, indicating the crystal selected by the C2[EREFS0] bit has been initialized.

   d. Loop until S[IREFST] is 0, indicating the external reference clock is the current source for the reference clock.

   e. Loop until S[CLKST] are 2'b00, indicating that the output of the FLL is selected to feed MCGOUTCLK.

   f. Now, with a 31.25 kHz reference frequency, a fixed DCO multiplier of 640, MCGOUTCLK = 31.25 kHz * 640 / 1 = 20 MHz.

   g. At this point, by default, the C4[DRST_DRS] bits are set to 2'b00 and C4[DMX32] is cleared to 0. If the MCGOUTCLK frequency of 40 MHz is desired instead, set the C4[DRST_DRS] bits to 0x01 to switch the FLL multiplication factor from 640 to 1280. To return the MCGOUTCLK frequency to 20 MHz, set C4[DRST_DRS] bits to 2'b00 again, and the FLL multiplication factor will switch back to 640.

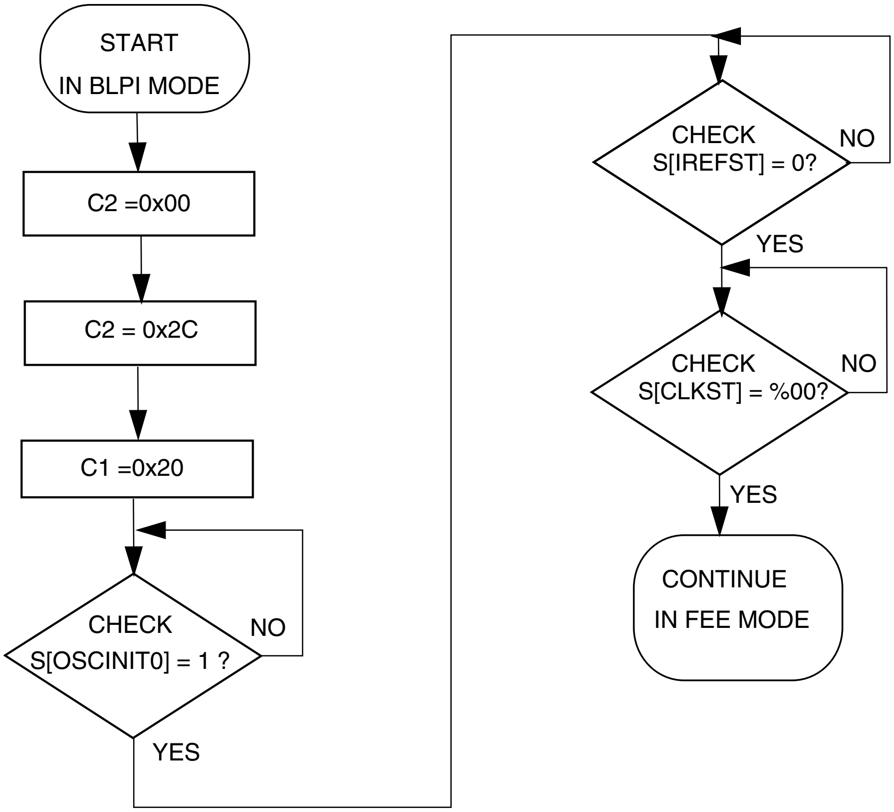


Figure 25-21. Flowchart of BLPI to FEE Mode Transition using an 16 MHz crystal

Chapter 26
Oscillator (OSC)

26.1 Introduction

NOTE

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

The OSC module is a crystal oscillator. The module, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

26.2 Features and Modes

Key features of the module are:

- Supports 32 kHz crystals (Low Range mode)
- Supports 3–8 MHz, 8–32 MHz crystals and resonators (High Range mode)
- Automatic Gain Control (AGC) to optimize power consumption in high frequency ranges 3–8 MHz, 8–32 MHz using low-power mode
- High gain option in frequency ranges: 32 kHz, 3–8 MHz, and 8–32 MHz
- Voltage and frequency filtering to guarantee clock frequency and stability
- Optionally external input bypass clock from EXTAL signal directly
- One clock for MCU clock system
- Two clocks for on-chip peripherals that can work in Stop modes

Functional Description describes the module's operation in more detail.

26.3 Block Diagram

The OSC module uses a crystal or resonator to generate three filtered oscillator clock signals. Three clocks are output from OSC module: OSCCLK for MCU system, OSCERCLK for on-chip peripherals, and OSC32KCLK. The OSCCLK can only work in run mode. OSCERCLK and OSC32KCLK can work in low power modes. For the clock source assignments, refer to the clock distribution information of this MCU.

Refer to the chip configuration chapter for the external reference clock source in this MCU.

The following figure shows the block diagram of the OSC module.

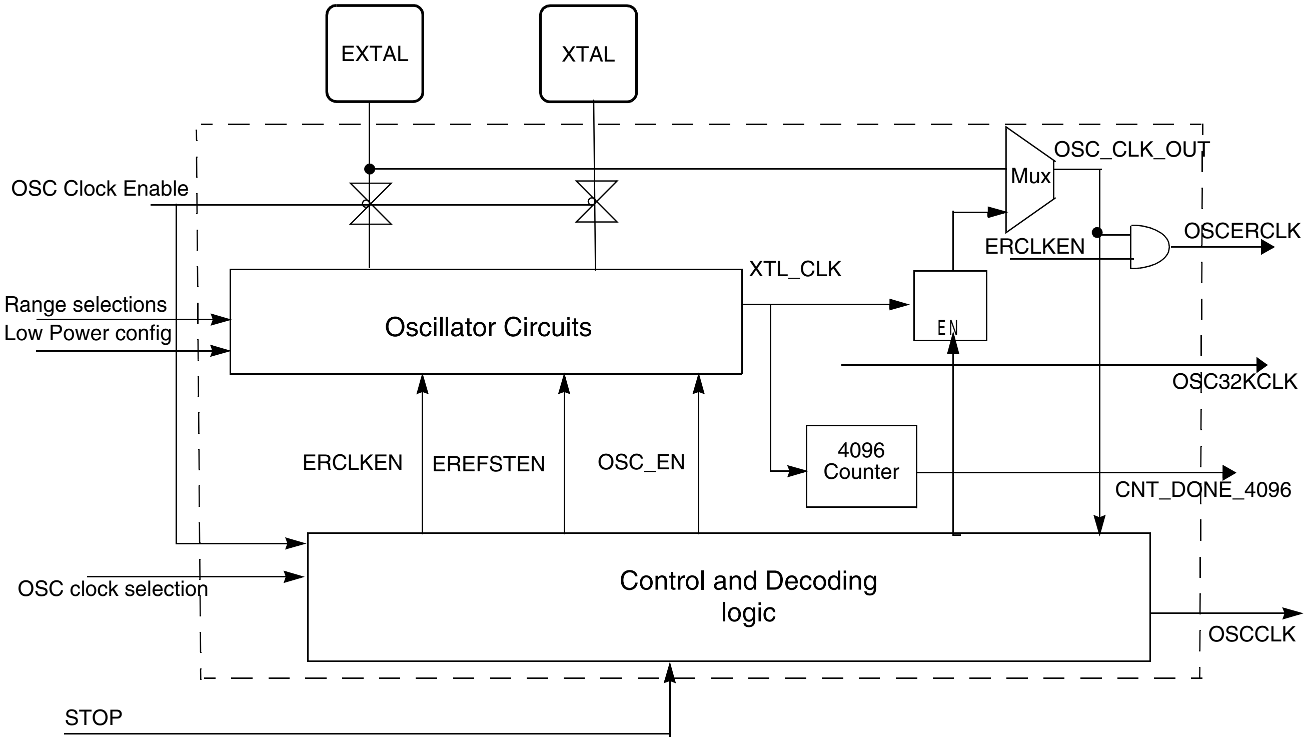


Figure 26-1. OSC Module Block Diagram

26.4 OSC Signal Descriptions

The following table shows the user-accessible signals available for the OSC module. Refer to signal multiplexing information for this MCU for more details.

Table 26-1. OSC Signal Descriptions

Signal	Description	I/O
EXTAL	External clock/Oscillator input	I
XTAL	Oscillator output	O

26.5 External Crystal / Resonator Connections

The connections for a crystal/resonator frequency reference are shown in the following figures. When using low-frequency, low-power mode, the only external component is the crystal or ceramic resonator itself. In the other oscillator modes, load capacitors (C_x, C_y) and feedback resistor (R_F) are required. The following table shows all possible connections.

Table 26-2. External Caystal/Resonator Connections

Oscillator Mode	Connections
Low-frequency (32 kHz), low-power	Connection 1
Low-frequency (32 kHz), high-gain	Connection 2/Connection 3[1]
High-frequency (3~32 MHz), low-power	Connection 1/Connection 3[2,2]
High-frequency (3~32 MHz), high-gain	Connection 2/Connection 3[2]

1. When the load capacitors (Cx, Cy) are greater than 30 pF, use Connection 3.
2. With the low-power mode, the oscillator has the internal feedback resistor R_F. Therefore, the feedback resistor must not be externally with the Connection 3.

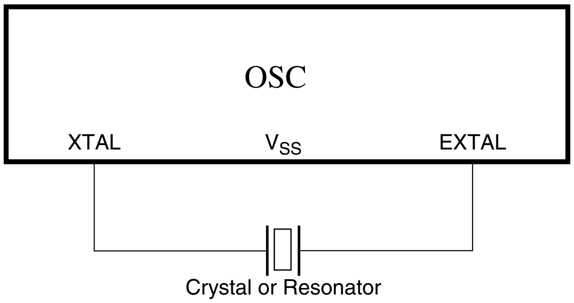


Figure 26-2. Crystal/Ceramic Resonator Connections - Connection 1

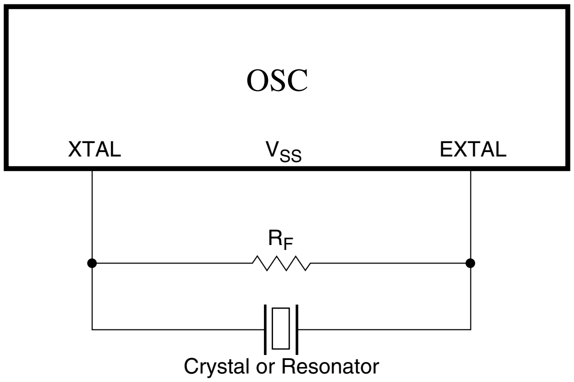


Figure 26-3. Crystal/Ceramic Resonator Connections - Connection 2

NOTE

Connection 1 and Connection 2 should use internal capacitors as the load of the oscillator by configuring the CR[SCxP] bits.

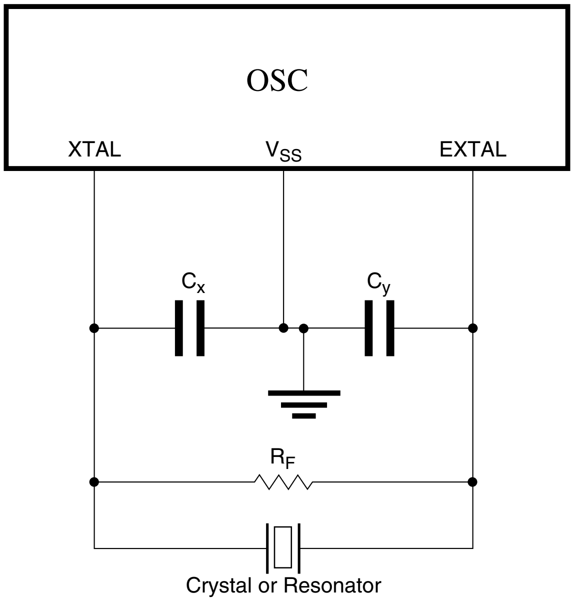


Figure 26-4. Crystal/Ceramic Resonator Connections - Connection 3

26.6 External Clock Connections

In external clock mode, the pins can be connected as shown below.

NOTE

XTAL can be used as a GPIO when the GPIO alternate function is configured for it.

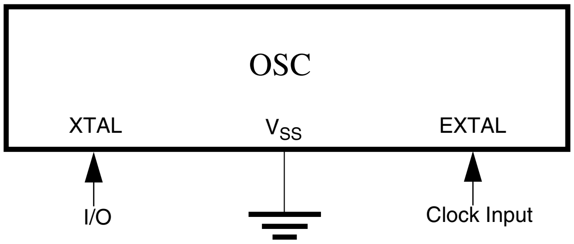


Figure 26-5. External Clock Connections

26.7 Memory Map/Register Definitions

Some oscillator module register bits are typically incorporated into other peripherals such as MCG or SIM.

26.7.1 OSC Memory Map/Register Definition

OSC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_5000	OSC Control Register (OSC0_CR)	8	R/W	00h	26.71.1/ 653
400E_5000	OSC Control Register (OSC1_CR)	8	R/W	00h	26.71.1/ 653

26.71.1 OSC Control Register (OSCx_CR)

NOTE

After OSC is enabled and starts generating the clocks, the configurations such as low power and frequency range, must not be changed.

Addresses: OSC0_CR is 4006_5000h base + 0h offset = 4006_5000h
OSC1_CR is 400E_5000h base + 0h offset = 400E_5000h

Bit	7	6	5	4	3	2	1	0
Read	ERCLKEN	0	EREFSTEN	0	SC2P	SC4P	SC8P	SC16P
Write		▨		▨				
Reset	0	0	0	0	0	0	0	0

OSCx_CR field descriptions

Field	Description
7 ERCLKEN	External Reference Enable Enables external reference clock (OSCERCLK). 0 External reference clock is inactive. 1 External reference clock is enabled.
6 Reserved	This read-only field is reserved and always has the value zero.
5 EREFSTEN	External Reference Stop Enable Controls whether or not the external reference clock (OSCERCLK) remains enabled when MCU enters Stop mode. 0 External reference clock is disabled in Stop mode. 1 External reference clock stays enabled in Stop mode if ERCLKEN is set before entering Stop mode.
4 Reserved	This read-only field is reserved and always has the value zero.
3 SC2P	Oscillator 2 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 2 pF capacitor to the oscillator load.
2 SC4P	Oscillator 4 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 4 pF capacitor to the oscillator load.
1 SC8P	Oscillator 8 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 8 pF capacitor to the oscillator load.
0 SC16P	Oscillator 16 pF Capacitor Load Configure Configures the oscillator load. 0 Disable the selection. 1 Add 16 pF capacitor to the oscillator load.

26.8 Functional Description

This following sections provide functional details of the module.

26.8.1 OSC Module States

The states of the OSC module are shown in the following figure. The states and their transitions between each other are described in this section.

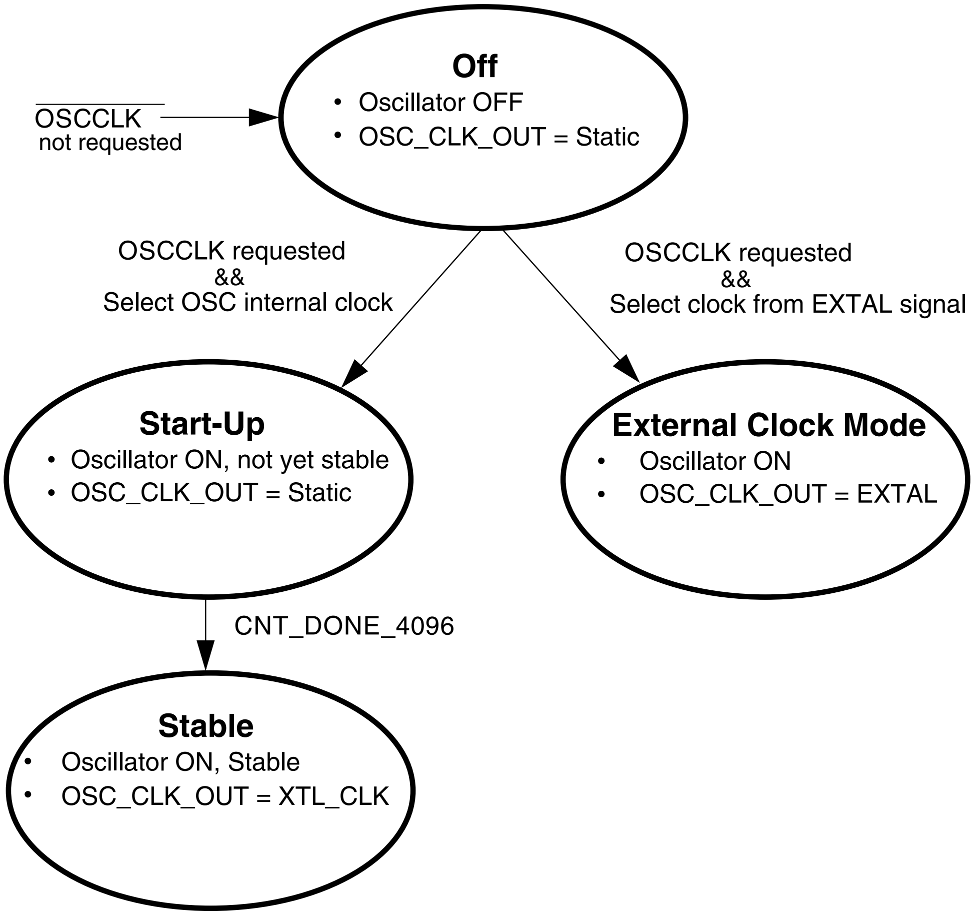


Figure 26-9. OSC Module State Diagram

NOTE

XTL_CLK is the clock generated internally from OSC circuits.

26.8.1.1 Off

The OSC enters the Off state when the system does not require OSC clocks. Upon entering this state, XTL_CLK is static unless OSC is configured to select the clock from the EXTAL pad by clearing the external reference clock selection bit. For details regarding the external reference clock source in this MCU, refer to the chip configuration chapter. The EXTAL and XTAL pins are also decoupled from all other oscillator circuitry in this state. The OSC module circuitry is configured to draw minimal current.

26.8.1.2 Oscillator Start-Up

The OSC enters start-up state when it is configured to generate clocks (internally the OSC_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit. In this state, the OSC module is enabled and oscillations are starting up, but have not yet stabilized. When the oscillation amplitude becomes large enough to pass through the input buffer, XTL_CLK begins clocking the counter. When the counter reaches 4096 cycles of XTL_CLK, the oscillator is considered stable and XTL_CLK is passed to the output clock OSC_CLK_OUT.

26.8.1.3 Oscillator Stable

The OSC enters stable state when it is configured to generate clocks (internally the OSC_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit and the counter reaches 4096 cycles of XTL_CLK (when CNT_DONE_4096 is high). In this state, the OSC module is producing a stable output clock on OSC_CLK_OUT. Its frequency is determined by the external components being used.

26.8.1.4 External Clock Mode

The OSC enters external clock state when it is enabled and external reference clock selection bit is cleared. For details regarding external reference clock source in this MCU, refer to the chip configuration chapter. In this state, the OSC module is set to buffer (with hysteresis) a clock from EXTAL onto the OSC_CLK_OUT. Its frequency is determined by the external clock being supplied.

26.8.2 OSC Module Modes

The OSC is a Pierce-type oscillator that supports external crystals or resonators operating over the frequency ranges shown in Table 26-9. These modes assume the following conditions: OSC is enabled to generate clocks (OSC_EN=1), configured to generate clocks internally (MCG_C2[EREFS] = 1), and some or one of the other peripherals (MCG, Timer, and so on) is configured to use the oscillator output clock (OSC_CLK_OUT).

Table 26-9. Oscillator Modes

Mode	Frequency Range
Low-frequency, high-gain	f_{osc_lo} (1 kHz) up to f_{osc_lo} (32.768 kHz)
Low-frequency, low-power (VLP)	
High-frequency mode1, high-gain	$f_{osc\_hi\_1}$ (3 MHz) up to $f_{osc\_hi\_1}$ (8 MHz)
High-frequency mode1, low-power	
High-frequency mode2, high-gain	$f_{osc\_hi\_2}$ (8 MHz) up to $f_{osc\_hi\_2}$ (32 MHz)
High-frequency mode2, low-power	

NOTE

For information about low power modes of operation used in this chip and their alignment with some OSC modes, refer to the chip's Power Management details.

26.8.2.1 Low-Frequency, High-Gain Mode

In Low-frequency, high-gain mode, the oscillator uses a simple inverter-style amplifier. The gain is set to achieve rail-to-rail oscillation amplitudes.

The oscillator input buffer in this mode is single-ended. It provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used.

26.8.2.2 Low-Frequency, Low-Power Mode

In low-frequency, low-power mode, the oscillator uses a gain control loop to minimize power consumption. As the oscillation amplitude increases, the amplifier current is reduced. This continues until a desired amplitude is achieved at steady-state. This mode provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used, the internal feedback resistor is connected, and no external resistor should be used.

In this mode, the amplifier inputs, gain-control input, and input buffer input are all capacitively coupled for leakage tolerance (not sensitive to the DC level of EXTAL).

Also in this mode, all external components except for the resonator itself are integrated, which includes the load capacitors and feeback resistor that biases EXTAL.

26.8.2.3 High-Frequency, High-Gain Mode

In high-frequency, high-gain mode, the oscillator uses a simple inverter-style amplifier. The gain is set to achieve rail-to-rail oscillation amplitudes. This mode provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used.

26.8.2.4 High-Frequency, Low-Power Mode

In high-frequency, low-power mode, the oscillator uses a gain control loop to minimize power consumption. As the oscillation amplitude increases, the amplifier current is reduced. This continues until a desired amplitude is achieved at steady-state. In this mode, the internal capacitors could be used, the internal feedback resistor is connected, and no external resistor should be used.

The oscillator input buffer in this mode is differential. It provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels.

26.8.3 Counter

The oscillator output clock (OSC_CLK_OUT) is gated off until the counter has detected 4096 cycles of its input clock (XTL_CLK). After 4096 cycles are completed, the counter passes XTL_CLK onto OSC_CLK_OUT. This counting time-out is used to guarantee output clock stability.

26.8.4 Reference Clock Pin Requirements

The OSC module requires use of both the EXTAL and XTAL pins to generate an output clock in Oscillator mode, but requires only the EXTAL pin in External clock mode. The EXTAL and XTAL pins are available for I/O. For the implementation of these pins on this device, refer to the Signal Multiplexing chapter.

26.9 Reset

There is no reset state associated with the OSC module. The counter logic is reset when the OSC is not configured to generate clocks.

There are no sources of reset requests for the OSC module.

26.10 Low Power Modes Operation

When the MCU enters Stop modes, the OSC is functional depending on ERCLKEN and EREFSETN bit settings. If both these bits are set, the OSC is in operation. In Low Leakage Stop (LLS) modes, the OSC holds all register settings. If ERCLKEN and EREFSTEN bits are set before entry to Low Leakage Stop modes, the OSC is still functional in these modes. After waking up from Very Low Leakage Stop (VLLSx) modes, all OSC register bits are reset and initialization is required through software.

26.11 Interrupts

The OSC module does not generate any interrupts.

Chapter 27
RTC Oscillator

27.1 Introduction

NOTE

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

The RTC oscillator module provides the clock source for the RTC. The RTC oscillator module, in conjunction with an external crystal, generates a reference clock for the RTC.

27.1.1 Features and Modes

The key features of the RTC oscillator are as follows:

- Supports 32 kHz crystals with very low power
- Consists of internal feed back resistor
- Consists of internal programmable capacitors as the Cload of the oscillator
- Automatic Gain Control (AGC) to optimize power consumption

The RTC oscillator operations are described in detail in Functional Description .

27.1.2 Block Diagram

The following is the block diagram of the RTC oscillator.

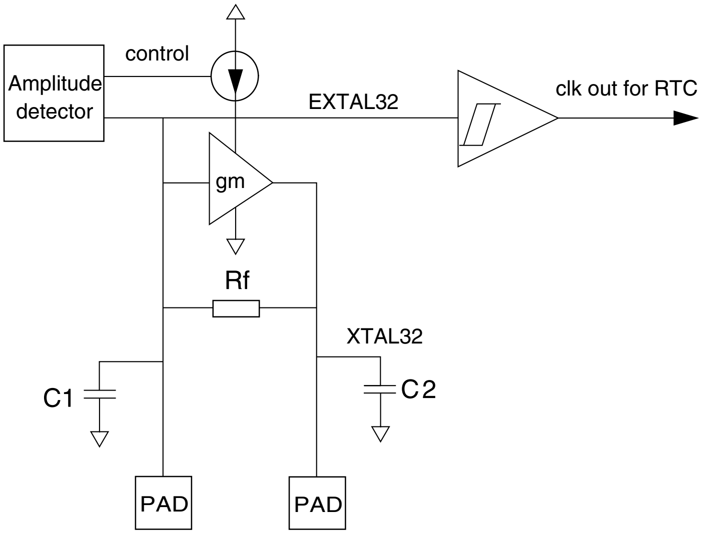


Figure 27-1. RTC Oscillator Block Diagram

27.2 RTC Signal Descriptions

The following table shows the user-accessible signals available for the RTC oscillator. See the chip-level specification to find out which signals are actually connected to the external pins.

Table 27-1. RTC Signal Descriptions

Signal	Description	I/O
EXTAL32	Oscillator Input	I
XTAL32	Oscillator Output	O

27.2.1 EXTAL32 — Oscillator Input

This signal is the analog input of the RTC oscillator.

27.2.2 XTAL32 — Oscillator Output

This signal is the analog output of the RTC oscillator module.

27.3 External Crystal Connections

The connections with a crystal is shown in the following figure. External load capacitors and feedback resistor are not required.

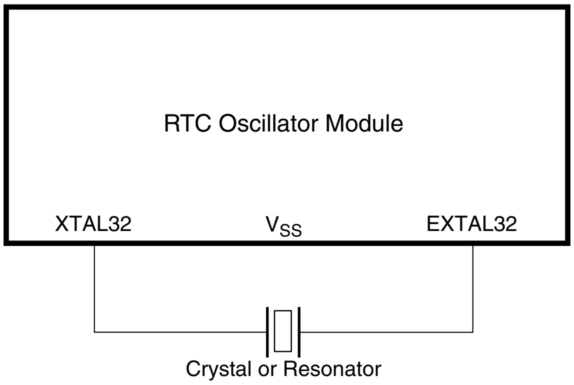


Figure 27-2. Crystal Connections

27.4 Memory Map/Register Descriptions

RTC oscillator control bits are part of the RTC registers. Refer to RTC_CR for more details.

27.5 Functional Description

As shown in Figure 27-1, the module includes an amplifier which supplies the negative resistor for the RTC oscillator. The gain of the amplifier is controlled by the amplitude detector, which optimizes the power consumption. A schmitt trigger is used to translate the sine-wave generated by this oscillator to a pulse clock out, which is a reference clock for the RTC digital core.

The oscillator includes an internal feedback resistor of approximately 100 MΩ between EXTAL32 and XTAL32.

In addition, there are two programmable capacitors with this oscillator, which can be used as the Cload of the oscillator. The programmable range is from 0pF to 30pF.

27.6 Reset Overview

There is no reset state associated with the RTC oscillator.

27.7 Interrupts

The RTC oscillator does not generate any interrupts.

Chapter 28
Local Memory Controller

28.1 Introduction

NOTE

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

The Local Memory Controller provides the ARM®Cortex-M4™ processor with tightly-coupled processor-local memories and bus paths to all slave memory spaces.

28.1.1 Block Diagram

The Cortex-M4 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 four 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
- Cache memory controller via the PS bus

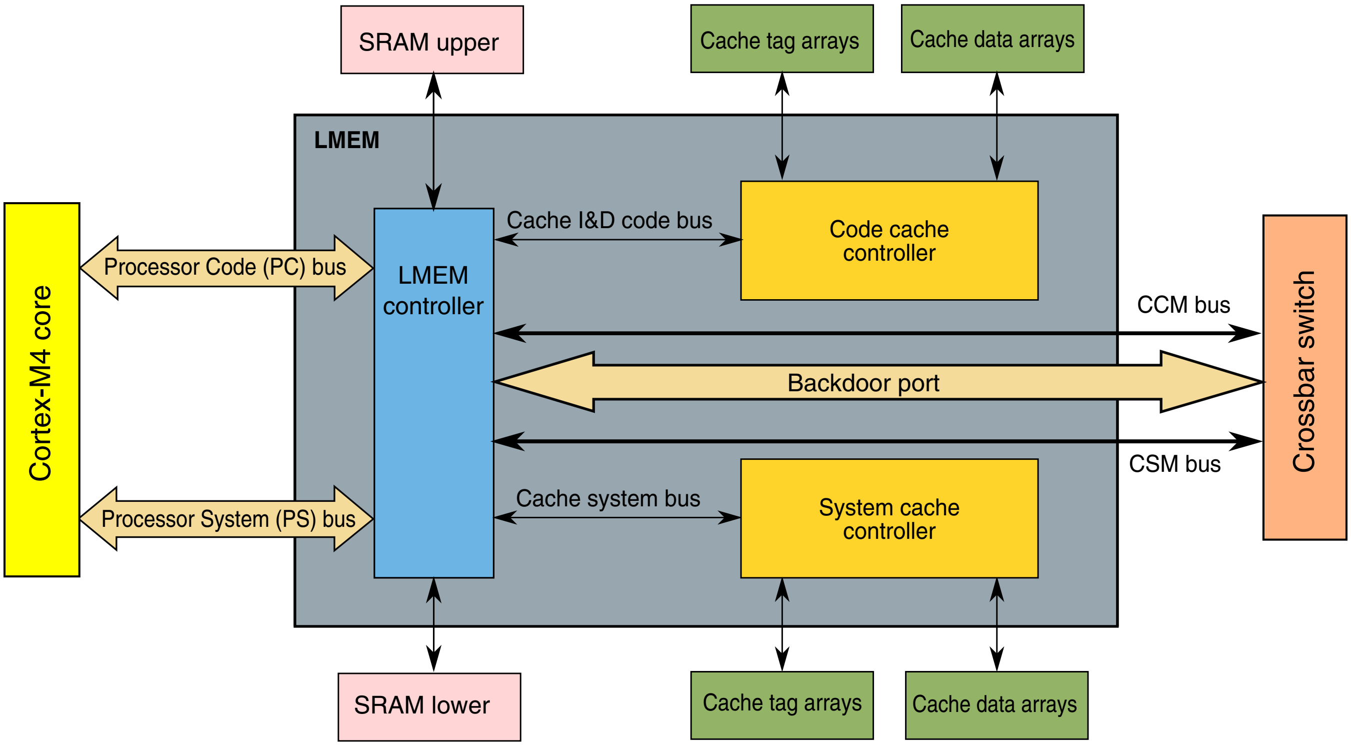


Figure 28-1. Local memory controller block diagram

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. The address spaces are device-specific and are specified in the device's Chip Configuration chapter.

28.1.2 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 three modes of operation:

1. Write-through — access to address spaces with this cache mode are cacheable.
   - A write-through read miss on the input bus causes a line read on the output bus of a 16-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 data and writes to the output bus.
2. Write-back — access to address spaces with this cache mode are cacheable.
   - A write-back read miss on the input bus will cause a line read on the output bus of a 16-byte-aligned memory address containing the desired address. This miss data is loaded into the cache and marked as valid and not modified.
   - A write-back read hit to a valid cache location will return data from the cache with no output bus access.
   - A write-back write miss will do a "read-to-write" (allocate on write miss policy for write-back mode spaces). A line read on the output bus of a 16 byte aligned memory address containing the desired write address is performed. This miss data is loaded into the cache and marked as valid and modified; and the write data will then update the appropriate cache data locations.

3. Non-cacheable — access to address spaces with this cache mode are not cacheable. These accesses bypass the cache and access the output bus.

28.2 Memory Map/Register Definition

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.

LMEM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E008_2000	Cache control register (LMEM_PCCCR)	32	R/W	0000_0000h	28.2.1/ 669
E008_2004	Cache line control register (LMEM_PCCLCR)	32	R/W	0000_0000h	28.2.2/ 670
E008_2008	Cache search address register (LMEM_PCCSAR)	32	R/W	0000_0000h	28.2.3/ 672
E008_200C	Cache read/write value register (LMEM_PCCCVR)	32	R/W	0000_0000h	28.2.4/ 673
E008_2020	Cache regions mode register (LMEM_PCCRMR)	32	R/W	AA0F_A000h	28.2.5/ 674
E008_2800	Cache control register (LMEM_PSCCR)	32	R/W	0000_0000h	28.2.6/ 677
E008_2804	Cache line control register (LMEM_PSCLCR)	32	R/W	0000_0000h	28.2.7/ 678
E008_2808	Cache search address register (LMEM_PSCSAR)	32	R/W	0000_0000h	28.2.8/ 681
E008_280C	Cache read/write value register (LMEM_PSCCVR)	32	R/W	0000_0000h	28.2.9/ 682
E008_2820	Cache regions mode register (LMEM_PSCRMR)	32	R/W	AA0F_A000h	28.2.10/ 682

28.2.1 Cache control register (LMEM_PCCCR)

Address: LMEM_PCCCR is E008_2000h base + 0h offset = E008_2000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GO		0		PUSHW1	INVW1	PUSHW0	INVW0				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							ENWRBUF	ENCACHE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PCCCR field descriptions

Field	Description
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. 0 Write: no effect. Read: no cache command active. 1 Write: initiate command indicated by bits 27-24. Read: cache command active.
30–28 Reserved	This read-only field is reserved and always has the value zero.
27 PUSHW1	Push Way 1 0 No operation 1 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). 0 No operation 1 When setting the GO bit, invalidate all lines in way 1
25 PUSHW0	Push Way 0 0 No operation 1 When setting the GO bit, push all modified lines in way 0
24 INVW0	Invalidate Way 0

Table continues on the next page...

LMEM_PCCCR field descriptions (continued)

Field	Description
	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). 0 No operation 1 When setting the GO bit, invalidate all lines in way 0.
23–2 Reserved	This read-only field is reserved and always has the value zero.
1 ENWRBUF	Enable Write Buffer 0 Write buffer disabled 1 Write buffer enabled
0 ENCACHE	Cache enable 0 Cache disabled 1 Cache enabled

28.2.2 Cache line control register (LMEM_PCCLCR)

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.

Address: LMEM_PCCLCR is E008_2000h base + 4h offset = E008_2004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LACC	LADSEL	LCMD		0	LCWAY	LCIMB	LCIVB	0			TDSEL
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	WSEL	0		CACHEADDR										0	LGO
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PCCLCR field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value zero.
27 LACC	Line access type 0 Read 1 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. 0 Cache address 1 Physical address
25–24 LCMD	Line Command 00 Search and read or write 01 Invalidate 10 Push 11 Clear
23 Reserved	This read-only field is reserved and always has the value zero.
22 LCWAY	Line Command Way Indicates the way used by the line command.
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	This read-only field is reserved and always has the value zero.
16 TDSEL	Tag/Data Select Selects tag or data for search and read or write commands. 0 Data 1 Tag
15 Reserved	This read-only field is reserved and always has the value zero.
14 WSEL	Way select Selects the way for line commands.

Table continues on the next page...

LMEM_PCCLCR field descriptions (continued)

Field	Description
	0 Way 0 1 Way 1
13–12 Reserved	This read-only field is reserved and always has the value zero.
11–2 CACHEADDR	Cache address CLCR[11:4] bits are used to access the tag arrays CLCR[11:2] bits are used to access the data arrays
1 Reserved	This read-only field is reserved and always has the value zero.
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] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits 27-24. Read: line command active.

28.2.3 Cache search address register (LMEM_PCCSAR)

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.

Address: LMEM_PCCSAR is E008_2000h base + 8h offset = E008_2008h

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	PHYADDR																														0	LGO
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

LMEM_PCCSAR field descriptions

Field	Description
31–2 PHYADDR	Physical Address PHYADDR represents bits [31:2] of the system address. CSAR[31:12] bits are used for tag compare CSAR[11:4] bits are used to access the tag arrays CSAR[11:2] bits are used to access the data arrays

Table continues on the next page...

LMEM_PCCSAR field descriptions (continued)

Field	Description
1 Reserved	This read-only field is reserved and always has the value zero.
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] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits CLCR[27:24]. Read: line command active.

28.2.4 Cache read/write value register (LMEM_PCCCVR)

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

Address: LMEM_PCCCVR is E008_2000h base + Ch offset = E008_200Ch

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	DATA																															
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

LMEM_PCCCVR field descriptions

Field	Description
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[11:4] bits are used for tag set address on reads; unused on writes • CCVR[3:2] bits are reserved For data search, read or write: • CCVR[31:0] bits are used for data array R/W value

28.2.5 Cache regions mode register (LMEM_PCCRMR)

The CRMR register allows you to demote the cache mode of various subregions within the device's memory map. Demoting the cache mode reduces the cache function applied to a memory region from write-back to write-through to non-cacheable. After a region is demoted, its cache mode can only be raised by a reset, which returns it to its default state.

To maintain cache coherency, changes to the cache mode should be completed while the address space being changed is not being accessed or the cache is disabled. Before a cache mode change, complete a cache clear all command to push and invalidate any cache entries that may have changed.

NOTE

> The address/module assignment of the 16 subregions is device-specific and are detailed in the Chip Configuration section. Some of the regions may not be used (non-cacheable), and some regions may not be capable of write-back.

Address: LMEM_PCCRMR is E008_2000h base + 20h offset = E008_2020h

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	R0		R1		R2		R3		R4		R5		R6		R7		R8		R9		R10		R11		R12		R13		R14		R15	
Reset	1	0	1	0	1	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PCCRMR field descriptions

Field	Description
31–30 R0	Region 0 mode Controls the cache mode for region 0 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
29–28 R1	Region 1 mode Controls the cache mode for region 1 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
27–26 R2	Region 2 mode Controls the cache mode for region 2

Table continues on the next page...

LMEM_PCCRMR field descriptions (continued)

Field	Description
	00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
25–24 R3	Region 3 mode Controls the cache mode for region 3 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
23–22 R4	Region 4 mode Controls the cache mode for region 4 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
21–20 R5	Region 5 mode Controls the cache mode for region 5 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
19–18 R6	Region 6 mode Controls the cache mode for region 6 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
17–16 R7	Region 7 mode Controls the cache mode for region 7 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
15–14 R8	Region 8 mode Controls the cache mode for region 8 00 Non-cacheable 01 Non-cacheable

Table continues on the next page...

LMEM_PCCRMR field descriptions (continued)

Field	Description
	10 Write-through 11 Write-back
13–12 R9	Region 9 mode Controls the cache mode for region 9 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
11–10 R10	Region 10 mode Controls the cache mode for region 10 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
9–8 R11	Region 11 mode Controls the cache mode for region 11 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
7–6 R12	Region 12 mode Controls the cache mode for region 12 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
5–4 R13	Region 13 mode Controls the cache mode for region 13 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
3–2 R14	Region 14 mode Controls the cache mode for region 14 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back

Table continues on the next page...

LMEM_PCCRMR field descriptions (continued)

Field	Description
1–0 R15	Region 15 mode Controls the cache mode for region 15 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back

28.2.6 Cache control register (LMEM_PSCCR)

Address: LMEM_PSCCR is E008_2000h base + 800h offset = E008_2800h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GO	0			PUSHW1	INVW1	PUSHW0	INVW0	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														ENWRBUF	ENCACHE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PSCCR field descriptions

Field	Description
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. 0 Write: no effect. Read: no cache command active. 1 Write: initiate command indicated by bits 27-24. Read: cache command active.
30–28 Reserved	This read-only field is reserved and always has the value zero.
27 PUSHW1	Push Way 1 0 No operation 1 When setting the GO bit, push all modified lines in way 1

Table continues on the next page...

LMEM_PSCCR field descriptions (continued)

Field	Description
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). 0 No operation 1 When setting the GO bit, invalidate all lines in way 1
25 PUSHW0	Push Way 0 0 No operation 1 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). 0 No operation 1 When setting the GO bit, invalidate all lines in way 0.
23–2 Reserved	This read-only field is reserved and always has the value zero.
1 ENWRBUF	Enable Write Buffer 0 Write buffer disabled 1 Write buffer enabled
0 ENCACHE	Cache enable 0 Cache disabled 1 Cache enabled

28.2.7 Cache line control register (LMEM_PSCLCR)

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.

Address: LMEM_PSCLCR is E008_2000h base + 804h offset = E008_2804h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LACC	LADSEL	LCMD		0	LCWAY	LCIMB	LCIVB	0			TDSEL
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	WSEL	0		CACHEADDR										0	LGO
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PSCLCR field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value zero.
27 LACC	Line access type 0 Read 1 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. 0 Cache address 1 Physical address
25–24 LCMD	Line Command 00 Search and read or write 01 Invalidate 10 Push 11 Clear
23 Reserved	This read-only field is reserved and always has the value zero.
22 LCWAY	Line Command Way Indicates the way used by the line command.
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.

Table continues on the next page...

LMEM_PSCLCR field descriptions (continued)

Field	Description
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	This read-only field is reserved and always has the value zero.
16 TDSEL	Tag/Data Select Selects tag or data for search and read or write commands. 0 Data 1 Tag
15 Reserved	This read-only field is reserved and always has the value zero.
14 WSEL	Way select Selects the way for line commands. 0 Way 0 1 Way 1
13–12 Reserved	This read-only field is reserved and always has the value zero.
11–2 CACHEADDR	Cache address CLCR[11:4] bits are used to access the tag arrays CLCR[11:2] bits are used to access the data arrays
1 Reserved	This read-only field is reserved and always has the value zero.
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] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits 27-24. Read: line command active.

28.2.8 Cache search address register (LMEM_PSCSAR)

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.

Address: LMEM_PSCSAR is E008_2000h base + 808h offset = E008_2808h

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	PHYADDR																														0	LGO
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

LMEM_PSCSAR field descriptions

Field	Description
31–2 PHYADDR	Physical Address PHYADDR represents bits [31:2] of the system address. CSAR[31:12] bits are used for tag compare CSAR[11:4] bits are used to access the tag arrays CSAR[11:2] bits are used to access the data arrays
1 Reserved	This read-only field is reserved and always has the value zero.
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] 0 Write: no effect. Read: no line command active. 1 Write: initiate line command indicated by bits CLCR[27:24]. Read: line command active.

28.2.9 Cache read/write value register (LMEM_PSCCVR)

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

Address: LMEM_PSCCVR is E008_2000h base + 80Ch offset = E008_280Ch

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	DATA																															
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

LMEM_PSCCVR field descriptions

Field	Description
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[11:4] bits are used for tag set address on reads; unused on writes • CCVR[3:2] bits are reserved For data search, read or write: • CCVR[31:0] bits are used for data array R/W value

28.2.10 Cache regions mode register (LMEM_PSCRMR)

The CRMR register allows you to demote the cache mode of various subregions within the device's memory map. Demoting the cache mode reduces the cache function applied to a memory region from write-back to write-through to non-cacheable. After a region is demoted, its cache mode can only be raised by a reset, which returns it to its default state.

To maintain cache coherency, changes to the cache mode should be completed while the address space being changed is not being accessed or the cache is disabled. Before a cache mode change, complete a cache clear all command to push and invalidate any cache entries that may have changed.

NOTE

The address/module assignment of the 16 subregions is device-specific and are detailed in the Chip Configuration section. Some of the regions may not be used (non-cacheable), and some regions may not be capable of write-back.

Address: LMEM_PSCRMR is E008_2000h base + 820h offset = E008_2820h

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	R0		R1		R2		R3		R4		R5		R6		R7		R8		R9		R10		R11		R12		R13		R14		R15	
Reset	1	0	1	0	1	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

LMEM_PSCRMR field descriptions

Field	Description
31–30 R0	Region 0 mode Controls the cache mode for region 0 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
29–28 R1	Region 1 mode Controls the cache mode for region 1 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
27–26 R2	Region 2 mode Controls the cache mode for region 2 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
25–24 R3	Region 3 mode Controls the cache mode for region 3 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
23–22 R4	Region 4 mode Controls the cache mode for region 4 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
21–20 R5	Region 5 mode Controls the cache mode for region 5

Table continues on the next page...

LMEM_PSCRMR field descriptions (continued)

Field	Description
	00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
19–18 R6	Region 6 mode Controls the cache mode for region 6 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
17–16 R7	Region 7 mode Controls the cache mode for region 7 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
15–14 R8	Region 8 mode Controls the cache mode for region 8 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
13–12 R9	Region 9 mode Controls the cache mode for region 9 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
11–10 R10	Region 10 mode Controls the cache mode for region 10 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
9–8 R11	Region 11 mode Controls the cache mode for region 11 00 Non-cacheable 01 Non-cacheable

Table continues on the next page...

LMEM_PSCRMR field descriptions (continued)

Field	Description
	10 Write-through 11 Write-back
7–6 R12	Region 12 mode Controls the cache mode for region 12 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
5–4 R13	Region 13 mode Controls the cache mode for region 13 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
3–2 R14	Region 14 mode Controls the cache mode for region 14 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back
1–0 R15	Region 15 mode Controls the cache mode for region 15 00 Non-cacheable 01 Non-cacheable 10 Write-through 11 Write-back

28.3 Functional Description

28.3.1 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 and System Caches is accessed via the core's Private Peripheral Bus (PPB).

28.3.1.1 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.

28.3.1.2 Processor Space 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 PS 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 Master1 port.

28.3.1.3 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.

28.3.2 SRAM Function

28.3.2.1 SRAM Configuration

SRAM Configuration shows how the SRAM controller is configured.

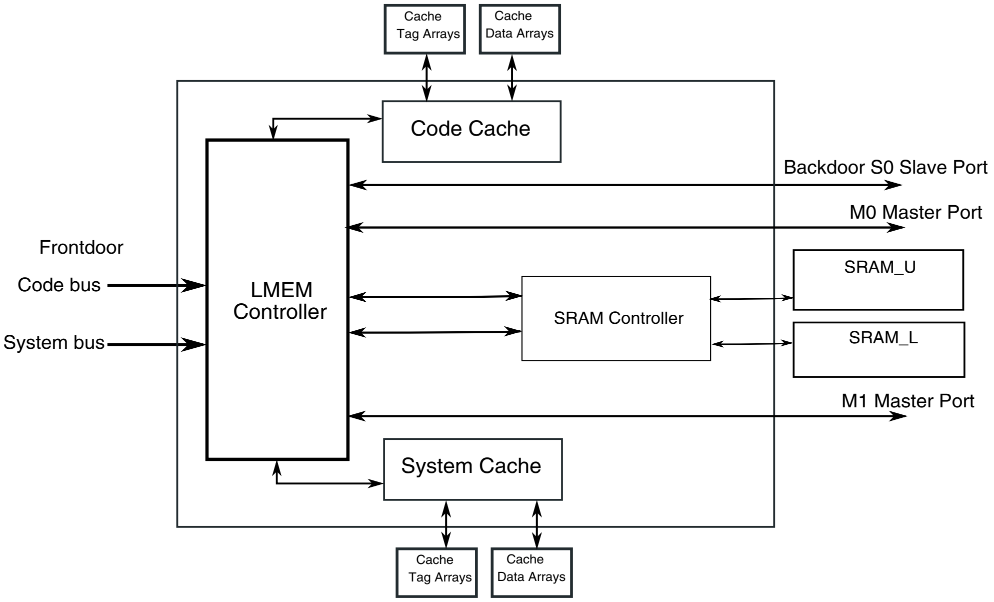


Figure 28-12. SRAM Configuration

28.3.2.2 SRAM Arrays

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

The on-chip SRAM is implemented such that the SRAM_L and SRAM_U ranges form a contiguous block in the memory map. As such:

- SRAM_L is anchored to 0x1FFF_FFFF and occupies the space before this ending address.
- SRAM_U is anchored to 0x2000_0000 and occupies the space after this beginning address.

From equal-sized memories, valid address ranges for SRAM_L and SRAM_U are then defined as:

- SRAM_L = [0x2000_0000–(SRAM_size/2)] to 0x1FFF_FFFF
- SRAM_U = 0x2000_0000 to [0x2000_0000+(SRAM_size/2)-1]

This is illustrated in Figure 28-13.

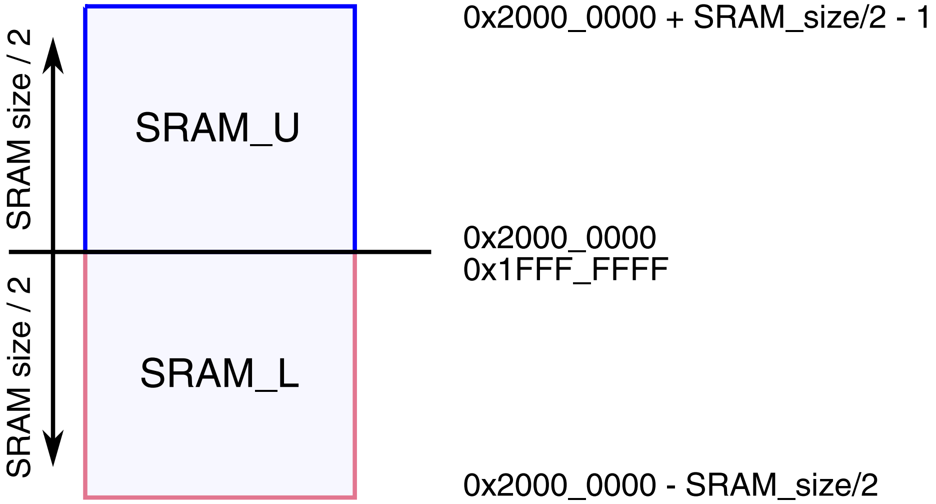


Figure 28-13. SRAM Arrays

For example, for a device containing 64 KB of SRAM the ranges are:

- SRAM_L: 0x1FFF_8000 – 0x1FFF_FFFF
- SRAM_U: 0x2000_0000 – 0x2000_7FFF

28.3.2.3 SRAM accesses

The SRAM is split into two logical arrays that are 32-bits wide:

- SRAM_L — Accessible by the code bus of the Cortex-M4 core and by the backdoor port.
- SRAM_U — Accessible by the system bus of the Cortex-M4 core and by the backdoor port.

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

Figure 28-14 illustrates the SRAM accesses within the device.

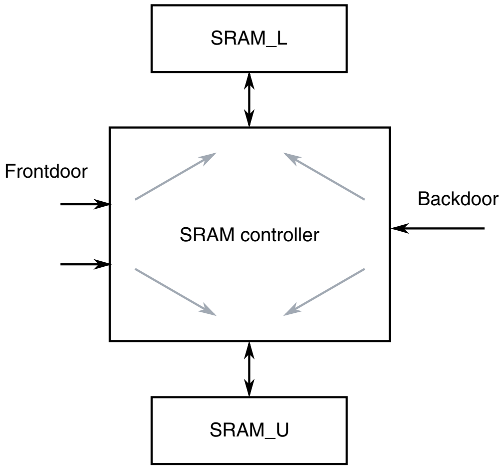


Figure 28-14. SRAM access diagram

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,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.

28.3.3 Cache Function

The caches on this device are structured as follows. Both caches have a 2-way set-associative cache structure with a total size of 8 KBytes. The caches have 32-bit address and data paths and a 16-byte line size. The cache tags and data storage use single-port, synchronous RAMs.

For these 8-KByte caches, each cache TAG function uses two 256 x 22-bit RAM arrays and the cache DATA function uses two 1024 x 32-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 four bytes of code or data.

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

CACHE - 8 KByte size = (256 sets) x (16-byte lines) x (2-way set-associative)

TAG:

- address[31:12] used in tag for compare (hit) logic
- address[11:4] used to select 1 of 256 sets
- address[3:0] not used

DATA

- address[31:12] not used
- address[11:4] used to select one of 256 sets
- address[3:2] used to select one of four 32-bit words within a set
- address[1:0] used to select the byte within the 32-bit word

28.3.4 Cache Control

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

28.3.4.1 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 28-12. 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 28-12. Cache Set Commands

CCR[27:24]				Command
PUSH W1	INVW1	PUSH W0	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.

28.3.4.2 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 bits [11:4] of the physical address. If they hit, the commands perform their action on the hit way.

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 28-13. 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

28.3.4.2.1 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 2 to step through data or at bit 4 to step through lines), and
- Set the line command go bit (CLCR[LGO]).

28.3.4.2.2 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 2 to step through data or at bit 4 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.

28.3.4.2.3 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 28-14. 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 29
Flash Memory Controller (FMC)

29.1 Introduction

NOTE

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

The Flash Memory Controller (FMC) is a memory acceleration unit that provides:

- an interface between the device and the multi-bank nonvolatile memory.
- buffers that can accelerate flash memory and FlexNVM data transfers.

29.1.1 Overview

The Flash Memory Controller manages the interface between the device and the multi-bank flash memory. The FMC receives status information detailing the configuration of the memory and uses this information to ensure a proper interface. The following table shows the supported 8-bit, 16-bit, and 32-bit read/write operations.

Flash memory type	Read	Write
Program flash memory	x	—[1]
FlexNVM used as data flash memory	x	—[1]
FlexNVM and FlexRAM used as EEPROM	x	x

[1] A write operation to program flash memory or to FlexNVM used as data flash memory results in a bus error.

In addition, the FMC provides three separate mechanisms for accelerating the interface between the device and the flash memory. A 128-bit speculation buffer can prefetch the next 128-bit flash memory location, and both a 4-way, 4-set cache (with 128-bit entries) and a single-entry 128-bit buffer can store previously accessed flash memory or FlexNVM data for quick access times.

29.1.2 Features

The FMC's features include:

- Interface between the device and the multi-bank flash memory and FlexMemory:
  - 8-bit, 16-bit, and 32-bit read operations to program flash memory and FlexNVM used as data flash memory.
  - 8-bit, 16-bit, and 32-bit read and write operations to FlexNVM and FlexRAM used as EEPROM.
  - Read accesses to consecutive 32-bit spaces in memory return the second, third, and fourth read data with no wait states. The memory returns 128 bits via the 32-bit bus access.
  - Crossbar master access protection for setting no access, read only access, write only access, or read/write access for each crossbar master.
- Acceleration of data transfer from program flash memory and FlexMemory to the device:
  - 128-bit prefetch speculation buffer with controls for instruction/data access per master
  - 4-way, 4-set, 128-bit line size cache for a total of sixteen 128-bit entries with controls for replacement algorithm and lock per way
  - Single-entry buffer with enable
  - Invalidation control for the speculation buffer and the single-entry buffer

29.2 Modes of operation

The FMC only operates when the device accesses the flash memory or FlexMemory.

For any device power mode where the flash memory or FlexMemory cannot be accessed, the FMC is disabled.

29.3 External signal description

The FMC has no external signals.

29.4 Memory map and register descriptions

The programming model consists of the FMC control registers and the program visible cache (data and tag/valid entries).

NOTE

Program the registers only while the flash controller is idle (for example, execute from RAM). Changing configuration settings while a flash access is in progress can lead to non-deterministic behavior.

Table 29-2. FMC register access

Registers	Read access		Write access	
	Mode	Length	Mode	Length
Control registers (PFAPR, PFB01CR, PFB23CR)	Supervisor (privileged) mode or user mode	32 bits	Supervisor (privileged) mode only	32 bits
Cache registers	Supervisor (privileged) mode or user mode	32 bits	Supervisor (privileged) mode only	32 bits

NOTE

Accesses to unimplemented registers within the FMC's address space return a bus error.

The 16 cache entries, both data and tag/valid, can be read at any time.

NOTE

System software is required to maintain memory coherence when any segment of the flash cache is programmed. For example, all buffer data associated with the reprogrammed flash should be invalidated. Accordingly, cache program visible writes must occur after a programming or erase event is completed and before the new memory image is accessed.

The cache is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. The following table elaborates on the tag/valid and data entries.

Table 29-3. Program visible cache registers

Cache storage	Based at offset	Contents of 32-bit read	Nomenclature	Nomenclature example
Tag	100h	12'h0, tag[19:6], 5'h0, valid	In TAGVDWxSy, x denotes the way and y denotes the set.	TAGVDW2S3 is the 14-bit tag and 1-bit valid for cache entry way 2, set 3.

Table continues on the next page...

Table 29-3. Program visible cache registers (continued)

Cache storage	Based at offset	Contents of 32-bit read	Nomenclature	Nomenclature example
Data	200h	One of the four longwords in a 128-bit cache entry	In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost).	For data entry way 1, set 3, DATAW1S3UM represents bits [127:96], DATAW1S3MU represents bits [95:64], DATAW1S3ML represents bits [63:32], and DATAW1S3LM represents bits [31:0].

FMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4001_F000	Flash Access Protection Register (FMC_PFAPR)	32	R/W	00F8_003Fh	29.4.1/ 702
4001_F004	Flash Bank 0-1 Control Register (FMC_PFB01CR)	32	R/W	3004_001Fh	29.4.2/ 705
4001_F008	Flash Bank 2-3 Control Register (FMC_PFB23CR)	32	R/W	3004_001Fh	29.4.3/ 708
4001_F100	Cache Tag Storage (FMC_TAGVDW0S0)	32	R/W	0000_0000h	29.4.4/ 710
4001_F104	Cache Tag Storage (FMC_TAGVDW0S1)	32	R/W	0000_0000h	29.4.4/ 710
4001_F108	Cache Tag Storage (FMC_TAGVDW0S2)	32	R/W	0000_0000h	29.4.4/ 710
4001_F10C	Cache Tag Storage (FMC_TAGVDW0S3)	32	R/W	0000_0000h	29.4.4/ 710
4001_F110	Cache Tag Storage (FMC_TAGVDW1S0)	32	R/W	0000_0000h	29.4.5/ 711
4001_F114	Cache Tag Storage (FMC_TAGVDW1S1)	32	R/W	0000_0000h	29.4.5/ 711
4001_F118	Cache Tag Storage (FMC_TAGVDW1S2)	32	R/W	0000_0000h	29.4.5/ 711
4001_F11C	Cache Tag Storage (FMC_TAGVDW1S3)	32	R/W	0000_0000h	29.4.5/ 711
4001_F120	Cache Tag Storage (FMC_TAGVDW2S0)	32	R/W	0000_0000h	29.4.6/ 712
4001_F124	Cache Tag Storage (FMC_TAGVDW2S1)	32	R/W	0000_0000h	29.4.6/ 712
4001_F128	Cache Tag Storage (FMC_TAGVDW2S2)	32	R/W	0000_0000h	29.4.6/ 712
4001_F12C	Cache Tag Storage (FMC_TAGVDW2S3)	32	R/W	0000_0000h	29.4.6/ 712

Table continues on the next page...

FMC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4001_F130	Cache Tag Storage (FMC_TAGVDW3S0)	32	R/W	0000_0000h	29.4.7/ 713
4001_F134	Cache Tag Storage (FMC_TAGVDW3S1)	32	R/W	0000_0000h	29.4.7/ 713
4001_F138	Cache Tag Storage (FMC_TAGVDW3S2)	32	R/W	0000_0000h	29.4.7/ 713
4001_F13C	Cache Tag Storage (FMC_TAGVDW3S3)	32	R/W	0000_0000h	29.4.7/ 713
4001_F200	Cache Data Storage (uppermost word) (FMC_DATAW0S0UM)	32	R/W	0000_0000h	29.4.8/ 714
4001_F204	Cache Data Storage (mid-upper word) (FMC_DATAW0S0MU)	32	R/W	0000_0000h	29.4.9/ 715
4001_F208	Cache Data Storage (mid-lower word) (FMC_DATAW0S0ML)	32	R/W	0000_0000h	29.4.10/ 716
4001_F20C	Cache Data Storage (lowermost word) (FMC_DATAW0S0LM)	32	R/W	0000_0000h	29.4.11/ 717
4001_F210	Cache Data Storage (uppermost word) (FMC_DATAW0S1UM)	32	R/W	0000_0000h	29.4.8/ 714
4001_F214	Cache Data Storage (mid-upper word) (FMC_DATAW0S1MU)	32	R/W	0000_0000h	29.4.9/ 715
4001_F218	Cache Data Storage (mid-lower word) (FMC_DATAW0S1ML)	32	R/W	0000_0000h	29.4.10/ 716
4001_F21C	Cache Data Storage (lowermost word) (FMC_DATAW0S1LM)	32	R/W	0000_0000h	29.4.11/ 717
4001_F220	Cache Data Storage (uppermost word) (FMC_DATAW0S2UM)	32	R/W	0000_0000h	29.4.8/ 714
4001_F224	Cache Data Storage (mid-upper word) (FMC_DATAW0S2MU)	32	R/W	0000_0000h	29.4.9/ 715
4001_F228	Cache Data Storage (mid-lower word) (FMC_DATAW0S2ML)	32	R/W	0000_0000h	29.4.10/ 716
4001_F22C	Cache Data Storage (lowermost word) (FMC_DATAW0S2LM)	32	R/W	0000_0000h	29.4.11/ 717
4001_F230	Cache Data Storage (uppermost word) (FMC_DATAW0S3UM)	32	R/W	0000_0000h	29.4.8/ 714
4001_F234	Cache Data Storage (mid-upper word) (FMC_DATAW0S3MU)	32	R/W	0000_0000h	29.4.9/ 715
4001_F238	Cache Data Storage (mid-lower word) (FMC_DATAW0S3ML)	32	R/W	0000_0000h	29.4.10/ 716
4001_F23C	Cache Data Storage (lowermost word) (FMC_DATAW0S3LM)	32	R/W	0000_0000h	29.4.11/ 717

Table continues on the next page...

FMC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4001_F240	Cache Data Storage (uppermost word) (FMC_DATAW1S0UM)	32	R/W	0000_0000h	29.4.12/ 718
4001_F244	Cache Data Storage (mid-upper word) (FMC_DATAW1S0MU)	32	R/W	0000_0000h	29.4.13/ 719
4001_F248	Cache Data Storage (mid-lower word) (FMC_DATAW1S0ML)	32	R/W	0000_0000h	29.4.14/ 720
4001_F24C	Cache Data Storage (lowermost word) (FMC_DATAW1S0LM)	32	R/W	0000_0000h	29.4.15/ 721
4001_F250	Cache Data Storage (uppermost word) (FMC_DATAW1S1UM)	32	R/W	0000_0000h	29.4.12/ 718
4001_F254	Cache Data Storage (mid-upper word) (FMC_DATAW1S1MU)	32	R/W	0000_0000h	29.4.13/ 719
4001_F258	Cache Data Storage (mid-lower word) (FMC_DATAW1S1ML)	32	R/W	0000_0000h	29.4.14/ 720
4001_F25C	Cache Data Storage (lowermost word) (FMC_DATAW1S1LM)	32	R/W	0000_0000h	29.4.15/ 721
4001_F260	Cache Data Storage (uppermost word) (FMC_DATAW1S2UM)	32	R/W	0000_0000h	29.4.12/ 718
4001_F264	Cache Data Storage (mid-upper word) (FMC_DATAW1S2MU)	32	R/W	0000_0000h	29.4.13/ 719
4001_F268	Cache Data Storage (mid-lower word) (FMC_DATAW1S2ML)	32	R/W	0000_0000h	29.4.14/ 720
4001_F26C	Cache Data Storage (lowermost word) (FMC_DATAW1S2LM)	32	R/W	0000_0000h	29.4.15/ 721
4001_F270	Cache Data Storage (uppermost word) (FMC_DATAW1S3UM)	32	R/W	0000_0000h	29.4.12/ 718
4001_F274	Cache Data Storage (mid-upper word) (FMC_DATAW1S3MU)	32	R/W	0000_0000h	29.4.13/ 719
4001_F278	Cache Data Storage (mid-lower word) (FMC_DATAW1S3ML)	32	R/W	0000_0000h	29.4.14/ 720
4001_F27C	Cache Data Storage (lowermost word) (FMC_DATAW1S3LM)	32	R/W	0000_0000h	29.4.15/ 721
4001_F280	Cache Data Storage (uppermost word) (FMC_DATAW2S0UM)	32	R/W	0000_0000h	29.4.16/ 722
4001_F284	Cache Data Storage (mid-upper word) (FMC_DATAW2S0MU)	32	R/W	0000_0000h	29.4.17/ 723
4001_F288	Cache Data Storage (mid-lower word) (FMC_DATAW2S0ML)	32	R/W	0000_0000h	29.4.18/ 724
4001_F28C	Cache Data Storage (lowermost word) (FMC_DATAW2S0LM)	32	R/W	0000_0000h	29.4.19/ 725

Table continues on the next page...

FMC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4001_F290	Cache Data Storage (uppermost word) (FMC_DATAW2S1UM)	32	R/W	0000_0000h	29.4.16/ 722
4001_F294	Cache Data Storage (mid-upper word) (FMC_DATAW2S1MU)	32	R/W	0000_0000h	29.4.17/ 723
4001_F298	Cache Data Storage (mid-lower word) (FMC_DATAW2S1ML)	32	R/W	0000_0000h	29.4.18/ 724
4001_F29C	Cache Data Storage (lowermost word) (FMC_DATAW2S1LM)	32	R/W	0000_0000h	29.4.19/ 725
4001_F2A0	Cache Data Storage (uppermost word) (FMC_DATAW2S2UM)	32	R/W	0000_0000h	29.4.16/ 722
4001_F2A4	Cache Data Storage (mid-upper word) (FMC_DATAW2S2MU)	32	R/W	0000_0000h	29.4.17/ 723
4001_F2A8	Cache Data Storage (mid-lower word) (FMC_DATAW2S2ML)	32	R/W	0000_0000h	29.4.18/ 724
4001_F2AC	Cache Data Storage (lowermost word) (FMC_DATAW2S2LM)	32	R/W	0000_0000h	29.4.19/ 725
4001_F2B0	Cache Data Storage (uppermost word) (FMC_DATAW2S3UM)	32	R/W	0000_0000h	29.4.16/ 722
4001_F2B4	Cache Data Storage (mid-upper word) (FMC_DATAW2S3MU)	32	R/W	0000_0000h	29.4.17/ 723
4001_F2B8	Cache Data Storage (mid-lower word) (FMC_DATAW2S3ML)	32	R/W	0000_0000h	29.4.18/ 724
4001_F2BC	Cache Data Storage (lowermost word) (FMC_DATAW2S3LM)	32	R/W	0000_0000h	29.4.19/ 725
4001_F2C0	Cache Data Storage (uppermost word) (FMC_DATAW3S0UM)	32	R/W	0000_0000h	29.4.20/ 726
4001_F2C4	Cache Data Storage (mid-upper word) (FMC_DATAW3S0MU)	32	R/W	0000_0000h	29.4.21/ 727
4001_F2C8	Cache Data Storage (mid-lower word) (FMC_DATAW3S0ML)	32	R/W	0000_0000h	29.4.22/ 728
4001_F2CC	Cache Data Storage (lowermost word) (FMC_DATAW3S0LM)	32	R/W	0000_0000h	29.4.23/ 729
4001_F2D0	Cache Data Storage (uppermost word) (FMC_DATAW3S1UM)	32	R/W	0000_0000h	29.4.20/ 726
4001_F2D4	Cache Data Storage (mid-upper word) (FMC_DATAW3S1MU)	32	R/W	0000_0000h	29.4.21/ 727
4001_F2D8	Cache Data Storage (mid-lower word) (FMC_DATAW3S1ML)	32	R/W	0000_0000h	29.4.22/ 728
4001_F2DC	Cache Data Storage (lowermost word) (FMC_DATAW3S1LM)	32	R/W	0000_0000h	29.4.23/ 729

Table continues on the next page...

FMC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4001_F2E0	Cache Data Storage (uppermost word) (FMC_DATAW3S2UM)	32	R/W	0000_0000h	29.4.20/ 726
4001_F2E4	Cache Data Storage (mid-upper word) (FMC_DATAW3S2MU)	32	R/W	0000_0000h	29.4.21/ 727
4001_F2E8	Cache Data Storage (mid-lower word) (FMC_DATAW3S2ML)	32	R/W	0000_0000h	29.4.22/ 728
4001_F2EC	Cache Data Storage (lowermost word) (FMC_DATAW3S2LM)	32	R/W	0000_0000h	29.4.23/ 729
4001_F2F0	Cache Data Storage (uppermost word) (FMC_DATAW3S3UM)	32	R/W	0000_0000h	29.4.20/ 726
4001_F2F4	Cache Data Storage (mid-upper word) (FMC_DATAW3S3MU)	32	R/W	0000_0000h	29.4.21/ 727
4001_F2F8	Cache Data Storage (mid-lower word) (FMC_DATAW3S3ML)	32	R/W	0000_0000h	29.4.22/ 728
4001_F2FC	Cache Data Storage (lowermost word) (FMC_DATAW3S3LM)	32	R/W	0000_0000h	29.4.23/ 729

29.4.1 Flash Access Protection Register (FMC_PFAPR)

Address: FMC_PFAPR is 4001_F000h base + 0h offset = 4001_F000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								M7PFD	M6PFD	M5PFD	M4PFD	M3PFD	M2PFD	M1PFD	M0PFD
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M7AP[1:0]		M6AP[1:0]		M5AP[1:0]		M4AP[1:0]		M3AP[1:0]		M2AP[1:0]		M1AP[1:0]		M0AP[1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

FMC_PFAPR field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value zero.
23 M7PFD	Master 7 Prefetch Disable

Table continues on the next page...

FMC_PFAPR field descriptions (continued)

Field	Description
	These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
22 M6PFD	Master 6 Prefetch Disable These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
21 M5PFD	Master 5 Prefetch Disable These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
20 M4PFD	Master 4 Prefetch Disable These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
19 M3PFD	Master 3 Prefetch Disable These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
18 M2PFD	Master 2 Prefetch Disable These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
17 M1PFD	Master 1 Prefetch Disable These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits. 0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
16 M0PFD	Master 0 Prefetch Disable These bits control whether prefetching is enabled based on the logical number of the requesting crossbar switch master. This field is further qualified by the PFBnCR[BxDPE,BxIPE] bits.

Table continues on the next page...

FMC_PFAPR field descriptions (continued)

Field	Description
	0 Prefetching for this master is enabled. 1 Prefetching for this master is disabled.
15–14 M7AP[1:0]	Master 7 Access Protection This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master. 01 Only read accesses may be performed by this master. 10 Only write accesses may be performed by this master. 11 Both read and write accesses may be performed by this master.
13–12 M6AP[1:0]	Master 6 Access Protection This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
11–10 M5AP[1:0]	Master 5 Access Protection This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
9–8 M4AP[1:0]	Master 4 Access Protection This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
7–6 M3AP[1:0]	Master 3 Access Protection This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
5–4 M2AP[1:0]	Master 2 Access Protection

Table continues on the next page...

FMC_PFAPR field descriptions (continued)

Field	Description
	This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
3–2 M1AP[1:0]	Master 1 Access Protection This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master
1–0 M0AP[1:0]	Master 0 Access Protection This field controls whether read and write access to the flash are allowed based on the logical master number of the requesting crossbar switch master. 00 No access may be performed by this master 01 Only read accesses may be performed by this master 10 Only write accesses may be performed by this master 11 Both read and write accesses may be performed by this master

29.4.2 Flash Bank 0-1 Control Register (FMC_PFB01CR)

This register controls the operation of memory banks 0 and 1.

Address: FMC_PFB01CR is 4001_F000h base + 4h offset = 4001_F004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	B01RWSC[3:0]				CLCK_WAY[3:0]				0				0	B01MW[1:0]		0
W									CINV_WAY[3:0]				S_B_INV			
Reset	0	0	1	1	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	0								CRC[2:0]			B01DCE	B01ICE	B01DPE	B01IPE	B01SEBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

FMC_PFB01CR field descriptions

Field	Description
31–28 B01RWSC[3:0]	Bank 0-1 Read Wait State Control This read-only field defines the number of wait states required to access the bank 0-1 flash memory. The relationship between the read access time of the flash array (expressed in system clock cycles) and RWSC is defined as: Access time of flash array [system clocks] = RWSC + 1 The FMC automatically calculates this value based on the ratio of the system clock speed to the flash clock speed. For example, when this ratio is 4:1, the field's value is 3h.
27–24 CLCK_WAY[3:0]	Cache Lock Way x These bits determine if the given cache way is locked such that its contents will not be displaced by future misses. The bit setting definitions are for each bit in the field. 0 Cache way is unlocked and may be displaced 1 Cache way is locked and its contents are not displaced
23–20 CINV_WAY[3:0]	Cache Invalidate Way x These bits determine if the given cache way is to be invalidated (cleared). When a bit within this field is written, the corresponding cache way is immediately invalidated: the way's tag, data, and valid contents are cleared. This field always reads as zero. Cache invalidation takes precedence over locking. The cache is invalidated by system reset. System software is required to maintain memory coherency when any segment of the flash memory is programmed or erased. Accordingly, cache invalidations must occur after a programming or erase event is completed and before the new memory image is accessed. The bit setting definitions are for each bit in the field. 0 No cache way invalidation for the corresponding cache 1 Invalidate cache way for the corresponding cache: clear the tag, data, and vld bits of ways selected
19 S_B_INV	Invalidate Prefetch Speculation Buffer This bit determines if the FMC's prefetch speculation buffer and the single entry page buffer are to be invalidated (cleared). When this bit is written, the speculation buffer and single entry buffer are immediately cleared. This bit always reads as zero. 0 Speculation buffer and single entry buffer are not affected. 1 Invalidate (clear) speculation buffer and single entry buffer.
18–17 B01MW[1:0]	Bank 0-1 Memory Width This read-only field defines the width of the bank 0-1 memory. 00 32 bits 01 64 bits 10 128 bits 11 Reserved
16 Reserved	This read-only field is reserved and always has the value zero.
15–8 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

FMC_PFB01CR field descriptions (continued)

Field	Description
7–5 CRC[2:0]	Cache Replacement Control This 3-bit field defines the replacement algorithm for accesses that are cached. 000 LRU replacement algorithm per set across all four ways 001 Reserved 010 Independent LRU with ways [0-1] for ifetches, [2-3] for data 011 Independent LRU with ways [0-2] for ifetches, [3] for data 1xx Reserved
4 B01DCE	Bank 0-1 Data Cache Enable This bit controls whether data references are loaded into the cache. 0 Do not cache data references. 1 Cache data references.
3 B01ICE	Bank 0-1 Instruction Cache Enable This bit controls whether instruction fetches are loaded into the cache. 0 Do not cache instruction fetches. 1 Cache instruction fetches.
2 B01DPE	Bank 0-1 Data Prefetch Enable This bit controls whether prefetches (or speculative accesses) are initiated in response to data references. 0 Do not prefetch in response to data references. 1 Enable prefetches in response to data references.
1 B01IPE	Bank 0-1 Instruction Prefetch Enable This bit controls whether prefetches (or speculative accesses) are initiated in response to instruction fetches. 0 Do not prefetch in response to instruction fetches. 1 Enable prefetches in response to instruction fetches.
0 B01SEBE	Bank 0-1 Single Entry Buffer Enable This bit controls whether the single entry page buffer is enabled in response to flash read accesses. Its operation is independent from the cache of banks 2-3. A high-to-low transition of this enable forces the page buffer to be invalidated. 0 Single entry buffer is disabled. 1 Single entry buffer is enabled.

29.4.3 Flash Bank 2-3 Control Register (FMC_PFB23CR)

This register has a format similar to that for PFB01CR, except it controls the operation of memory banks 2-3, and the "global" cache control fields are empty.

Address: FMC_PFB23CR is 4001_F000h base + 8h offset = 4001_F008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	B23RWSC[3:0]				0									B23MW[1:0]		0
W																
Reset	0	0	1	1	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	0								0			B23DCE	B23ICE	B23DPE	B23IPE	B23SEBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

FMC_PFB23CR field descriptions

Field	Description
31–28 B23RWSC[3:0]	Bank 2-3 Read Wait State Control This read-only field defines the number of wait states required to access the bank 2-3 flash memory. The relationship between the read access time of the flash array (expressed in system clock cycles) and RWSC is defined as: Access time of flash array [system clocks] = RWSC + 1 The FMC automatically calculates this value based on the ratio of the system clock speed to the flash clock speed. For example, when this ratio is 4:1, the field's value is 3h.
27–19 Reserved	This read-only field is reserved and always has the value zero.
18–17 B23MW[1:0]	Bank 2-3 Memory Width This read-only field defines the width of the bank 2-3 memory. 00 32 bits 01 64 bits 10 128 bits 11 Reserved
16 Reserved	This read-only field is reserved and always has the value zero.
15–8 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

FMC_PFB23CR field descriptions (continued)

Field	Description
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 B23DCE	Bank 2-3 Data Cache Enable This bit controls whether data references are loaded into the cache. 0 Do not cache data references. 1 Cache data references.
3 B23ICE	Bank 2-3 Instruction Cache Enable This bit controls whether instruction fetches are loaded into the cache. 0 Do not cache instruction fetches. 1 Cache instruction fetches.
2 B23DPE	Bank 2-3 Data Prefetch Enable This bit controls whether prefetches (or speculative accesses) are initiated in response to data references. 0 Do not prefetch in response to data references. 1 Enable prefetches in response to data references.
1 B23IPE	Bank 2-3 Instruction Prefetch Enable This bit controls whether prefetches (or speculative accesses) are initiated in response to instruction fetches. 0 Do not prefetch in response to instruction fetches. 1 Enable prefetches in response to instruction fetches.
0 B23SEBE	Bank 2-3 Single Entry Buffer Enable This bit controls whether the single entry buffer is enabled in response to flash read accesses. Its operation is independent from the cache of banks 0-1. A high-to-low transition of this enable forces the page buffer to be invalidated. 0 Single entry buffer is disabled. 1 Single entry buffer is enabled.

29.4.4 Cache Tag Storage (FMC_TAGVDW0S*n*)

The 128-entry cache is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all 3 sets (n=0-3) in way 0.

Addresses: FMC_TAGVDW0S0 is 4001_F000h base + 100h offset = 4001_F100h
FMC_TAGVDW0S1 is 4001_F000h base + 104h offset = 4001_F104h
FMC_TAGVDW0S2 is 4001_F000h base + 108h offset = 4001_F108h
FMC_TAGVDW0S3 is 4001_F000h base + 10Ch offset = 4001_F10Ch

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												tag[19:6]														0					valid
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

FMC_TAGVDW0S*n* field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–6 tag[19:6]	13-bit tag for cache entry
5–1 Reserved	This read-only field is reserved and always has the value zero.
0 valid	1-bit valid for cache entry

29.4.5 Cache Tag Storage (FMC_TAGVDW1S*n*)

The 128-entry cache is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all 3 sets (n=0-3) in way 1.

Addresses: FMC_TAGVDW1S0 is 4001_F000h base + 110h offset = 4001_F110h
FMC_TAGVDW1S1 is 4001_F000h base + 114h offset = 4001_F114h
FMC_TAGVDW1S2 is 4001_F000h base + 118h offset = 4001_F118h
FMC_TAGVDW1S3 is 4001_F000h base + 11Ch offset = 4001_F11Ch

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												tag[19:6]														0					valid
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

FMC_TAGVDW1S*n* field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–6 tag[19:6]	13-bit tag for cache entry
5–1 Reserved	This read-only field is reserved and always has the value zero.
0 valid	1-bit valid for cache entry

29.4.6 Cache Tag Storage (FMC_TAGVDW2S*n*)

The 128-entry cache is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all 3 sets (n=0-3) in way 2.

Addresses: FMC_TAGVDW2S0 is 4001_F000h base + 120h offset = 4001_F120h
FMC_TAGVDW2S1 is 4001_F000h base + 124h offset = 4001_F124h
FMC_TAGVDW2S2 is 4001_F000h base + 128h offset = 4001_F128h
FMC_TAGVDW2S3 is 4001_F000h base + 12Ch offset = 4001_F12Ch

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												tag[19:6]														0					valid
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

FMC_TAGVDW2S*n* field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–6 tag[19:6]	13-bit tag for cache entry
5–1 Reserved	This read-only field is reserved and always has the value zero.
0 valid	1-bit valid for cache entry

29.4.7 Cache Tag Storage (FMC_TAGVDW3S*n*)

The 128-entry cache is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In TAGVDWxSy, x denotes the way, and y denotes the set. This section represents tag/vld information for all 3 sets (n=0-3) in way 3.

Addresses: FMC_TAGVDW3S0 is 4001_F000h base + 130h offset = 4001_F130h
FMC_TAGVDW3S1 is 4001_F000h base + 134h offset = 4001_F134h
FMC_TAGVDW3S2 is 4001_F000h base + 138h offset = 4001_F138h
FMC_TAGVDW3S3 is 4001_F000h base + 13Ch offset = 4001_F13Ch

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												tag[19:6]														0					valid
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

FMC_TAGVDW3S*n* field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–6 tag[19:6]	13-bit tag for cache entry
5–1 Reserved	This read-only field is reserved and always has the value zero.
0 valid	1-bit valid for cache entry

29.4.8 Cache Data Storage (uppermost word) (FMC_DATAW0SUM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW0S0UM is 4001_F000h base + 200h offset = 4001_F200h
FMC_DATAW0S1UM is 4001_F000h base + 210h offset = 4001_F210h
FMC_DATAW0S2UM is 4001_F000h base + 220h offset = 4001_F220h
FMC_DATAW0S3UM is 4001_F000h base + 230h offset = 4001_F230h

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	data[127:96]																															
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

FMC_DATAW0S*n*UM field descriptions

Field	Description
31–0 data[127:96]	Bits [127:96] of data entry

29.4.9 Cache Data Storage (mid-upper word) (FMC_DATAW0SMU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW0S0MU is 4001_F000h base + 204h offset = 4001_F204h
FMC_DATAW0S1MU is 4001_F000h base + 214h offset = 4001_F214h
FMC_DATAW0S2MU is 4001_F000h base + 224h offset = 4001_F224h
FMC_DATAW0S3MU is 4001_F000h base + 234h offset = 4001_F234h

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	data[95:64]																															
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

FMC_DATAW0S*n*MU field descriptions

Field	Description
31–0 data[95:64]	Bits [95:64] of data entry

29.4.10 Cache Data Storage (mid-lower word) (FMC_DATAW0SML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW0S0ML is 4001_F000h base + 208h offset = 4001_F208h
FMC_DATAW0S1ML is 4001_F000h base + 218h offset = 4001_F218h
FMC_DATAW0S2ML is 4001_F000h base + 228h offset = 4001_F228h
FMC_DATAW0S3ML is 4001_F000h base + 238h offset = 4001_F238h

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	data[63:32]																															
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

FMC_DATAW0S*n*ML field descriptions

Field	Description
31–0 data[63:32]	Bits [63:32] of data entry

29.4.11 Cache Data Storage (lowermost word) (FMC_DATAW0SLM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW0S0LM is 4001_F000h base + 20Ch offset = 4001_F20Ch
FMC_DATAW0S1LM is 4001_F000h base + 21Ch offset = 4001_F21Ch
FMC_DATAW0S2LM is 4001_F000h base + 22Ch offset = 4001_F22Ch
FMC_DATAW0S3LM is 4001_F000h base + 23Ch offset = 4001_F23Ch

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	data[31:0]																															
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

FMC_DATAW0S*n*LM field descriptions

Field	Description
31–0 data[31:0]	Bits [31:0] of data entry

29.4.12 Cache Data Storage (uppermost word) (FMC_DATAW1SUM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW1S0UM is 4001_F000h base + 240h offset = 4001_F240h
FMC_DATAW1S1UM is 4001_F000h base + 250h offset = 4001_F250h
FMC_DATAW1S2UM is 4001_F000h base + 260h offset = 4001_F260h
FMC_DATAW1S3UM is 4001_F000h base + 270h offset = 4001_F270h

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	data[127:96]																															
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

FMC_DATAW1S*n*UM field descriptions

Field	Description
31–0 data[127:96]	Bits [127:96] of data entry

29.4.13 Cache Data Storage (mid-upper word) (FMC_DATAW1SMU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW1S0MU is 4001_F000h base + 244h offset = 4001_F244h
FMC_DATAW1S1MU is 4001_F000h base + 254h offset = 4001_F254h
FMC_DATAW1S2MU is 4001_F000h base + 264h offset = 4001_F264h
FMC_DATAW1S3MU is 4001_F000h base + 274h offset = 4001_F274h

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	data[95:64]																															
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

FMC_DATAW1S*n*MU field descriptions

Field	Description
31–0 data[95:64]	Bits [95:64] of data entry

29.4.14 Cache Data Storage (mid-lower word) (FMC_DATAW1SML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW1S0ML is 4001_F000h base + 248h offset = 4001_F248h
FMC_DATAW1S1ML is 4001_F000h base + 258h offset = 4001_F258h
FMC_DATAW1S2ML is 4001_F000h base + 268h offset = 4001_F268h
FMC_DATAW1S3ML is 4001_F000h base + 278h offset = 4001_F278h

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	data[63:32]																															
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

FMC_DATAW1S*n*ML field descriptions

Field	Description
31–0 data[63:32]	Bits [63:32] of data entry

29.4.15 Cache Data Storage (lowermost word) (FMC_DATAW1SLM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW1S0LM is 4001_F000h base + 24Ch offset = 4001_F24Ch
FMC_DATAW1S1LM is 4001_F000h base + 25Ch offset = 4001_F25Ch
FMC_DATAW1S2LM is 4001_F000h base + 26Ch offset = 4001_F26Ch
FMC_DATAW1S3LM is 4001_F000h base + 27Ch offset = 4001_F27Ch

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	data[31:0]																															
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

FMC_DATAW1S*n*LM field descriptions

Field	Description
31–0 data[31:0]	Bits [31:0] of data entry

29.4.16 Cache Data Storage (uppermost word) (FMC_DATAW2SUM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW2S0UM is 4001_F000h base + 280h offset = 4001_F280h
FMC_DATAW2S1UM is 4001_F000h base + 290h offset = 4001_F290h
FMC_DATAW2S2UM is 4001_F000h base + 2A0h offset = 4001_F2A0h
FMC_DATAW2S3UM is 4001_F000h base + 2B0h offset = 4001_F2B0h

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	data[127:96]																															
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

FMC_DATAW2S*n*UM field descriptions

Field	Description
31–0 data[127:96]	Bits [127:96] of data entry

29.4.17 Cache Data Storage (mid-upper word) (FMC_DATAW2SMU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW2S0MU is 4001_F000h base + 284h offset = 4001_F284h
FMC_DATAW2S1MU is 4001_F000h base + 294h offset = 4001_F294h
FMC_DATAW2S2MU is 4001_F000h base + 2A4h offset = 4001_F2A4h
FMC_DATAW2S3MU is 4001_F000h base + 2B4h offset = 4001_F2B4h

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	data[95:64]																															
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

FMC_DATAW2S*n*MU field descriptions

Field	Description
31–0 data[95:64]	Bits [95:64] of data entry

29.4.18 Cache Data Storage (mid-lower word) (FMC_DATAW2SML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW2S0ML is 4001_F000h base + 288h offset = 4001_F288h
FMC_DATAW2S1ML is 4001_F000h base + 298h offset = 4001_F298h
FMC_DATAW2S2ML is 4001_F000h base + 2A8h offset = 4001_F2A8h
FMC_DATAW2S3ML is 4001_F000h base + 2B8h offset = 4001_F2B8h

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	data[63:32]																															
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

FMC_DATAW2S*n*ML field descriptions

Field	Description
31–0 data[63:32]	Bits [63:32] of data entry

29.4.19 Cache Data Storage (lowermost word) (FMC_DATAW2SLM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW2S0LM is 4001_F000h base + 28Ch offset = 4001_F28Ch
FMC_DATAW2S1LM is 4001_F000h base + 29Ch offset = 4001_F29Ch
FMC_DATAW2S2LM is 4001_F000h base + 2ACh offset = 4001_F2ACh
FMC_DATAW2S3LM is 4001_F000h base + 2BCh offset = 4001_F2BCh

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	data[31:0]																															
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

FMC_DATAW2S*n*LM field descriptions

Field	Description
31–0 data[31:0]	Bits [31:0] of data entry

29.4.20 Cache Data Storage (uppermost word) (FMC_DATAW3SUM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the uppermost word (bits [127:96]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW3S0UM is 4001_F000h base + 2C0h offset = 4001_F2C0h
FMC_DATAW3S1UM is 4001_F000h base + 2D0h offset = 4001_F2D0h
FMC_DATAW3S2UM is 4001_F000h base + 2E0h offset = 4001_F2E0h
FMC_DATAW3S3UM is 4001_F000h base + 2F0h offset = 4001_F2F0h

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	data[127:96]																															
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

FMC_DATAW3S*n*UM field descriptions

Field	Description
31–0 data[127:96]	Bits [127:96] of data entry

29.4.21 Cache Data Storage (mid-upper word) (FMC_DATAW3SMU)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-upper word (bits [95:64]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW3S0MU is 4001_F000h base + 2C4h offset = 4001_F2C4h
FMC_DATAW3S1MU is 4001_F000h base + 2D4h offset = 4001_F2D4h
FMC_DATAW3S2MU is 4001_F000h base + 2E4h offset = 4001_F2E4h
FMC_DATAW3S3MU is 4001_F000h base + 2F4h offset = 4001_F2F4h

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	data[95:64]																															
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

FMC_DATAW3S*n*MU field descriptions

Field	Description
31–0 data[95:64]	Bits [95:64] of data entry

29.4.22 Cache Data Storage (mid-lower word) (FMC_DATAW3SML)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the mid-lower word (bits [63:32]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW3S0ML is 4001_F000h base + 2C8h offset = 4001_F2C8h
FMC_DATAW3S1ML is 4001_F000h base + 2D8h offset = 4001_F2D8h
FMC_DATAW3S2ML is 4001_F000h base + 2E8h offset = 4001_F2E8h
FMC_DATAW3S3ML is 4001_F000h base + 2F8h offset = 4001_F2F8h

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	data[63:32]																															
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

FMC_DATAW3S*n*ML field descriptions

Field	Description
31–0 data[63:32]	Bits [63:32] of data entry

29.4.23 Cache Data Storage (lowermost word) (FMC_DATAW3SLM)

The cache of sixteen 128-bit entries is a 4-way, set-associative cache with 4 sets. The ways are numbered 0-3 and the sets are numbered 0-3. In DATAWxSyUM, DATAWxSyMU, DATAWxSyML, and DATAWxSyLM, x denotes the way, y denotes the set, and the final two letters identify the word: UM (uppermost), MU (mid-upper), ML (mid-lower), and LM (lowermost). This section represents data for the lowermost word (bits [31:0]) of all 4 sets (n=0-3) in way 0.

Addresses: FMC_DATAW3S0LM is 4001_F000h base + 2CCh offset = 4001_F2CCh
FMC_DATAW3S1LM is 4001_F000h base + 2DCh offset = 4001_F2DCh
FMC_DATAW3S2LM is 4001_F000h base + 2ECh offset = 4001_F2ECh
FMC_DATAW3S3LM is 4001_F000h base + 2FCh offset = 4001_F2FCh

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	data[31:0]																															
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

FMC_DATAW3S*n*LM field descriptions

Field	Description
31–0 data[31:0]	Bits [31:0] of data entry

29.5 Functional description

The FMC is a flash acceleration unit with flexible buffers for user configuration. Besides managing the interface between the device and the flash memory and FlexMemory, the FMC can be used to restrict access from crossbar switch masters and customize the cache and buffers to provide single-cycle system-clock data-access times. Whenever a hit occurs for the prefetch speculation buffer, the cache, or the single-entry buffer, the requested data is transferred within a single system clock.

Upon system reset, the FMC is configured to provide a significant level of buffering for transfers from the flash memory or FlexMemory:

- Crossbar masters 0-2 have read access to banks 0-3.
- Crossbar masters 0-2 have write access to a portion of banks 2-3 when FlexNVM is used with FlexRAM as EEPROM.
- Prefetch support for data and instructions is enabled for crossbar masters 0-2.

- The cache is configured for least recently used (LRU) replacement for all four ways.
- The cache is configured for data or instruction replacement.
- The single-entry buffer is enabled.

Though the default configuration provides a high degree of flash acceleration, advanced users may desire to customize the FMC buffer configurations to maximize throughput for their use cases. When reconfiguring the FMC for custom use cases, do not program the FMC's control registers while the flash memory or FlexMemory is being accessed. Instead, change the control registers with a routine executing from RAM in supervisor mode.

The FMC's cache and buffering controls within PFB01CR and PFB23CR allow the tuning of resources to suit particular applications' needs. The cache and two buffers are each controlled individually. The register controls enable buffering and prefetching per memory bank pair (banks 0-1 and 2-3) and access type (instruction fetch or data reference). The cache also supports three types of LRU replacement algorithms:

- LRU per set across all four ways,
- LRU with ways [0-1] for instruction fetches and ways [2-3] for data fetches, and
- LRU with ways [0-2] for instruction fetches and way [3] for data fetches.

As an application example: if both instruction fetches and data references are accessing banks 0-1, control is available to send instruction fetches, data references, or both to the cache or the single-entry buffer. Likewise, speculation can be enabled or disabled for either type of access. If both instruction fetches and data references are cached, the cache's way resources may be divided in several ways between the instruction fetches and data references.

In another application example, the cache can be configured for replacement from banks 0-1, while the single-entry buffer can be enabled for banks 2-3 only. This configuration is ideal for applications that use banks 0-1 for program space and banks 2-3 for data space.

Chapter 30
Flash Memory Module (FTFE)

30.1 Introduction

NOTE

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

The FTFE module includes the following accessible memory regions:

- Program flash memory for vector space and code store
- For FlexNVM devices: FlexNVM for data store and additional code store
- For FlexNVM devices: FlexRAM for high-endurance data store or traditional RAM
- For program flash only devices: Programming acceleration RAM to speed flash programming

Flash memory is ideal for single-supply applications, permitting in-the-field erase and reprogramming operations without the need for any external high voltage power sources.

The FTFE module includes a memory controller that executes commands to modify flash memory contents. An erased bit reads '1' and a programmed bit reads '0'. The programming operation is unidirectional; it can only move bits from the '1' state (erased) to the '0' state (programmed). Only the erase operation restores bits from '0' to '1'; bits cannot be programmed from a '0' to a '1'.

CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

The standard shipping condition for flash memory is erased with security disabled. Data loss over time may occur due to degradation of the erased ('1') states and/or programmed ('0') states. Therefore, it is recommended that each flash block or sector be re-erased immediately prior to factory programming to ensure that the full data retention capability is achieved.

30.1.1 Features

The FTFE module includes the following features.

NOTE

See the device's Chip Configuration details for the exact amount of flash memory available on your device.

30.1.1.1 Program Flash Memory Features

- Sector size of 4 Kbytes
- Program flash protection scheme prevents accidental program or erase of stored data
- Automated, built-in, program and erase algorithms with verify
- Section programming for faster bulk programming times
- For devices containing only program flash memory: Read access to one program flash block is possible while programming or erasing data in another program flash block
- For devices containing FlexNVM memory: Read access to one program flash block is possible while programming or erasing data in another program flash block, data flash block, or FlexRAM

30.1.1.2 FlexNVM memory features

When FlexNVM is partitioned for data flash memory (on devices that contain FlexNVM memory):

- Sector size of 4 Kbytes
- Protection scheme prevents accidental program or erase of stored data

- Automated, built-in program and erase algorithms with verify
- Section programming for faster bulk programming times
- Read access to one data flash block possible while programming or erasing data in another program flash block or data flash block

30.1.1.3 Program acceleration RAM features

- For devices with only program flash memory: RAM to support section programming

30.1.1.4 FlexRAM features

For devices with FlexNVM memory:

- Memory that can be used as traditional RAM or as high-endurance EEPROM storage
- Up to 16 Kbytes of FlexRAM configured for EEPROM or traditional RAM operations
- When configured for EEPROM:
  - Protection scheme prevents accidental program or erase of data written for EEPROM
  - Built-in hardware emulation scheme to automate EEPROM record maintenance functions
  - Programmable EEPROM data set size and FlexNVM partition code facilitating EEPROM memory endurance trade-offs
  - Supports FlexRAM aligned writes of 1, 2, or 4 bytes at a time
  - Read access to FlexRAM possible while programming or erasing data in the program or data flash memory
- When configured for traditional RAM:
  - Read and write access possible to the FlexRAM while programming or erasing data in the program or data flash memory

30.1.1.5 Other FTFE module features

- Internal high-voltage supply generator for flash memory program and erase operations
- Optional interrupt generation upon flash command completion
- Supports MCU security mechanisms which prevent unauthorized access to the flash memory contents

30.1.2 Block diagram

The block diagram of the FTFE module is shown in the following figure.

For devices with FlexNVM feature:

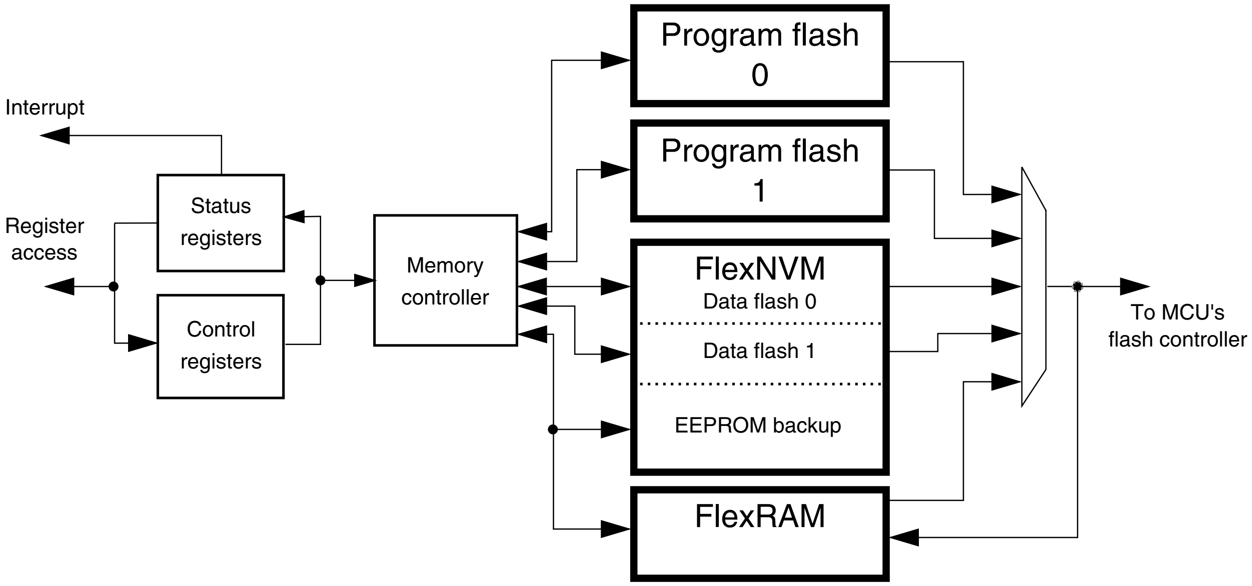


Figure 30-1. FTFE block diagram

For devices that contain only program flash:

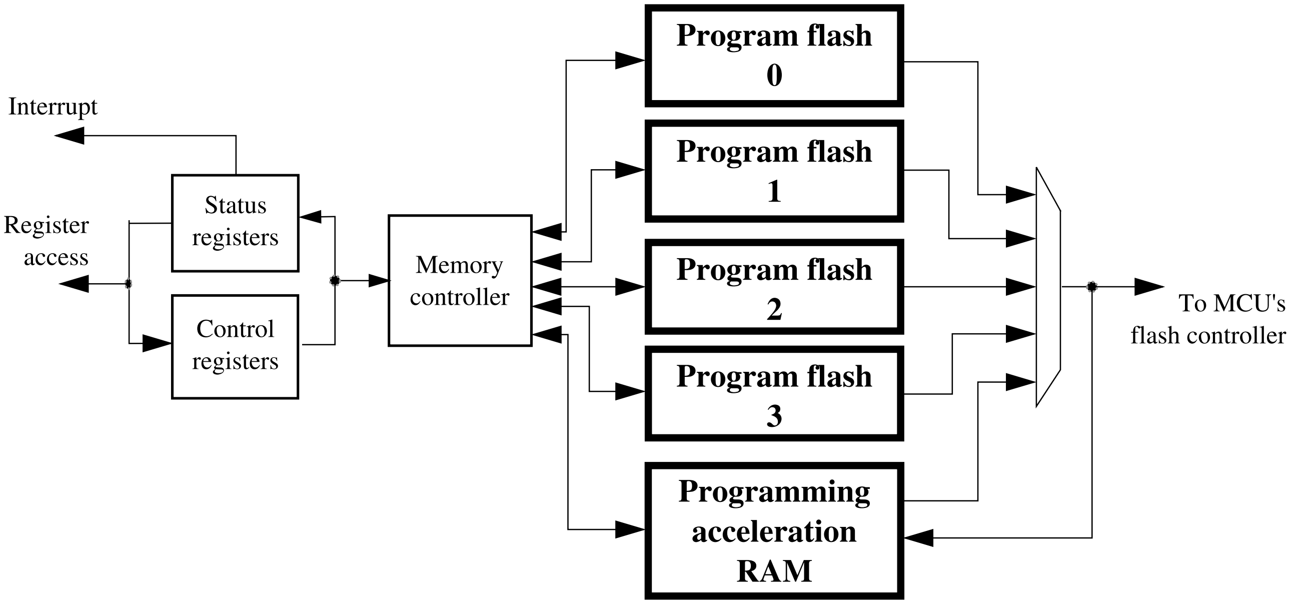


Figure 30-2. FTFE block diagram

30.1.3 Glossary

Command write sequence — A series of MCU writes to the Flash FCCOB register group that initiates and controls the execution of Flash algorithms that are built into the FTFE module.

Data flash memory — Partitioned from the FlexNVM block, the data flash memory provides nonvolatile storage for user data, boot code, and additional code store.

Data flash sector — The data flash sector is the smallest portion of the data flash memory that can be erased.

EEPROM — Using a built-in filing system, the FTFE module emulates the characteristics of an EEPROM by effectively providing a high-endurance, byte-writeable (program and erase) NVM.

EEPROM backup data header — The EEPROM backup data header is comprised of a 64-bit field found in EEPROM backup data memory which contains information used by the EEPROM filing system to determine the status of a specific EEPROM backup flash sector.

EEPROM backup data record — The EEPROM backup data record is comprised of a 7-bit status field, a 13-bit address field, and a 32-bit data field found in EEPROM backup data memory which is used by the EEPROM filing system. If the status field indicates a record is valid, the data field is mirrored in the FlexRAM at a location determined by the address field.

EEPROM backup data memory — Partitioned from the FlexNVM block, EEPROM backup data memory provides nonvolatile storage for the EEPROM filing system representing data written to the FlexRAM requiring highest endurance.

EEPROM backup data sector — The EEPROM backup data sector contains one EEPROM header and up to 255 EEPROM backup data records, which are used by the EEPROM filing system.

Endurance — The number of times that a flash memory location can be erased and reprogrammed.

FCCOB (Flash Common Command Object) — A group of flash registers that are used to pass command, address, data, and any associated parameters to the memory controller in the FTFE module.

Flash block — A macro within the FTFE module which provides the nonvolatile memory storage.

FlexMemory — FTFE configuration that supports data flash, EEPROM, and FlexRAM.

FlexNVM Block — The FlexNVM block can be configured to be used as data flash memory, EEPROM backup flash memory, or a combination of both.

FlexRAM — The FlexRAM refers to a RAM, dedicated to the FTFE module, that can be configured to store EEPROM data or as traditional RAM. When configured for EEPROM, valid writes to the FlexRAM generates a new EEPROM backup data record stored in the EEPROM backup flash memory.

FTFE Module — All flash blocks plus an FMU and an interface to MCU buses.

IFR — Nonvolatile information register found in each flash block, separate from the main memory array.

NVM — Nonvolatile memory. A memory technology that maintains stored data during power-off. The flash array is an NVM using NOR-type flash memory technology.

NVM Normal Mode — An NVM mode that provides basic user access to FTFE resources. The CPU or other bus masters initiate flash program and erase operations (or other flash commands) using writes to the FCCOB register group in the FTFE module.

NVM Special Mode — An NVM mode enabling external, off-chip access to the memory resources in the FTFE module. A reduced flash command set is available when the MCU is secured. See the Chip Configuration details for information on when this mode is used.

Double-Phrase — 128 bits of data with an aligned double-phrase having byte-address[3:0] = 0000.

Phrase — 64 bits of data with an aligned phrase having byte-address[2:0] = 000.

Longword — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

Word — 16 bits of data with an aligned word having byte-address[0] = 0.

Program flash — The program flash memory provides nonvolatile storage for vectors and code store.

Program flash sector — The smallest portion of the program flash memory (consecutive addresses) that can be erased.

Retention — The length of time that data can be kept in the NVM without experiencing errors upon readout. Since erased (1) states are subject to degradation just like programmed (0) states, the data retention limit may be reached from the last erase operation (not from the programming time).

RWW— Read-While-Write. The ability to simultaneously read from one memory resource while commanded operations are active in another memory resource.

Section program buffer — Lower half of the programming acceleration FlexRAM allocated for storing large amounts of data for programming via the Program Section command.

Secure — An MCU state conveyed to the FTFE module as described in the Chip Configuration details for this device. In the secure state, reading and changing NVM contents is restricted.

30.2 External signal description

The FTFE module contains no signals that connect off-chip.

30.3 Memory map and registers

This section describes the memory map and registers for the FTFE module. Data read from unimplemented memory space in the FTFE module is undefined. Writes to unimplemented or reserved memory space (registers) in the FTFE module are ignored.

30.3.1 Flash configuration field description

The program flash memory contains a 16-byte flash configuration field that stores default protection settings (loaded on reset) and security information that allows the MCU to restrict access to the FTFE module.

Flash Configuration Field Byte Address	Size (Bytes)	Field Description
0x0_0400 - 0x0_0407	8	Backdoor Comparison Key. Refer to Verify Backdoor Access Key command and Unsecuring the MCU Using Backdoor Key Access.
0x0_0408 - 0x0_040B	4	Program flash protection bytes. Refer to the description of the Program Flash Protection Registers (FPROT0-3).
0x0_040F	1	Program flash only devices: Reserved FlexNVM devices: Data flash protection byte. Refer to the description of the Data Flash Protection Register (FDPROT).
0x0_040E	1	Program flash only devices: Reserved FlexNVM devices: EEPROM protection byte. Refer to the description of the EEPROM Protection Register (FEPROT).
0x0_040D	1	Flash nonvolatile option byte. Refer to the description of the Flash Option Register (FOPT).
0x0_040C	1	Flash security byte. Refer to the description of the Flash Security Register (FSEC).

30.3.2 Program flash 0 IFR map

The program flash 0 IFR is a 1 Kbyte nonvolatile information memory that can be read freely, but the user has no erase and limited program capabilities (see the Read Once, Program Once, and Read Resource commands in Read Once Command, Program Once command and Read Resource Command). The contents of the program flash 0 IFR are summarized in the following table and further described in the subsequent paragraphs.

The program flash 0 IFR is located within the program flash 0 memory block.

Address Range	Size (Bytes)	Field Description
0x000 – 0x3BF	960	Reserved
0x3C0 – 03xFF	64	Program Once Field

30.3.2.1 Program Once field

The Program Once field in the program flash 0 IFR provides 64 bytes of user data storage separate from the program flash 0 main array. The user can program the Program Once field one time only as there is no program flash IFR erase mechanism available to the user. The Program Once field can be read any number of times. This section of the program flash 0 IFR is accessed in 8 byte records using the Read Once and Program Once commands (see Read Once Command and Program Once command).

30.3.3 Data flash 0 IFR map

The following only applies to devices with FlexNVM.

The data flash 0 IFR is a 1 Kbyte nonvolatile information memory that can be read and erased, but the user has limited program capabilities in the data flash 0 IFR (see the Program Partition command in Program Partition command, the Erase All Blocks command in Erase All Blocks Command, and the Read Resource command in Read Resource Command). The contents of the data flash 0 IFR are summarized in the following table and further described in the subsequent paragraphs.

The data flash 0 IFR is located within the data flash 0 memory block.

Address Range	Size (Bytes)	Field Description
0x00 – 0x3FB, 0x3FE – 0x3FF	1022	Reserved
0x3FD	1	EEPROM Data Set Size
0x3FC	1	FlexNVM Partition Code

30.3.3.1 EEPROM Data Set Size

The EEPROM data set size byte in the data flash IFR supplies information which determines the amount of FlexRAM used in each of the available EEPROM subsystems. To program the EEESPLIT and EEESIZE values, see the Program Partition command described in Program Partition command.

Table 30-1. EEPROM Data Set Size

Data flash IFR: 0x03FD							
7	6	5	4	3	2	1	0
1	1	EEESPLIT		EEESIZE			
▨	= Unimplemented or Reserved						

Table 30-2. EEPROM Data Set Size Field Description

Field	Description
7-6 Reserved	This read-only bitfield is reserved and must always be written as one.

Table continues on the next page...

Table 30-2. EEPROM Data Set Size Field Description (continued)

Field	Description
5-4 EEESPLIT	**EEPROM Split Factor** — Determines the relative sizes of the two EEPROM subsystems. Each subsystem is allocated half of the available EEPROM-backup as defined by DEPART. ‘00’ = Subsystem A: EEESIZE*1/8, subsystem B: EEESIZE*7/8 ‘01’ = Subsystem A: EEESIZE*1/4, subsystem B: EEESIZE*3/4 ‘10’ = Subsystem A: EEESIZE*1/2, subsystem B: EEESIZE*1/2 ‘11’ = Subsystem A: EEESIZE*1/2, subsystem B: EEESIZE*1/2
3-0 EEESIZE	**EEPROM Size** — Encoding of the total available FlexRAM for EEPROM use. **NOTE:** EEESIZE must be 0 bytes (1111b) when the FlexNVM partition code (FlexNVM partition code) is set to 'No EEPROM'. '0000' = 16,384 Bytes '0001' = 8,192 Bytes '0010' = 4,096 Bytes '0011' = 2,048 Bytes '0100' = 1,024 Bytes '0101' = 512 Bytes '0110' = 256 Bytes '0111' = 128 Bytes '1000' = 64 Bytes '1001' = 32 Bytes '1010' = Reserved '1011' = Reserved '1100' = Reserved '1101' = Reserved '1110' = Reserved '1111' = 0 Bytes

30.3.3.2 FlexNVM partition code

The FlexNVM partition code byte in the data flash 0 IFR supplies a code which specifies how to split the FlexNVM block between data flash memory and EEPROM backup memory supporting EEPROM functions. To program the DEPART value, see the Program Partition command described in Program Partition command.

Table 30-3. FlexNVM partition code

Data Flash IFR: 0x03FC							
7	6	5	4	3	2	1	0
1	1	1	1	DEPART			

Table continues on the next page...

Table 30-3. FlexNVM partition code (continued)

▨	= Unimplemented or Reserved

Table 30-4. FlexNVM partition code field description

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>7-4
Reserved</td><td>This read-only bitfield is reserved and must always be written as one.</td></tr>
<tr><td>3-0
DEPART</td><td><b>FlexNVM Partition Code</b> — Encoding of the data flash / EEPROM backup split within the FlexNVM memory block. FlexNVM memory not partitioned for data flash is used to store EEPROM records.
<table>
<tr><th>DEPART</th><th>Data flash (KByte)</th><th>EEPROM backup (KByte)</th></tr>
<tr><td>0000</td><td>512</td><td>0</td></tr>
<tr><td>0001</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0010</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0011</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>0100</td><td>448</td><td>64</td></tr>
<tr><td>0101</td><td>384</td><td>128</td></tr>
<tr><td>0110</td><td>256</td><td>256</td></tr>
<tr><td>0111</td><td>0</td><td>512</td></tr>
<tr><td>1000</td><td>0</td><td>512</td></tr>
<tr><td>1001</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>1010</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>1011</td><td>Reserved</td><td>Reserved</td></tr>
<tr><td>1100</td><td>64</td><td>448</td></tr>
<tr><td>1101</td><td>128</td><td>384</td></tr>
<tr><td>1110</td><td>256</td><td>256</td></tr>
<tr><td>1111</td><td>512</td><td>0</td></tr>
</table></td></tr>
</table>

30.3.4 Register descriptions

The FTFE module contains a set of memory-mapped control and status registers.

NOTE

While a command is running (FSTAT[CCIF]=0), register writes are not accepted to any register except FCNFG and FSTAT. The no-write rule is relaxed during the start-up reset sequence, prior to the initial rise of CCIF. During this initialization period the user may write any register. All register writes are also disabled (except for registers FCNFG and

FSTAT) whenever an erase suspend request is active (FCNFG[ERSSUSP]=1).

FTFE memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0000	Flash Status Register (FTFE_FSTAT)	8	R/W	00h	30.34.1/ 743
4002_0001	Flash Configuration Register (FTFE_FCNFG)	8	R/W	00h	30.34.2/ 744
4002_0002	Flash Security Register (FTFE_FSEC)	8	R	Undefined	30.34.3/ 747
4002_0003	Flash Option Register (FTFE_FOPT)	8	R	Undefined	30.34.4/ 748
4002_0004	Flash Common Command Object Registers (FTFE_FCCOB3)	8	R/W	00h	30.34.5/ 749
4002_0005	Flash Common Command Object Registers (FTFE_FCCOB2)	8	R/W	00h	30.34.5/ 749
4002_0006	Flash Common Command Object Registers (FTFE_FCCOB1)	8	R/W	00h	30.34.5/ 749
4002_0007	Flash Common Command Object Registers (FTFE_FCCOB0)	8	R/W	00h	30.34.5/ 749
4002_0008	Flash Common Command Object Registers (FTFE_FCCOB7)	8	R/W	00h	30.34.5/ 749
4002_0009	Flash Common Command Object Registers (FTFE_FCCOB6)	8	R/W	00h	30.34.5/ 749
4002_000A	Flash Common Command Object Registers (FTFE_FCCOB5)	8	R/W	00h	30.34.5/ 749
4002_000B	Flash Common Command Object Registers (FTFE_FCCOB4)	8	R/W	00h	30.34.5/ 749
4002_000C	Flash Common Command Object Registers (FTFE_FCCOBB)	8	R/W	00h	30.34.5/ 749
4002_000D	Flash Common Command Object Registers (FTFE_FCCOBA)	8	R/W	00h	30.34.5/ 749
4002_000E	Flash Common Command Object Registers (FTFE_FCCOB9)	8	R/W	00h	30.34.5/ 749
4002_000F	Flash Common Command Object Registers (FTFE_FCCOB8)	8	R/W	00h	30.34.5/ 749
4002_0010	Program Flash Protection Registers (FTFE_FPROT3)	8	R/W	Undefined	30.34.6/ 750
4002_0011	Program Flash Protection Registers (FTFE_FPROT2)	8	R/W	Undefined	30.34.6/ 750

Table continues on the next page...

FTFE memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_0012	Program Flash Protection Registers (FTFE_FPROT1)	8	R/W	Undefined	30.34.6/ 750
4002_0013	Program Flash Protection Registers (FTFE_FPROT0)	8	R/W	Undefined	30.34.6/ 750
4002_0016	EEPROM Protection Register (FTFE_FEPROT)	8	R/W	Undefined	30.34.7/ 751
4002_0017	Data Flash Protection Register (FTFE_FDPROT)	8	R/W	Undefined	30.34.8/ 753

30.34.1 Flash Status Register (FTFE_FSTAT)

The FSTAT register reports the operational status of the FTFE module.

The CCIF, RDCOLERR, ACCERR, and FPVIOL bits are readable and writable. The MGSTAT0 bit is read only. The unassigned bits read 0 and are not writable.

NOTE

> When set, the Access Error (ACCERR) and Flash Protection Violation (FPVIOL) bits in this register prevent the launch of any more commands or writes to the FlexRAM (when EEERDY is set) until the flag is cleared (by writing a one to it).

Address: FTFE_FSTAT is 4002_0000h base + 0h offset = 4002_0000h

Bit	7	6	5	4	3	2	1	0
Read	CCIF	RDCOLERR	ACCERR	FPVIOL	0			MGSTAT0
Write	w1c	w1c	w1c	w1c				
Reset	0	0	0	0	0	0	0	0

FTFE_FSTAT field descriptions

Field	Description
7 CCIF	Command Complete Interrupt Flag The CCIF flag indicates that a FTFE command or EEPROM file system operation has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command, and CCIF stays low until command completion or command violation. The CCIF flag is also cleared by a successful write to FlexRAM while enabled for EEE, and CCIF stays low until the EEPROM file system has created the associated EEPROM data record. The CCIF bit is reset to 0 but is set to 1 by the memory controller at the end of the reset initialization sequence. Depending on how quickly the read occurs after reset release, the user may or may not see the 0 hardware reset value.

Table continues on the next page...

FTFE_FSTAT field descriptions (continued)

Field	Description
	0 FTFE command or EEPROM file system operation in progress 1 FTFE command or EEPROM file system operation has completed
6 RDCOLERR	FTFE Read Collision Error Flag The RDCOLERR error bit indicates that the MCU attempted a read from an FTFE resource that was being manipulated by an FTFE command (CCIF=0). Any simultaneous access is detected as a collision error by the block arbitration logic. The read data in this case cannot be guaranteed. The RDCOLERR bit is cleared by writing a 1 to it. Writing a 0 to RDCOLERR has no effect. 0 No collision error detected 1 Collision error detected
5 ACCERR	Flash Access Error Flag The ACCERR error bit indicates an illegal access has occurred to an FTFE resource caused by a violation of the command write sequence or issuing an illegal FTFE command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to it. Writing a 0 to the ACCERR bit has no effect. 0 No access error detected 1 Access error detected
4 FPVIOL	Flash Protection Violation Flag The FPVIOL error bit indicates an attempt was made to program or erase an address in a protected area of program flash or data flash memory during a command write sequence or a write was attempted to a protected area of the FlexRAM while enabled for EEPROM. While FPVIOL is set, the CCIF flag cannot be cleared to launch a command. The FPVIOL bit is cleared by writing a 1 to it. Writing a 0 to the FPVIOL bit has no effect. 0 No protection violation detected 1 Protection violation detected
3–1 Reserved	This read-only field is reserved and always has the value zero.
0 MGSTAT0	Memory Controller Command Completion Status Flag The MGSTAT0 status flag is set if an error is detected during execution of an FTFE command or during the flash reset sequence. As a status flag, this bit cannot (and need not) be cleared by the user like the other error flags in this register. The value of the MGSTAT0 bit for "command-N" is valid only at the end of the "command-N" execution when CCIF=1 and before the next command has been launched. At some point during the execution of "command-N+1," the previous result is discarded and any previous error is cleared.

30.34.2 Flash Configuration Register (FTFE_FCNFG)

This register provides information on the current functional state of the FTFE module.

The erase control bits (ERSAREQ and ERSSUSP) have write restrictions. SWAP, PFLSH, RAMRDY, and EEERDY are read-only status bits. The unassigned bits read as noted and are not writable. The reset values for the SWAP, PFLSH, RAMRDY, and EEERDY bits are determined during the reset sequence.

Address: FTFE_FCNFG is 4002_0000h base + 1h offset = 4002_0001h

Bit	7	6	5	4	3	2	1	0
Read	CCIE	RDCOLLIE	ERSAREQ	ERSSUSP	SWAP	PFLSH	RAMRDY	EEERDY
Write								
Reset	0	0	0	0	0	0	0	0

FTFE_FCNFG field descriptions

Field	Description
7 CCIE	Command Complete Interrupt Enable The CCIE bit controls interrupt generation when an FTFE command completes. 0 Command complete interrupt disabled 1 Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.
6 RDCOLLIE	Read Collision Error Interrupt Enable The RDCOLLIE bit controls interrupt generation when an FTFE read collision error occurs. 0 Read collision error interrupt disabled 1 Read collision error interrupt enabled. An interrupt request is generated whenever an FTFE read collision error is detected (see the description of FSTAT[RDCOLERR]).
5 ERSAREQ	Erase All Request This bit issues a request to the memory controller to execute the Erase All Blocks command and release security. ERSAREQ is not directly writable but is under indirect user control. Refer to the device's Chip Configuration details on how to request this command. The ERSAREQ bit sets when an erase all request is triggered external to the FTFE and CCIF is set (no command is currently being executed). ERSAREQ is cleared by the FTFE when the operation completes. 0 No request or request complete 1 Request to: 1. run the Erase All Blocks command, 2. verify the erased state, 3. program the security byte in the Flash Configuration Field to the unsecure state, and 4. release MCU security by setting the FSEC[SEC] field to the unsecure state.
4 ERSSUSP	Erase Suspend The ERSSUSP bit allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing. 0 No suspend requested 1 Suspend the current Erase Flash Sector command execution.
3 SWAP	Swap

Table continues on the next page...

FTFE_FCNFG field descriptions (continued)

Field	Description
	The SWAP flag indicates which half of the program flash space is located at relative address 0x0000. The state of the SWAP flag is set by the FTFE during the reset sequence. See Swap Control command for information on swap management. 0 For devices with FlexNVM: Logical program flash 0 block is located at relative address 0x0000 For devices with program flash only: Logical program flash 0/1 blocks are located at relative address 0x0000 1 For devices with FlexNVM: Logical program flash 1 block is located at relative address 0x0000 For devices with program flash only: Logical program flash 2/3 blocks are located at relative address 0x0000
2 PFLSH	FTFE configuration 0 For devices with FlexNVM: FTFE configuration supports two logical program flash blocks and two logical FlexNVM blocks For devices with program flash only: Reserved 1 For devices with FlexNVM: Reserved For devices with program flash only: FTFE configuration supports four logical program flash blocks
1 RAMRDY	RAM Ready This flag indicates the current status of the FlexRAM/programming acceleration RAM. For devices with FlexNVM: The state of the RAMRDY flag is normally controlled by the Set FlexRAM Function command. During the reset sequence, the RAMRDY flag is cleared if the FlexNVM block is partitioned for EEPROM and is set if the FlexNVM block is not partitioned for EEPROM. The RAMRDY flag is cleared if the Program Partition command is run to partition the FlexNVM block for EEPROM. The RAMRDY flag sets after completion of the Erase All Blocks command or execution of the erase-all operation triggered external to the FTFE. For devices without FlexNVM: This bit should always be set. 0 For devices with FlexNVM: FlexRAM is not available for traditional RAM access. For devices without FlexNVM: Programming acceleration RAM is not available. 1 For devices with FlexNVM: FlexRAM is available as traditional RAM only; writes to the FlexRAM do not trigger EEPROM operations. For devices without FlexNVM: Programming acceleration RAM is available.
0 EEERDY	For devices with FlexNVM: This flag indicates if the EEPROM backup data has been copied to the FlexRAM and is therefore available for read access. During the reset sequence, the EEERDY flag remains clear while CCIF=0 and only sets if the FlexNVM block is partitioned for EEPROM. For devices without FlexNVM: This bit is reserved. 0 For devices with FlexNVM: FlexRAM is not available for EEPROM operation. 1 For devices with FlexNVM: FlexRAM is available for EEPROM operations where: • reads from the FlexRAM return data previously written to the FlexRAM in EEPROM mode and • writes launch an EEPROM operation to store the written data in the FlexRAM and EEPROM backup.

30.34.3 Flash Security Register (FTFE_FSEC)

This read-only register holds all bits associated with the security of the MCU and FTFE module.

During the reset sequence, the register is loaded with the contents of the flash security byte in the Flash Configuration Field located in program flash memory. The Flash basis for the values is signified by X in the reset value.

Address: FTFE_FSEC is 4002_0000h base + 2h offset = 4002_0002h

Bit	7	6	5	4	3	2	1	0
Read	KEYEN		MEEN		FSLACC		SEC	
Write	▨		▨		▨		▨	
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_FSEC field descriptions

Field	Description
7–6 KEYEN	Backdoor Key Security Enable These bits enable and disable backdoor key access to the FTFE module. 00 Backdoor key access disabled 01 Backdoor key access disabled (preferred KEYEN state to disable backdoor key access) 10 Backdoor key access enabled 11 Backdoor key access disabled
5–4 MEEN	Mass Erase Enable Bits Enables and disables mass erase capability of the FTFE module. The state of the MEEN bits is only relevant when the SEC bits are set to secure outside of NVM Normal Mode. When the SEC field is set to unsecure, the MEEN setting does not matter. 00 Mass erase is enabled 01 Mass erase is enabled 10 Mass erase is disabled 11 Mass erase is enabled
3–2 FSLACC	Freescale Failure Analysis Access Code These bits enable or disable access to the flash memory contents during returned part failure analysis at Freescale. When SEC is secure and FSLACC is denied, access to the program flash contents is denied and any failure analysis performed by Freescale factory test must begin with a full erase to unsecure the part. When access is granted (SEC is unsecure, or SEC is secure and FSLACC is granted), Freescale factory testing has visibility of the current flash contents. The state of the FSLACC bits is only relevant when the SEC bits are set to secure. When the SEC field is set to unsecure, the FSLACC setting does not matter.

Table continues on the next page...

FTFE_FSEC field descriptions (continued)

Field	Description
	00 Freescale factory access granted 01 Freescale factory access denied 10 Freescale factory access denied 11 Freescale factory access granted
1–0 SEC	Flash Security These bits define the security state of the MCU. In the secure state, the MCU limits access to FTFE module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the FTFE module is unsecured using backdoor key access, the SEC bits are forced to 10b. 00 MCU security status is secure 01 MCU security status is secure 10 MCU security status is unsecure (The standard shipping condition of the FTFE is unsecure.) 11 MCU security status is secure

30.34.4 Flash Option Register (FTFE_FOPT)

The flash option register allows the MCU to customize its operations by examining the state of these read-only bits, which are loaded from NVM at reset. The function of the bits is defined in the device's Chip Configuration details.

All bits in the register are read-only.

During the reset sequence, the register is loaded from the flash nonvolatile option byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: FTFE_FOPT is 4002_0000h base + 3h offset = 4002_0003h

Bit	7	6	5	4	3	2	1	0
Read	OPT							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

FTFE_FOPT field descriptions

Field	Description
7–0 OPT	Nonvolatile Option These bits are loaded from flash to this register at reset. Refer to the device's Chip Configuration details for the definition and use of these bits.

30.34.5 Flash Common Command Object Registers (FTFE_FCCOB*n*)

The FCCOB register group provides 12 bytes for command codes and parameters. The individual bytes within the set append a 0-B hex identifier to the FCCOB register name: FCCOB0, FCCOB1, ..., FCCOBB.

Addresses: 4002_0000h base + 4h offset + (1d × *n*), where *n* = 0d to 11d

Bit	7	6	5	4	3	2	1	0
Read	CCOBn							
Write								
Reset	0	0	0	0	0	0	0	0

FTFE_FCCOB*n* field descriptions

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>7–0
CCOBn</td><td>
<p>The FCCOB register provides a command code and relevant parameters to the memory controller. The individual registers that compose the FCCOB data set can be written in any order, but you must provide all needed values, which vary from command to command. First, set up all required FCCOB fields and then initiate the command's execution by writing a 1 to the FSTAT[CCIF] bit. This clears the CCIF bit, which locks all FCCOB parameter fields and they cannot be changed by the user until the command completes (CCIF returns to 1). No command buffering or queueing is provided; the next command can be loaded only after the current command completes.</p>
<p>Some commands return information to the FCCOB registers. Any values returned to FCCOB are available for reading after the FSTAT[CCIF] flag returns to 1 by the memory controller.</p>
<p>The following table shows a generic FTFE command format. The first FCCOB register, FCCOB0, always contains the command code. This 8-bit value defines the command to be executed. The command code is followed by the parameters required for this specific FTFE command, typically an address and/or data values.</p>
<p><b>NOTE:</b> The command parameter table is written in terms of FCCOB Number (which is equivalent to the byte number). This number is a reference to the FCCOB register name and is not the register address.</p>
<table>
<tr><th>FCCOB Number[1]</th><th>Typical Command Parameter Contents [7:0]</th></tr>
<tr><td>0</td><td>FCMD (a code that defines the FTFE command)</td></tr>
<tr><td>1</td><td>Flash address [23:16]</td></tr>
<tr><td>2</td><td>Flash address [15:8]</td></tr>
<tr><td>3</td><td>Flash address [7:0]</td></tr>
<tr><td>4</td><td>Data Byte 0</td></tr>
<tr><td>5</td><td>Data Byte 1</td></tr>
<tr><td>6</td><td>Data Byte 2</td></tr>
<tr><td>7</td><td>Data Byte 3</td></tr>
</table>
</td></tr>
</table>

FTFE_FCCOB*n* field descriptions (continued)

Field	Description
	<table><tr><th>FCCOB Number[1]</th><th>Typical Command Parameter Contents [7:0]</th></tr><tr><td>8</td><td>Data Byte 4</td></tr><tr><td>9</td><td>Data Byte 5</td></tr><tr><td>A</td><td>Data Byte 6</td></tr><tr><td>B</td><td>Data Byte 7</td></tr></table> 1. Refers to FCCOB register name, not register address **FCCOB Endianness and Multi-Byte Access:** The FCCOB register group uses a big endian addressing convention. For all command parameter fields larger than 1 byte, the most significant data resides in the lowest FCCOB register number. The FCCOB register group may be read and written as individual bytes, aligned words (2 bytes) or aligned longwords (4 bytes).

1. Refers to FCCOB register name, not register address

30.34.6 Program Flash Protection Registers (FTFE_FPROT*n*)

The FPROT registers define which logical program flash regions are protected from program and erase operations. Protected flash regions cannot have their content changed; that is, these regions cannot be programmed and cannot be erased by any FTFE command. Unprotected regions can be changed by program and erase operations.

The four FPROT registers allow 32 protectable regions. Each bit protects a 1/32 region of the program flash memory.

Program flash protection register	Program flash protection bits
FPROT0	PROT[31:24]
FPROT1	PROT[23:16]
FPROT2	PROT[15:8]
FPROT3	PROT[7:0]

During the reset sequence, the FPROT registers are loaded with the contents of the program flash protection bytes in the Flash Configuration Field as indicated in the following table.

Program flash protection register	Flash Configuration Field offset address
FPROT0	0x0008
FPROT1	0x0009
FPROT2	0x000A
FPROT3	0x000B

To change the program flash protection that is loaded during the reset sequence, unprotect the sector of program flash memory that contains the Flash Configuration Field. Then, reprogram the program flash protection byte.

Addresses: FTFE_FPROT3 is 4002_0000h base + 10h offset = 4002_0010h
FTFE_FPROT2 is 4002_0000h base + 11h offset = 4002_0011h
FTFE_FPROT1 is 4002_0000h base + 12h offset = 4002_0012h
FTFE_FPROT0 is 4002_0000h base + 13h offset = 4002_0013h

Bit	7	6	5	4	3	2	1	0
Read Write	PROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTFE_FPROT*n* field descriptions

Field	Description
7–0 PROT	Program Flash Region Protect Each program flash region can be protected from program and erase operations by setting the associated PROT bit. **In NVM Normal mode:** The protection can only be increased, meaning that currently unprotected memory can be protected, but currently protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **In NVM Special mode:** All bits of FPROT are writable without restriction. Unprotected areas can be protected and protected areas can be unprotected. **Restriction:** The user must never write to any FPROT register while a command is running (CCIF=0). Trying to alter data in any protected area in the program flash memory results in a protection violation error and sets the FSTAT[FPVIOL] bit. A full block erase of a program flash block is not possible if it contains any protected region. Each bit in the 32-bit protection register represents 1/32 of the total program flash. 0 Program flash region is protected. 1 Program flash region is not protected

30.34.7 EEPROM Protection Register (FTFE_FEPROT)

For devices with FlexNVM: The FEPROT register defines which EEPROM regions of the FlexRAM are protected against program and erase operations. Protected EEPROM regions cannot have their content changed by writing to it. Unprotected regions can be changed by writing to the FlexRAM.

For devices with program flash only: This register is reserved and not used.

Address: FTFE_FEPROT is 4002_0000h base + 16h offset = 4002_0016h

Bit	7	6	5	4	3	2	1	0
Read Write	EPROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

FTFE_FEPROT field descriptions

Field	Description
7–0 EPROT	EEPROM Region Protect For devices with program flash only: Reserved For devices with FlexNVM: Individual EEPROM regions can be protected from alteration by setting the associated EPROT bit. The EPROT bits are not used when the FlexNVM Partition Code is set to data flash only. When the FlexNVM Partition Code is set to data flash and EEPROM or EEPROM only, each EPROT bit covers one-eighth of the configured EEPROM data (see the EEPROM Data Set Size parameter description). **In NVM Normal mode:** The protection can only be increased. This means that currently-unprotected memory can be protected, but currently-protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FEPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **In NVM Special mode:** All bits of the FEPROT register are writable without restriction. Unprotected areas can be protected and protected areas can be unprotected. **Restriction:** Never write to the FEPROT register while a command is running (CCIF=0). **Reset:** During the reset sequence, the FEPROT register is loaded with the contents of the FlexRAM protection byte in the Flash Configuration Field located in program flash. The flash basis for the reset values is signified by X in the register diagram. To change the EEPROM protection that will be loaded during the reset sequence, the sector of program flash that contains the Flash Configuration Field must be unprotected; then the EEPROM protection byte must be erased and reprogrammed. Trying to alter data by writing to any protected area in the EEPROM results in a protection violation error and sets the FSTAT[FPVIOL] bit. 0 For devices with program flash only: Reserved For devices with FlexNVM: EEPROM region is protected 1 For devices with program flash only: Reserved For devices with FlexNVM: EEPROM region is not protected

30.34.8 Data Flash Protection Register (FTFE_FDPROT)

The FDPROT register defines which data flash regions are protected against program and erase operations. Protected Flash regions cannot have their content changed; that is, these regions cannot be programmed and cannot be erased by any FTFE command. Unprotected regions can be changed by both program and erase operations.

Address: FTFE_FDPROT is 4002_0000h base + 17h offset = 4002_0017h

Bit	7	6	5	4	3	2	1	0
Read Write	DPROT							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

FTFE_FDPROT field descriptions

Field	Description
7–0 DPROT	Data Flash Region Protect Individual data flash regions can be protected from program and erase operations by setting the associated DPROT bit. Each DPROT bit protects one-eighth of the partitioned data flash memory space. The granularity of data flash protection cannot be less than the data flash sector size. If an unused DPROT bit is set, the Erase all Blocks command does not execute and sets the FSTAT[FPVIOL] bit. **In NVM Normal mode:** The protection can only be increased, meaning that currently unprotected memory can be protected but currently protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FDPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored. **In NVM Special mode:** All bits of the FDPROT register are writable without restriction. Unprotected areas can be protected and protected areas can be unprotected. **Restriction:** The user must never write to the FDPROT register while a command is running (CCIF=0). **Reset:** During the reset sequence, the FDPROT register is loaded with the contents of the data flash protection byte in the Flash Configuration Field located in program flash memory. The flash basis for the reset values is signified by X in the register diagram. To change the data flash protection that will be loaded during the reset sequence, unprotect the sector of program flash that contains the Flash Configuration Field. Then, erase and reprogram the data flash protection byte. Trying to alter data with the program and erase commands in any protected area in the data flash memory results in a protection violation error and sets the FSTAT[FPVIOL] bit. A block erase of any data flash memory block (see the Erase Flash Block command description) is not possible if the data flash block contains any protected region or if the FlexNVM memory has been partitioned for EEPROM. 0 Data Flash region is protected 1 Data Flash region is not protected

30.4 Functional Description

The following sections describe functional details of the FTFE module.

30.4.1 Program flash memory swap

The user can configure the memory map of the program flash space such that either half of the program flash memory can exist at relative address 0x0000. This swap feature enables the lower half of the program flash space to be operational while the upper half is being updated for future use.

The Swap Control command handles swapping the two halves of program flash memory within the memory map. See Swap Control command for details.

30.4.2 Flash Protection

Individual regions within the flash memory can be protected from program and erase operations. Protection is controlled by the following registers:

- FPROT*n* — Four registers that protect 32 regions of the program flash memory as shown in the following figure

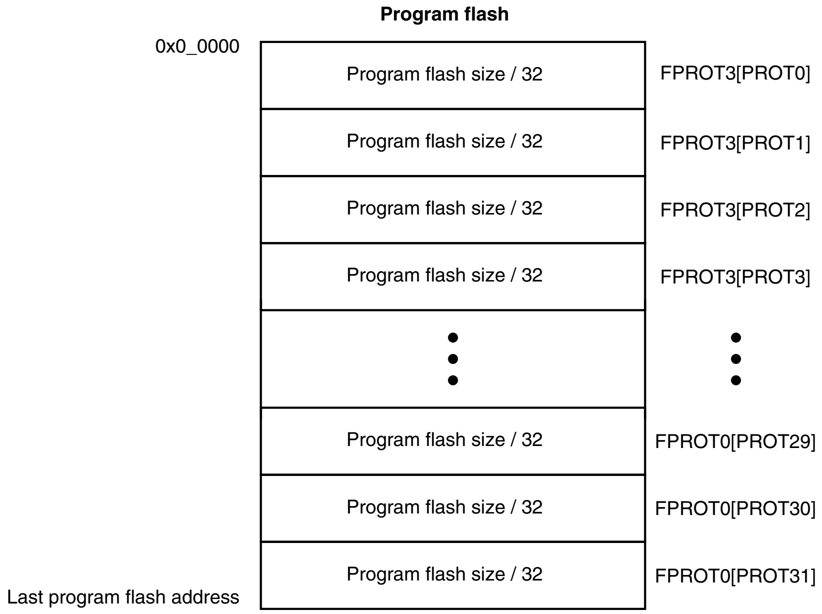


Figure 30-27. Program flash protection

- FDPROT —

- For 2^n data flash sizes, protects eight regions of the data flash memory as shown in the following figure

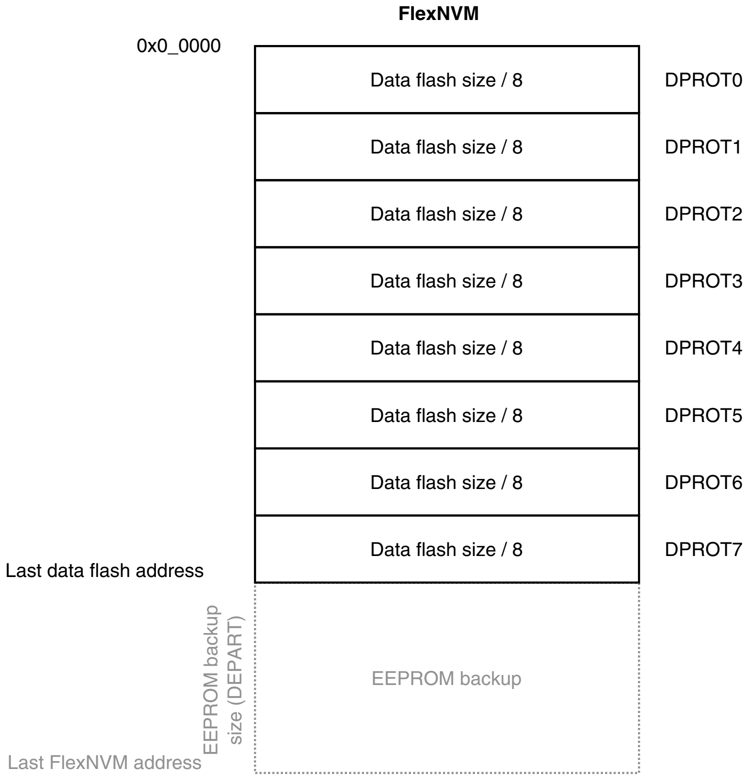


Figure 30-28. Data flash protection (2^n data flash sizes)

- For the non-2^n data flash sizes, the protection granularity is 64KB. Therefore, for 384KB data flash size, only the DPROT[5:0] bits are used, and for 448KB data flash size, only the DPROT[6:0] bits are used.

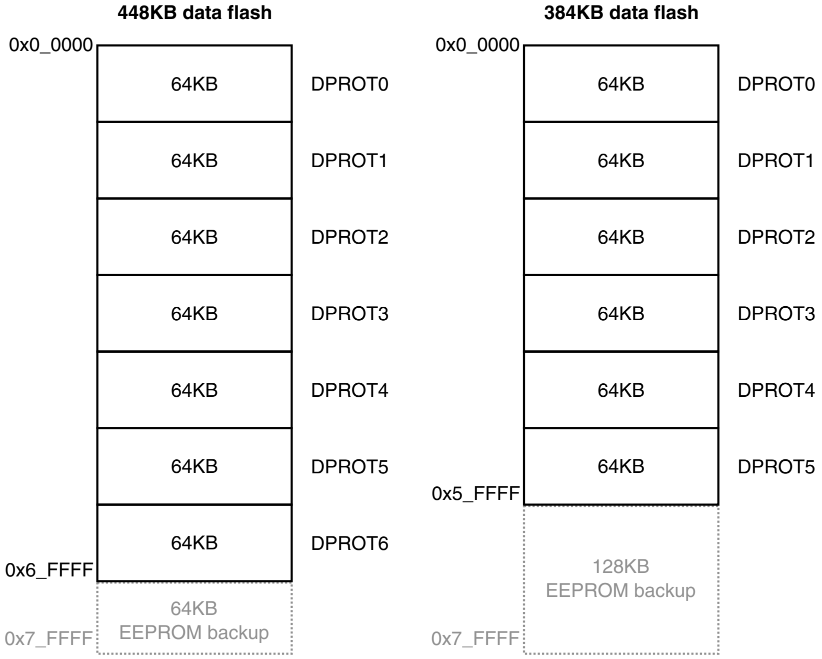


Figure 30-29. Data flash protection (384KB and 448KB data flash sizes)

- FEPROT — Protects eight regions of the EEPROM memory as shown in the following figure

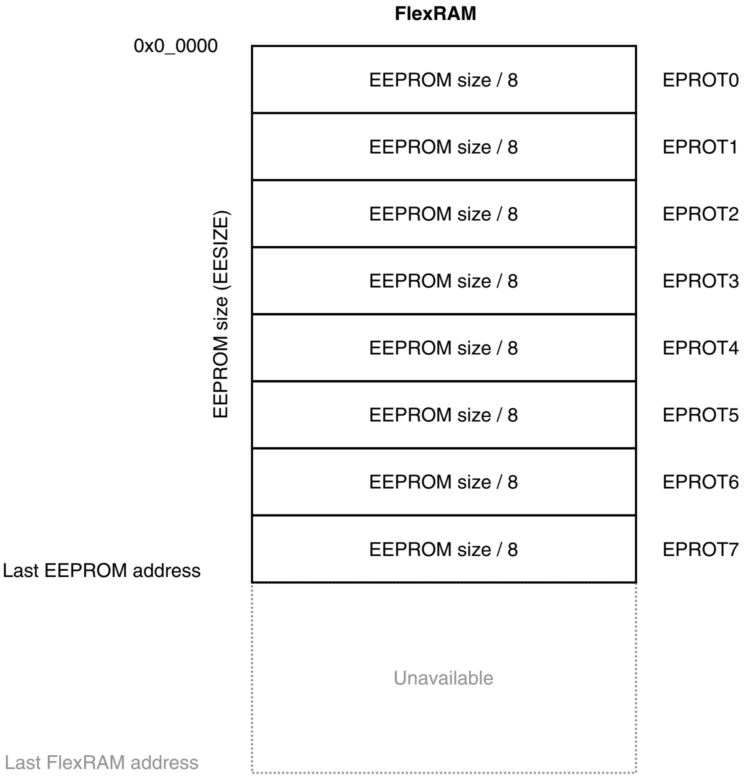


Figure 30-30. EEPROM protection

30.4.3 FlexNVM Description

This section describes the FlexNVM memory. This section does not apply for devices that contain only program flash memory.

30.4.3.1 FlexNVM Block Partitioning for FlexRAM

The user can configure the FlexNVM block as either:

- Basic data flash,
- EEPROM flash records to support the built-in EEPROM feature, or
- A combination of both.

The user's FlexNVM configuration choice is specified using the Program Partition command described in Program Partition command.

CAUTION

> While different partitions of the FlexNVM block are available, the intention is that a single partition choice is used throughout the entire lifetime of a given application. The FlexNVM partition code choices affect the endurance and data retention characteristics of the device.

30.4.3.2 EEPROM User Perspective

The EEPROM system is shown in the following figure.

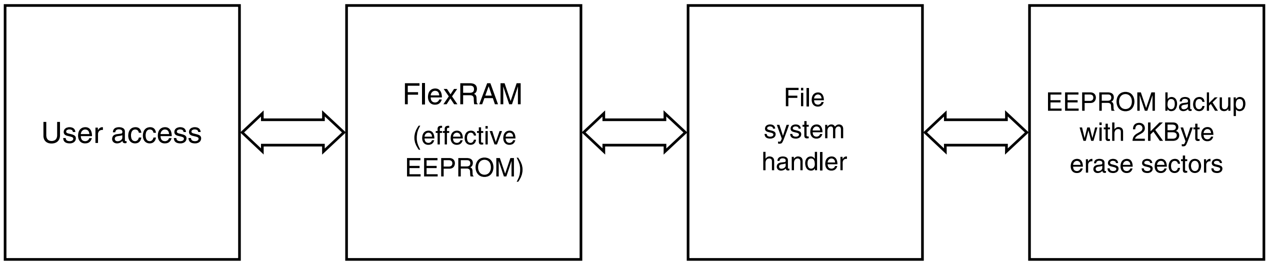


Figure 30-31. Top Level EEPROM Architecture

To handle varying customer requirements, the FlexRAM and FlexNVM blocks can be split into partitions as shown in the figure below.

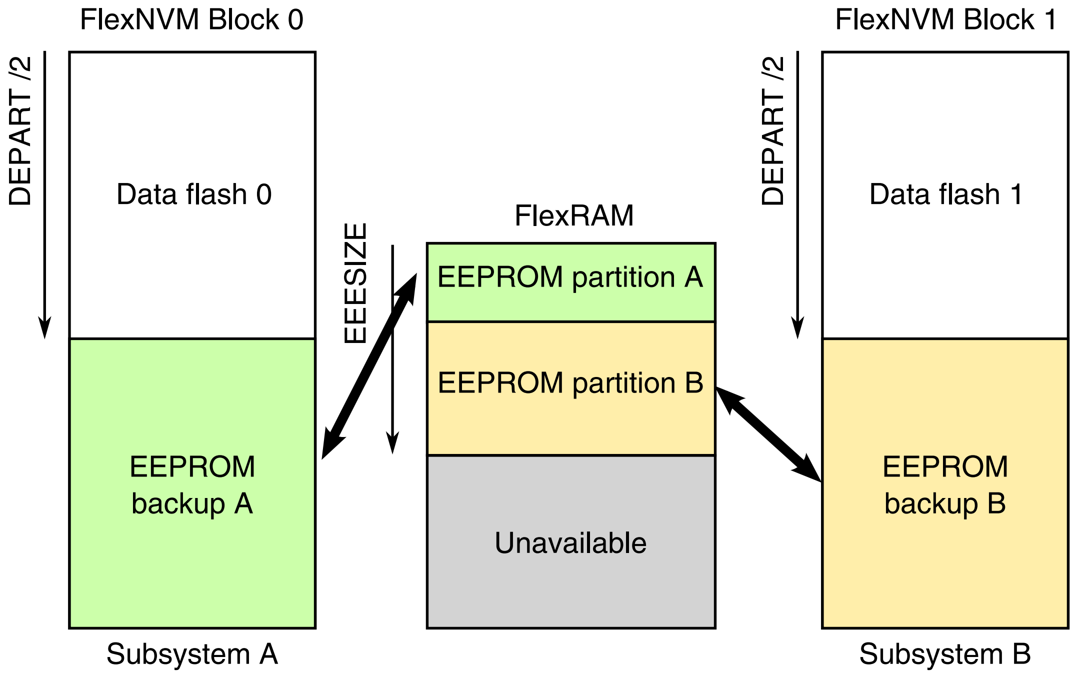
1. **EEPROM partition** (EEESIZE) — The amount of FlexRAM used for EEPROM can be set from 0 Bytes (no EEPROM) to the maximum FlexRAM size (see Table 30-2). The remainder of the FlexRAM not used for EEPROM is not accessible while the FlexRAM is configured for EEPROM (see Set FlexRAM Function command).

The EEPROM partition grows upward from the bottom of the FlexRAM address space.

2. **Data flash partition** (DEPART) — The amount of FlexNVM memory used for data flash can be programmed from 0 bytes (all of the FlexNVM block is available for EEPROM backup) to the maximum size of the FlexNVM block (see Table 30-4).
3. **FlexNVM EEPROM partition** — The amount of FlexNVM memory used for EEPROM backup, which is equal to the FlexNVM block size minus the data flash memory partition size. The EEPROM backup size must be at least 16 times the EEPROM partition size in FlexRAM.
4. **EEPROM split factor** (EEESPLIT) — The FlexRAM partitioned for EEPROM can be divided into two subsystems, each backed by half of the partitioned EEPROM backup. One subsystem (A) is 1/8, 1/4, or 1/2 of the partitioned FlexRAM with the remainder belonging to the other subsystem (B).

The partition information (EEESIZE, DEPART, EEESPLIT) is stored in the data flash IFR and is programmed using the Program Partition command (see Program Partition command). Typically, the Program Partition command is executed only once in the lifetime of the device.

Data flash memory is useful for applications that need to quickly store large amounts of data or store data that is static. The EEPROM partition in FlexRAM is useful for storing smaller amounts of data that will be changed often. The EEPROM partition in FlexRAM can be further sub-divided to provide subsystems, each backed by the same amount of EEPROM backup with subsytem A having higher endurance if the split factor is 1/8 or 1/4.



EEESPLIT = 1/8, 1/4, or 1/2
Size of EEPROM partition A = EEESIZE x EEESPLIT
Size of EEPROM partition B = EEESIZE x (1 - EEESPLIT)
Data flash 0 and 1 interleaved

Figure 30-32. FlexRAM to FlexNVM Memory Mapping for EEPROM

30.4.3.3 EEPROM implementation overview

Out of reset with the FSTAT[CCIF] bit clear, the partition settings (EEESIZE, DEPART, EEESPLIT) are read from the data flash IFR and the EEPROM file system is initialized accordingly. The EEPROM file system locates all valid EEPROM data records in EEPROM backup and copies the newest data to FlexRAM. The FSTAT[CCIF] and FCNFG[EEERDY] bits are set after data from all valid EEPROM data records is copied to the FlexRAM. After the CCIF bit is set, the FlexRAM is available for read or write access.

When configured for EEPROM use, writes to an unprotected location in FlexRAM invokes the EEPROM file system to program a new EEPROM data record in the EEPROM backup memory in a round-robin fashion. As needed, the EEPROM file system identifies the EEPROM backup sector that is being erased for future use and partially erases that EEPROM backup sector. After a write to the FlexRAM, the FlexRAM is not accessible until the FSTAT[CCIF] bit is set. The FCNFG[EEERDY] bit will also be set. If enabled, the interrupt associated with the FSTAT[CCIF] bit can be used to determine when the FlexRAM is available for read or write access.

After a sector in EEPROM backup is full of EEPROM data records, EEPROM data records from the sector holding the oldest data are gradually copied over to a previously-erased EEPROM backup sector. When the sector copy completes, the EEPROM backup sector holding the oldest data is tagged for erase.

30.4.3.4 Write endurance to FlexRAM for EEPROM

TBD

30.4.4 Interrupts

The FTFE module can generate interrupt requests to the MCU upon the occurrence of various FTFE events. These interrupt events and their associated status and control bits are shown in the following table.

Table 30-30. FTFE Interrupt Sources

FTFE Event	Readable Status Bit	Interrupt Enable Bit
FTFE Command Complete	FSTAT[CCIF]	FCNFG[CCIE]
FTFE Read Collision Error	FSTAT[RDCOLERR]	FCNFG[RDCOLLIE]

Note

> Vector addresses and their relative interrupt priority are determined at the MCU level.

30.4.5 Flash Operation in Low-Power Modes

30.4.5.1 Wait Mode

When the MCU enters wait mode, the FTFE module is not affected. The FTFE module can recover the MCU from wait via the command complete interrupt (see Interrupts).

30.4.5.2 Stop Mode

When the MCU requests stop mode, if an FTFE command is active (CCIF = 0) the command execution completes before the MCU is allowed to enter stop mode.

CAUTION

The MCU should never enter stop mode while any FTFE command is running (CCIF = 0).

NOTE

While the MCU is in very-low-power modes (VLPR, VLPW, VLPS), the FTFE module does not accept flash commands.

30.4.6 Functional modes of operation

The FTFE module has two operating modes: NVM Normal and NVM Special. The operating mode affects the command set availability (see Table 30-31). Refer to the Chip Configuration details of this device for how to activate each mode.

30.4.7 Flash memory reads and ignored writes

The FTFE module requires only the flash address to execute a flash memory read. MCU read access is available to all flash memory.

The MCU must not read from the flash memory while commands are running (as evidenced by CCIF=0) on that block. Read data cannot be guaranteed from a flash block while any command is processing within that block. The block arbitration logic detects any simultaneous access and reports this as a read collision error (see the FSTAT[RDCOLERR] bit).

30.4.8 Read while write (RWW)

The following simultaneous accesses are allowed for devices with FlexNVM:

- The user may read from the program flash memory while commands (typically program and erase operations) are active in the data flash and FlexRAM memory space.
- The user may read from one logical program flash memory space while commands are active in another logical program flash memory space.
- The MCU can fetch instructions from program flash during both data flash program and erase operations and while EEPROM-backup is maintained by the EEPROM commands.
- Conversely, the user may read from data flash and FlexRAM while program and erase commands are executing on the program flash.

- The user may also read from one logical data flash memory space while commands other than Program Partition are active in the other logical data flash memory space.
- When configured as traditional RAM, writes to the FlexRAM are allowed during data flash operations.

Simultaneous data flash operations and FlexRAM writes, when FlexRAM is used for EEE, are not possible.

The following simultaneous accesses are allowed for devices with program flash only:

- The user may read from one logical program flash memory space while commands are active in the other logical program flash memory space.

Simultaneous operations are further discussed in Allowed simultaneous flash operations.

30.4.9 Flash Program and Erase

All flash functions except read require the user to setup and launch an FTFE command through a series of peripheral bus writes. The user cannot initiate any further FTFE commands until notified that the current command has completed. The FTFE command structure and operation are detailed in FTFE Command Operations.

30.4.10 FTFE Command Operations

FTFE command operations are typically used to modify flash memory contents. The next sections describe:

- The command write sequence used to set FTFE command parameters and launch execution
- A description of all FTFE commands available

30.4.10.1 Command Write Sequence

FTFE commands are specified using a command write sequence illustrated in Figure 30-33. The FTFE module performs various checks on the command (FCCOB) content and continues with command execution if all requirements are fulfilled.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be zero and the CCIF flag must read 1 to verify that any previous command has completed. If CCIF is zero, the previous command execution is still active, a new command write sequence cannot be started, and all writes to the FCCOB registers are ignored.

30.4.10.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the desired FTFE command. The individual registers that make up the FCCOB data set can be written in any order.

30.4.10.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing the FSTAT[CCIF] bit by writing a '1' to it. The CCIF flag remains zero until the FTFE command completes.

The FSTAT register contains a blocking mechanism, which prevents a new command from launching (can't clear CCIF) if the previous command resulted in an access error (FSTAT[ACCERR]=1) or a protection violation (FSTAT[FPVIOL]=1). In error scenarios, two writes to FSTAT are required to initiate the next command: the first write clears the error flags, the second write clears CCIF.

30.4.10.1.3 Command Execution and Error Reporting

The command processing has several steps:

1. The FTFE reads the command code and performs a series of parameter checks and protection checks, if applicable, which are unique to each command.

   If the parameter check fails, the FSTAT[ACCERR] (access error) flag is set. ACCERR reports invalid instruction codes and out-of bounds addresses. Usually, access errors suggest that the command was not set-up with valid parameters in the FCCOB register group.

   Program and erase commands also check the address to determine if the operation is requested to execute on protected areas. If the protection check fails, the FSTAT[FPVIOL] (protection error) flag is set.

   Command processing never proceeds to execution when the parameter or protection step fails. Instead, command processing is terminated after setting the FSTAT[CCIF] bit.

2. If the parameter and protection checks pass, the command proceeds to execution. Run-time errors, such as failure to erase verify, may occur during the execution phase. Run-time errors are reported in the FSTAT[MGSTAT0] bit. A command may have access errors, protection errors, and run-time errors, but the run-time errors are not seen until all access and protection errors have been corrected.
3. Command execution results, if applicalbe, are reported back to the user via the FCCOB and FSTAT registers.
4. The FTFE sets the FSTAT[CCIF] bit signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

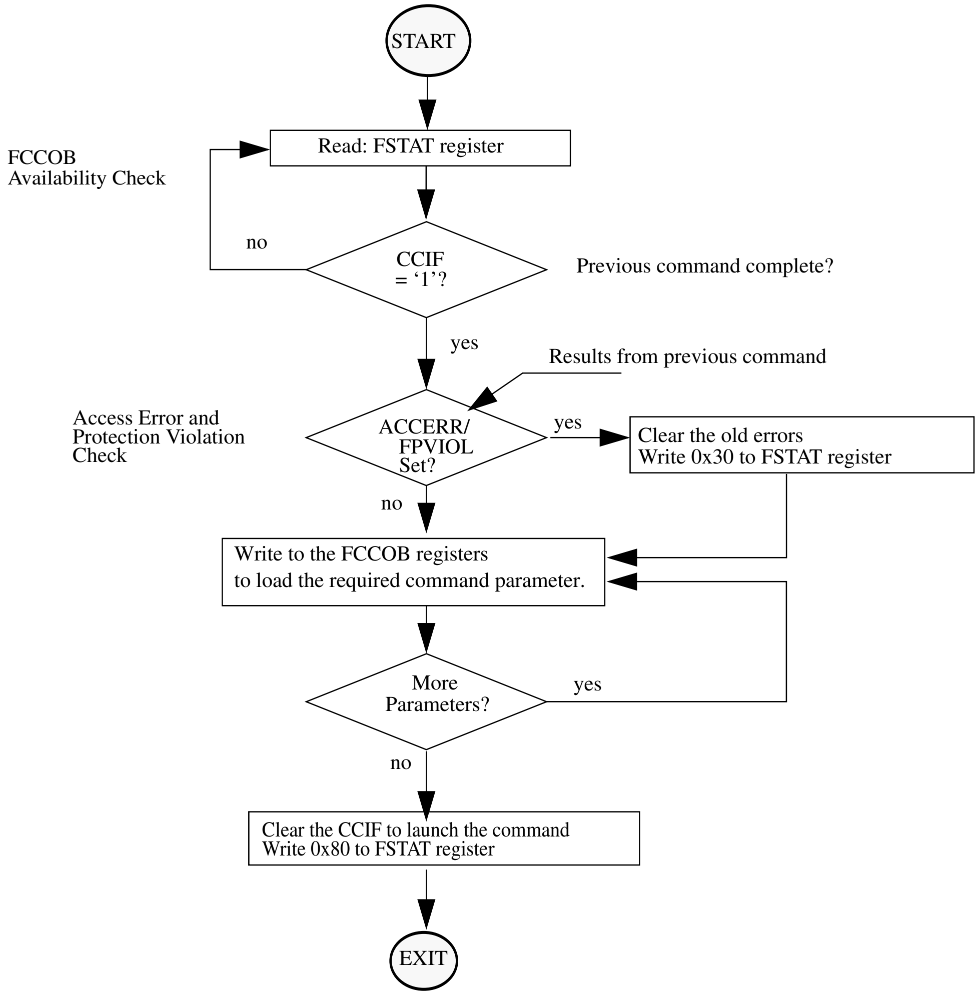

START
Read: FSTAT register
FCCOB
Availability Check
no
CCIF
= ‘1’?
Previous command complete?
yes
Results from previous command
Access Error and
Protection Violation
Check
ACCERR/
FPVIOL
Set?
yes
Clear the old errors
Write 0x30 to FSTAT register
no
Write to the FCCOB registers
to load the required command parameter.
More
Parameters?
yes
no
Clear the CCIF to launch the command
Write 0x80 to FSTAT register
EXIT

Figure 30-33. Generic Flash Command Write Sequence Flowchart

30.4.10.2 Flash commands

The following table summarizes the function of all flash commands. If any column is marked with an 'X', the flash command is relevant to that particular memory resource.

FCMD	Command	Program flash 0	Program flash 1	Program flash 2 (Devices with only program flash)	Program flash 3 (Devices with only program flash)	Data flash 0-1 (Devices with FlexNVM)	FlexRAM (Devices with FlexNVM)	Function
0x00	Read 1s Block	×	×	×	×	×		Verify that a program flash or data flash block is erased. FlexNVM block must not be partitioned for EEPROM.
0x01	Read 1s Section	×	×	×	×	×		Verify that a given number of program flash or data flash locations from a starting address are erased.
0x02	Program Check	×	×	×	×	×		Tests previously-programmed phrases at margin read levels.
0x03	Read Resource	IFR,ID	IFR	IFR	IFR	IFR		Read 8 bytes from program flash IFR, data flash IFR, or version ID.
0x07	Program Phrase	×	×	×	×	×		Program 8 bytes in a program flash block or a data flash block.

Table continues on the next page...

FCMD	Command	Program flash 0	Program flash 1	Program flash 2 (Devices with only program flash)	Program flash 3 (Devices with only program flash)	Data flash 0-1 (Devices with FlexNVM)	FlexRAM (Devices with FlexNVM)	Function
0x08	Erase Flash Block	×	×	×	×	×		Erase a program flash block or data flash block. An erase of any flash block is only possible when unprotected. FlexNVM block must not be partitioned for EEPROM.
0x09	Erase Flash Sector	×	×	×	×	×		Erase all bytes in a program flash or data flash sector.
0x0B	Program Section	×	×	×	×	×	×	Program data from the Section Program Buffer to a program flash or data flash block.
0x40	Read 1s All Blocks	×	×	×	×	×	×	Verify that all program flash, data flash blocks, EEPROM backup data records, and data flash IFR are erased then release MCU security.

Table continues on the next page...

FCMD	Command	Program flash 0	Program flash 1	Program flash 2 (Devices with only program flash)	Program flash 3 (Devices with only program flash)	Data flash 0-1 (Devices with FlexNVM)	FlexRAM (Devices with FlexNVM)	Function
0x41	Read Once	IFR						Read 8 bytes of a dedicated 64 byte field in the program flash 0 IFR.
0x43	Program Once	IFR						One-time program of 8 bytes of a dedicated 64-byte field in the program flash 0 IFR.
0x44	Erase All Blocks	×	×	×	×	×	×	Erase all program flash, data flash blocks, FlexRAM, EEPROM backup data records, and data flash IFR. Then, verify-erase and release MCU security. **NOTE:** An erase is only possible when all memory locations are unprotected.

Table continues on the next page...

FCMD	Command	Program flash 0	Program flash 1	Program flash 2 (Devices with only program flash)	Program flash 3 (Devices with only program flash)	Data flash 0-1 (Devices with FlexNVM)	FlexRAM (Devices with FlexNVM)	Function
0x45	Verify Backdoor Access Key	×		×				Release MCU security after comparing a set of user-supplied security keys to those stored in the program flash.
0x46	Swap Control	x	x	×	×			Handles swap-related activities.
0x80	Program Partition					IFR, ×	×	Program the FlexNVM Partition Code and EEPROM Data Set Size into the data flash IFR. format all EEPROM backup data sectors allocated for EEPROM, initialize the FlexRAM.

Table continues on the next page...

FCMD	Command	Program flash 0	Program flash 1	Program flash 2 (Devices with only program flash)	Program flash 3 (Devices with only program flash)	Data flash 0-1 (Devices with FlexNVM)	FlexRAM (Devices with FlexNVM)	Function
0x81	Set FlexRAM Function					×	×	Switches FlexRAM function between RAM and EEPROM. When switching to EEPROM, FlexNVM is not available while valid data records are being copied from EEPROM backup to FlexRAM.

30.4.10.3 Flash commands by mode

The following table shows the flash commands that can be executed in each flash operating mode.

Table 30-31. Flash commands by mode

FCMD	Command	NVM Normal			NVM Special		
		Unsecure	Secure	MEEN=10	Unsecure	Secure	MEEN=10
0x00	Read 1s Block	×	×	×	×	—	—
0x01	Read 1s Section	×	×	×	×	—	—
0x02	Program Check	×	×	×	×	—	—
0x03	Read Resource	×	×	×	×	—	—
0x07	Program Phrase	×	×	×	×	—	—
0x08	Erase Flash Block	×	×	×	×	—	—
0x09	Erase Flash Sector	×	×	×	×	—	—
0x0B	Program Section	×	×	×	×	—	—
0x40	Read 1s All Blocks	×	×	×	×	×	—
0x41	Read Once	×	×	×	×	—	—
0x43	Program Once	×	×	×	×	—	—

Table continues on the next page...

Table 30-31. Flash commands by mode (continued)

FCMD	Command	NVM Normal			NVM Special		
		Unsecure	Secure	MEEN=10	Unsecure	Secure	MEEN=10
0x44	Erase All Blocks	×	×	×	×	×	—
0x45	Verify Backdoor Access Key	×	×	×	×	—	—
0x46	Swap Control	×	×	×	×	—	—
0x80	Program Partition	×	×	×	×	—	—
0x81	Set FlexRAM Function	×	×	×	×	—	—

30.4.10.4 Allowed simultaneous flash operations

Only the operations marked 'OK' in the following table are permitted to run simultaneously on the program flash, data flash, and FlexRAM memories. Some operations cannot be executed simultaneously because certain hardware resources are shared by the memories. The priority has been placed on permitting program flash reads while program and erase operations execute on the FlexNVM and FlexRAM. This provides read (program flash) while write (FlexNVM, FlexRAM) functionality.

For devices containing FlexNVM:

Table 30-32. Allowed Simultaneous Memory Operations

		Program flash 0/1			Data flash 0/1			FlexRAM		
		Read	Program Phrase	Erase Flash Sector[1]	Read	Program Phrase	Erase Flash Sector[2]	Read	E-Write[3]	R-Write[4]
Program flash 1/0	Read		OK	OK		OK	OK		OK	
	Program Phrase	OK			OK			OK		OK
	Erase Flash Sector[1]	OK			OK			OK		OK
Data flash 1/0	Read		OK	OK		OK	OK			
	Program Phrase	OK			OK			OK		OK
	Erase Flash Sector[1]	OK			OK			OK		OK
FlexRAM	Read		OK	OK		OK	OK			
	E-Write[3]	OK								
	R-Write[4]		OK	OK		OK	OK			

1. Also applies to Erase Flash Block

2. Also applies to Erase Flash Block
3. When FlexRAM configured for EEPROM (EEERDY=1).
4. When FlexRAM configured as traditional RAM (RAMRDY=1); single cycle operation.

For devices containing program flash only:

Table 30-33. Allowed Simultaneous Memory Operations

		Program flash Y				
		Read	**Program Phrase**	**Erase Flash Sector**	**Erase Flash Block**	
Program flash X	**Read**			OK	OK	OK
	Program Phrase	OK				
	Erase Flash Sector	OK				
	Erase Flash Block	OK				

30.4.11 Margin Read Commands

The Read-1s commands (Read 1s All Blocks, Read 1s Block, and Read 1s Section) and the Program Check command have a margin choice parameter that allows the user to apply non-standard read reference levels to the program flash and data flash array reads performed by these commands. Using the preset 'user' and 'factory' margin levels, these commands perform their associated read operations at tighter tolerances than a 'normal' read. These non-standard read levels are applied only during the command execution. All simple (uncommanded) flash array reads to the MCU always use the standard, un-margined, read reference level.

Only the 'normal' read level should be employed during normal flash usage. The non-standard, 'user' and 'factory' margin levels should be employed only in special cases. They can be used during special diagnostic routines to gain confidence that the device is not suffering from the end-of-life data loss customary of flash memory devices.

Erased ('1') and programmed ('0') bit states can degrade due to elapsed time and data cycling (number of times a bit is erased and re-programmed). The lifetime of the erased states is relative to the last erase operation. The lifetime of the programmed states is measured from the last program time.

The 'user' and 'factory' levels become, in effect, a minimum safety margin; i.e. if the reads pass at the tighter tolerances of the 'user' and 'factory' margins, then the 'normal' reads have at least this much safety margin before they experience data loss.

The 'user' margin is a small delta to the normal read reference level. 'User' margin levels can be employed to check that flash memory contents have adequate margin for normal level read operations. If unexpected read results are encountered when checking flash memory contents at the 'user' margin levels, loss of information might soon occur during 'normal' readout.

The 'factory' margin is a bigger deviation from the norm, a more stringent read criteria that should only be attempted immediately (or very soon) after completion of an erase or program command, early in the cycling life. 'Factory' margin levels can be used to check that flash memory contents have adequate margin for long-term data retention at the normal level setting. If unexpected results are encountered when checking flash memory contents at 'factory' margin levels, the flash memory contents should be erased and reprogrammed.

CAUTION

> Factory margin levels must only be used during verify of the initial factory programming.

30.4.12 Flash command descriptions

This section describes all flash commands that can be launched by a command write sequence. The FTFE sets the FSTAT[ACCERR] bit and aborts the command execution if any of the following illegal conditions occur:

- There is an unrecognized command code in the FCCOB FCMD field.
- There is an error in a FCCOB field for the specific commands. Refer to the error handling table provided for each command.

Ensure that the ACCERR and FPVIOL bits in the FSTAT register are cleared prior to starting the command write sequence. As described in Launch the Command by Clearing CCIF, a new command cannot be launched while these error flags are set.

Do not attempt to read a flash block while the FTFE is running a command (CCIF = 0) on that same block. The FTFE may return invalid data to the MCU with the collision error flag (FSTAT[RDCOLERR]) set.

When required by the command, address bit 23 selects between:

- program flash memory (=0)
- data flash memory (=1)

CAUTION

Flash data must be in the erased state before being programmed. Cumulative programming of bits (adding more zeros) is not allowed.

30.4.12.1 Read 1s Block command

The Read 1s Block command checks to see if an entire program flash or data flash logical block has been erased to the specified margin level. The FCCOB flash address bits determine which logical block is erase-verified.

Table 30-34. Read 1s Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x00 (RD1BLK)
1	Flash address [23:16] in the flash block to be verified
2	Flash address [15:8] in the flash block to be verified
3	Flash address [7:0][1] in the flash block to be verified
4	Read-1 Margin Choice

[1] Must be 128-bit aligned (Flash address [3:0] = 0000).

After clearing CCIF to launch the Read 1s Block command, the FTFE sets the read margin for 1s according to Table 30-35 and then reads all locations within the selected program flash or data flash block.

When the data flash is targeted, DEPART must be set for no EEPROM, else the Read 1s Block command aborts setting the FSTAT[ACCERR] bit. If the FTFE fails to read all 1s (i.e. the flash block is not fully erased), the FSTAT[MGSTAT0] bit is set. The CCIF flag sets after the Read 1s Block operation has completed.

Table 30-35. Margin Level Choices for Read 1s Block

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 30-36. Read 1s Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin choice is specified	FSTAT[ACCERR]

Table continues on the next page...

Table 30-36. Read 1s Block Command Error Handling (continued)

Error Condition	Error Bit
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Data flash is selected and the address is out of data flash range	FSTAT[ACCERR]
Data flash is selected with EEPROM enabled	FSTAT[ACCERR]
Flash address is not 128-bit aligned	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

30.4.12.2 Read 1s Section command

The Read 1s Section command checks if a section of program flash or data flash memory is erased to the specified read margin level. The Read 1s Section command defines the starting address and the number of 128 bits to be verified.

Table 30-37. Read 1s Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x01 (RD1SEC)
1	Flash address [23:16] of the first 128 bits to be verified
2	Flash address [15:8] of the first 128 bits to be verified
3	Flash address [7:0][1] of the first 128 bits to be verified
4	Number of 128 bits to be verified [15:8]
5	Number of 128 bits to be verified [7:0]
6	Read-1 Margin Choice

[1] Must be 128-bit aligned (Flash address [3:0] = 0000).

Upon clearing CCIF to launch the Read 1s Section command, the FTFE sets the read margin for 1s according to Table 30-38 and then reads all locations within the specified section of flash memory.

If the FTFE fails to read all 1s (i.e. the flash section is not erased), the FSTAT(MGSTAT0) bit is set. The CCIF flag sets after the Read 1s Section operation completes.

Table 30-38. Margin Level Choices for Read 1s Section

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 30-39. Read 1s Section Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid margin code is supplied	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not 128-bit aligned	FSTAT[ACCERR]
The requested section crosses a logical flash block boundary	FSTAT[ACCERR]
The requested number of 128 bits is zero	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

30.4.12.3 Program Check command

The Program Check command tests a previously programmed program flash or data flash longword to see if it reads correctly at the specified margin level.

Table 30-40. Program Check Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x02 (PGMCHK)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Margin Choice
8	Byte 0 expected data
9	Byte 1 expected data
A	Byte 2 expected data
B	Byte 3 expected data

[1] Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Check command, the FTFE sets the read margin for 1s based on the provided margin choice according to Table 30-41. The Program Check operation then reads the specified longword, and compares the actual read data to the expected data provided by the FCCOB. If the comparison at margin-1 fails, the MGSTAT0 bit is set.

The MGATE will then set the read margin for 0s based on the provided margin choice.The Program Check operation will then read the specified longword and compare the actual read data to the expected data provided by the FCCOB. If the comparison at margin-0 fails, the MGSTAT0 bit will be set. The CCIF flag will set after the Program Check operation has completed.

The starting address must be longword aligned (the lowest two bits of the byte address must be 00):

- Byte 0 data is expected at the supplied 32-bit aligned address,
- Byte 1 data is expected at byte address specified + 0b01,
- Byte 2 data is expected at byte address specified + 0b10, and
- Byte 3 data is expected at byte address specified + 0b11.

NOTE

See the description of margin reads, Margin Read Commands

Table 30-41. Margin Level Choices for Program Check

Read Margin Choice	Margin Level Description
0x01	Read at 'User' margin-1 and 'User' margin-0
0x02	Read at 'Factory' margin-1 and 'Factory' margin-0

Table 30-42. Program Check Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not longword aligned	FSTAT[ACCERR]
An invalid margin choice is supplied	FSTAT[ACCERR]
Either of the margin reads does not match the expected data	FSTAT[MGSTAT0]

30.4.12.4 Read Resource Command

The Read Resource command is provided for the user to read data from special-purpose memory resources located within the Flash module. The special-purpose memory resources available include program flash IFR, data flash (User) IFR space, and the Version ID field. The Version ID field contains an 8 byte code that indicates a specific FTFE implementation.

Table 30-43. Read Resource Command FCCOB Requirements

FCCOB Number	FCCOB contents [7:0]
0	0x03 (RDRSRC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Resource select code (see Table 30-44)

Table continues on the next page...

Table 30-43. Read Resource Command FCCOB Requirements (continued)

FCCOB Number	FCCOB contents [7:0]
Returned values	
4	Read Data [64:56]
5	Read Data [55:48]
6	Read Data [47:40]
7	Read Data [39:32]
8	Read Data [31:24]
9	Read Data [23:16]
A	Read Data [15:8]
B	Read Data [7:0]

1. Must be 64-bit aligned (Flash address [2:0] = 000).

Table 30-44. Read Resource Select Codes

Resource Select Code[1]	Description	Resource Size	Local Address Range
0x00	IFR	1024 Bytes	0x0000 - 0x03FF
0x01[2]	Version ID	8 Bytes	0x0008 - 0x000F

1. Flash address [23] is used to select between program flash 0 (=0) and FlexNVM 0 (=1) IFR resources.
2. Located in program flash 0 reserved space; Flash address [23] = 0

After clearing CCIF to launch the Read Resource command, eight consecutive bytes are read from the selected resource at the provided relative address and stored in the FCCOB register. The CCIF flag will set after the Read Resource operation has completed. The Read Resource command exits with an access error if an invalid resource code is provided or if the address for the applicable area is out-of-range.

Table 30-45. Read Resource Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid resource code is entered	FSTAT[ACCERR]
Flash address is out-of-range for the targeted resource.	FSTAT[ACCERR]
Flash address is not 64-bit aligned	FSTAT[ACCERR]

30.4.12.5 Program Phrase command

The Program Phrase command programs eight previously-erased bytes in the program flash memory or in the data flash memory using an embedded algorithm.

CAUTION

A Flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a Flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

Table 30-46. Program Phrase Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x07 (PGM8)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Byte 0 program value
5	Byte 1 program value
6	Byte 2 program value
7	Byte 3 program value
8	Byte 4 program value
9	Byte 5 program value
A	Byte 6 program value
B	Byte 7 program value

[1] Must be 64-bit aligned (Flash address [2:0] = 000)

Upon clearing CCIF to launch the Program Phrase command, the FTFE programs the data bytes into the flash using the supplied address. The protection status is always checked. The swap indicator address is implicitly protected from programming in NVM Normal and Special modes. The targeted flash locations must be currently unprotected (see the description of the FPROT registers) to permit execution of the Program Phrase operation.

The programming operation is unidirectional. It can only move NVM bits from the erased state ('1') to the programmed state ('0'). Erased bits that fail to program to the '0' state are flagged as errors in MGSTAT0. The CCIF flag is set after the Program Phrase operation completes.

The starting address must be 64-bit aligned (flash address [3:0] = 000):

- Byte 0 data is written to the starting address ('start'),
- Byte 1 data is programmed to byte address start+0b01,
- Byte 2 data is programmed to byte address start+0b10, and
- Byte 3 data is programmed to byte address start+0b11, etc.

Table 30-47. Program Phrase Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not 64-bit aligned	FSTAT[ACCERR]
Flash address points to a protected area	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation.	FSTAT[MGSTAT0]

30.4.12.6 Erase Flash Block Command

The Erase Flash Block operation erases all addresses in a single program flash or data flash block.

Table 30-48. Erase Flash Block Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x08 (ERSBLK)
1	Flash address [23:16] in the flash block to be erased
2	Flash address [15:8] in the flash block to be erased
3	Flash address [7:0][1] in the flash block to be erased

[1] Must be 128-bit aligned (Flash address [3:0] = 0000).

Upon clearing CCIF to launch the Erase Flash Block command, the FTFE erases the main array of the selected flash block and verifies that it is erased. When the data flash is targeted, DEPART must be set for no EEPROM (see Table 30-4) else the Erase Flash Block command aborts setting the FSTAT[ACCERR] bit. The Erase Flash Block command aborts and sets the FSTAT[FPVIOL] bit if any region within the block is protected (see the description of the program flash protection (FPROT) registers and the data flash protection (FDPROT) registers). The swap indicator address is implicitly protected from block erase unless the swap system is in the update mode and the program flash block being erased is the non-active block that contains the swap indicator address. If the erase verify fails, the MGSTAT0 bit in FSTAT is set. The CCIF flag will set after the Erase Flash Block operation has completed.

Table 30-49. Erase Flash Block Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Program flash is selected and the address is out of program flash range	FSTAT[ACCERR]
Data flash is selected and the address is out of data flash range	FSTAT[ACCERR]

Table continues on the next page...

Table 30-49. Erase Flash Block Command Error Handling (continued)

Error Condition	Error Bit
Data flash is selected with EEPROM enabled	FSTAT[ACCERR]
Flash address is not 128-bit aligned	FSTAT[ACCERR]
Any area of the selected flash block is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

30.4.12.7 Erase Flash Sector command

The Erase Flash Sector operation erases all addresses in a flash sector.

Table 30-50. Erase Flash Sector Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x09 (ERSSCR)
1	Flash address [23:16] in the flash sector to be erased
2	Flash address [15:8] in the flash sector to be erased
3	Flash address [7:0][1] in the flash sector to be erased

[1] Must be 128-bit aligned (flash address [3:0] = 0000).

After clearing CCIF to launch the Erase Flash Sector command, the FTFE erases the selected program flash or data flash sector and then verifies that it is erased. The Erase Flash Sector command aborts if the selected sector is protected (see the description of the FPROT registers). The swap indicator address in each program flash block is implicitly protected from sector erase unless the swap system is in the update mode and the program flash sector containing the swap indicator address being erased is in the non-active block. If the erase-verify fails the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase Flash Sector operation completes. The Erase Flash Sector command is suspendable (see the FCNFG[ERSSUSP] bit and Figure 30-34).

Table 30-51. Erase Flash Sector Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid Flash address is supplied	FSTAT[ACCERR]
Flash address is not 128-bit aligned	FSTAT[ACCERR]
The selected program flash or data flash sector is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

30.4.12.7.1 Suspending an Erase Flash Sector Operation

To suspend an Erase Flash Sector operation set the FCNFG[ERSSUSP] bit (see Flash configuration field description) when CCIF is clear and the CCOB command field holds the code for the Erase Flash Sector command. During the Erase Flash Sector operation (see Erase Flash Sector command), the flash samples the state of the ERSSUSP bit at convenient points. If the FTFE detects that the ERSSUSP bit is set, the Erase Flash Sector operation is suspended and the FTFE sets CCIF. While ERSSUSP is set, all writes to flash registers are ignored except for writes to the FSTAT and FCNFG registers.

If an Erase Flash Sector operation effectively completes before the FTFE detects that a suspend request has been made, the FTFE clears the ERSSUSP bit prior to setting CCIF. When an Erase Flash Sector operation has been successfully suspended, the FTFE sets CCIF and leaves the ERSSUSP bit set. While CCIF is set, the ERSSUSP bit can only be cleared to prevent the withdrawal of a suspend request before the FTFE has acknowledged it.

30.4.12.7.2 Resuming a Suspended Erase Flash Sector Operation

If the ERSSUSP bit is still set when CCIF is cleared to launch the next command, the previous Erase Flash Sector operation resumes. The FTFE acknowledges the request to resume a suspended operation by clearing the ERSSUSP bit. A new suspend request can then be made by setting ERSSUSP. A single Erase Flash Sector operation can be suspended and resumed multiple times.

There is a minimum elapsed time limit between the request to resume the Erase Flash Sector operation (CCIF is cleared) and the request to suspend the operation again (ERSSUSP is set). This minimum time period is required to ensure that the Erase Flash Sector operation will eventually complete. If the minimum period is continually violated, i.e. the suspend requests come repeatedly and too quickly, no forward progress is made by the Erase Flash Sector algorithm. The resume/suspend sequence runs indefinitely without completing the erase.

30.4.12.7.3 Aborting a Suspended Erase Flash Sector Operation

The user may choose to abort a suspended Erase Flash Sector operation by clearing the ERSSUSP bit prior to clearing CCIF for the next command launch. When a suspended operation is aborted, the FTFE starts the new command using the new FCCOB contents.

While FCNFG[ERSSUSP] is set, a write to the FlexRAM while FCNFG[EEERDY] is set clears ERSSUSP and aborts the suspended operation. The FlexRAM write operation is executed by the FTFE.

Note

Aborting the erase leaves the bitcells in an indeterminate, partially-erased state. Data in this sector is not reliable until a new erase command fully completes.

The following figure shows how to suspend and resume the Erase Flash Sector operation.

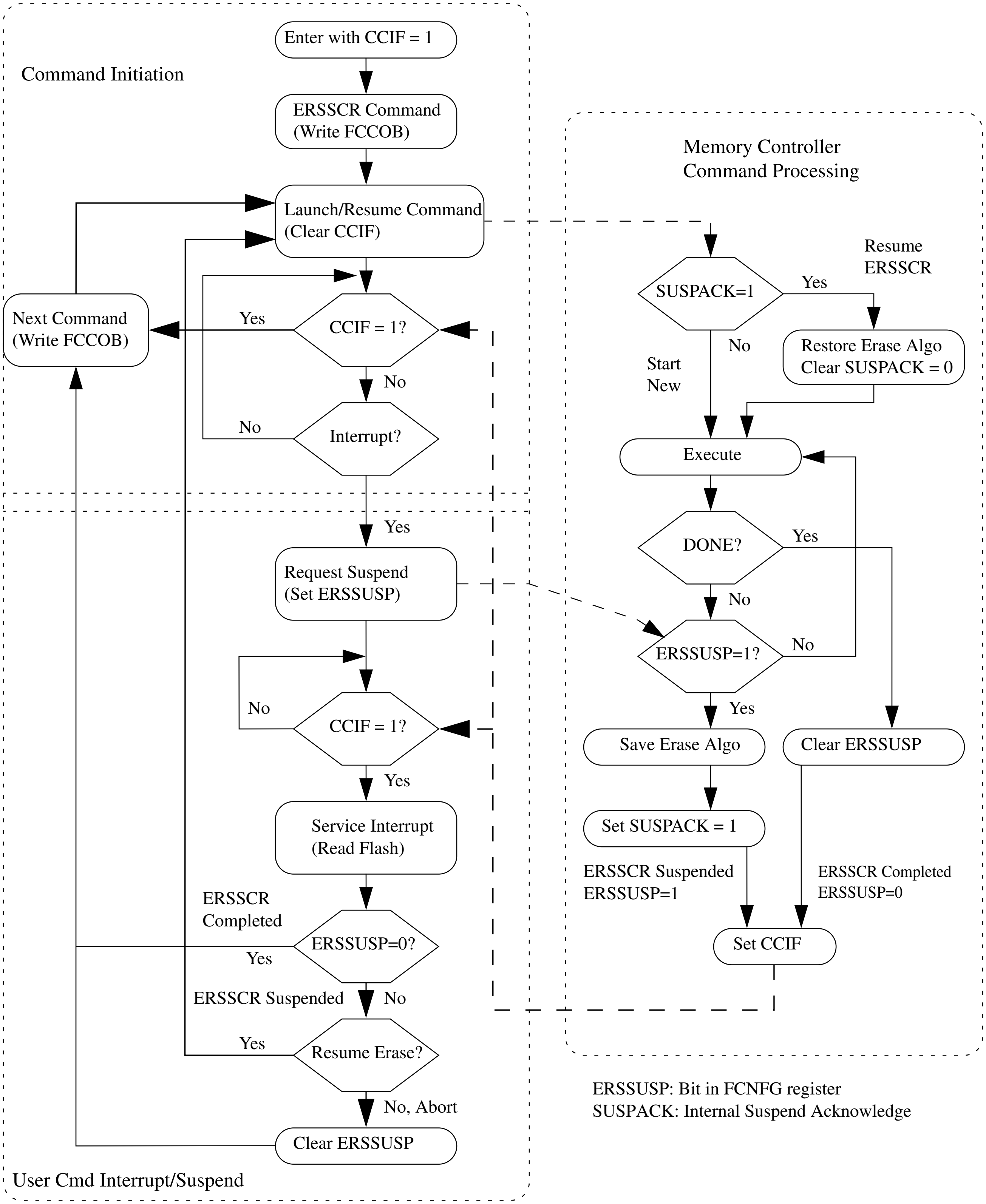

Command Initiation
Enter with CCIF = 1
ERSSCR Command (Write FCCOB)
Launch/Resume Command (Clear CCIF)
Next Command (Write FCCOB)
Yes
CCIF = 1?
No
No
Interrupt?
Yes
Request Suspend (Set ERSSUSP)
No
CCIF = 1?
Yes
Service Interrupt (Read Flash)
ERSSCR Completed
ERSSUSP=0?
Yes
ERSSCR Suspended
No
Yes
Resume Erase?
No, Abort
Clear ERSSUSP
User Cmd Interrupt/Suspend
Memory Controller Command Processing
Resume ERSSCR
SUSPACK=1
Yes
Restore Erase Algo Clear SUSPACK = 0
No
Start New
Execute
DONE?
Yes
No
ERSSUSP=1?
No
Yes
Save Erase Algo
Clear ERSSUSP
Set SUSPACK = 1
ERSSCR Suspended ERSSUSP=1
ERSSCR Completed ERSSUSP=0
Set CCIF
ERSSUSP: Bit in FCNFG register
SUSPACK: Internal Suspend Acknowledge

Figure 30-34. Suspend and Resume of Erase Flash Sector Operation

30.4.12.8 Program Section command

The Program Section operation programs the data found in the section program buffer to previously erased locations in the flash memory using an embedded algorithm. Data is preloaded into the section program buffer by writing to the FlexRAM while it is set to function as traditional RAM (see Flash sector programming).

The section program buffer is limited to the lower quarter of the FlexRAM (byte addresses 0x0000-0x0FFF). Data written to the remainder of the FlexRAM is ignored and may be overwritten during Program Section command execution.

CAUTION

> A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

Table 30-52. Program Section Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x0B (PGMSEC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Number of 128 bits to program [15:8]
5	Number of 128 bits to program [7:0]

[1] Must be 128-bit aligned (Flash address [3:0] = 0000).

After clearing CCIF to launch the Program Section command, the FTFE will block access to the programming acceleration RAM (program flash only devices) or FlexRAM (FlexNVM devices) and program the data residing in the Section Program Buffer into the flash memory starting at the flash address provided.

The starting address must be unprotected (see the description of the FPROT registers) to permit execution of the Program Section operation. The swap indicator address in both program flash blocsk is implicitly protected from programming. If the swap indicator address is encountered during the Program Section operation, it will be bypassed without setting FPVIOL and the contents will not be programmed. Programming, which is not allowed to cross a flash sector boundary, continues until all requested double-phrases have been programmed.

After the Program Section operation has completed, the CCIF flag will set and normal access to the FlexRAM is restored. The contents of the Section Program Buffer is not changed by the Program Section operation.

Table 30-53. Program Section Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid flash address is supplied	FSTAT[ACCERR]
Flash address is not 128-bit aligned	FSTAT[ACCERR]
The requested section crosses a program flash sector boundary	FSTAT[ACCERR]
The requested number of double phrases is zero	FSTAT[ACCERR]
The space required to store data for the requested number of double phrases is more than one quarter the size of the programming acceleration RAM (program flash only devices) or FlexRAM (FlexNVM devices)	FSTAT[ACCERR]
The FlexRAM is not set to function as a traditional RAM, i.e. set if RAMRDY=0	FSTAT[ACCERR]
The flash address falls in a protected area	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

30.4.12.8.1 Flash sector programming

The process of programming an entire flash sector using the Program Section command is as follows:

1. If required, execute the Set FlexRAM Function command to make the FlexRAM available as traditional RAM and initialize the FlexRAM to all ones.
2. Launch the Erase Flash Sector command to erase the flash sector to be programmed.
3. Beginning with the starting address of the programming acceleration RAM (program flash only devices) or FlexRAM (FlexNVM devices), sequentially write enough data to the RAM to fill an entire flash sector. This area of the RAM serves as the section program buffer.

   NOTE

   In step 1, the section program buffer was initialized to all ones, the erased state of the flash memory.

   The section program buffer can be written to while the operation launched in step 2 is executing, i.e. while CCIF = 0.
4. Execute the Program Section command to program the contents of the section program buffer into the selected flash sector.
5. To program additional flash sectors, repeat steps 2 through 4.
6. To restore EEPROM functionality, execute the Set FlexRAM Function command to make the FlexRAM available for EEPROM.

30.4.12.9 Read 1s All Blocks Command

The Read 1s All Blocks command checks if the program flash blocks, data flash blocks, EEPROM backup records, and data flash IFR have been erased to the specified read margin level, if applicable, and releases security if the readout passes, i.e. all data reads as '1'.

Table 30-54. Read 1s All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x40 (RD1ALL)
1	Read-1 Margin Choice

After clearing CCIF to launch the Read 1s All Blocks command, the FTFE :

- sets the read margin for 1s according to Table 30-55,
- checks the contents of the program flash, data flash, EEPROM backup records, and data flash IFR are in the erased state.

If the FTFE confirms that these memory resources are erased, security is released by setting the FSEC[SEC] field to the unsecure state. The security byte in the flash configuration field (see Flash configuration field description) remains unaffected by the Read 1s All Blocks command. If the read fails, i.e. all flash memory resources are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The EEERDY and RAMRDY bits are clear during the Read 1s All Blocks operation and are restored at the end of the Read 1s All Blocks operation.

The CCIF flag sets after the Read 1s All Blocks operation has completed.

Table 30-55. Margin Level Choices for Read 1s All Blocks

Read Margin Choice	Margin Level Description
0x00	Use the 'normal' read level for 1s
0x01	Apply the 'User' margin to the normal read-1 level
0x02	Apply the 'Factory' margin to the normal read-1 level

Table 30-56. Read 1s All Blocks Command Error Handling

Error Condition	Error Bit
An invalid margin choice is specified	FSTAT[ACCERR]
Read-1s fails	FSTAT[MGSTAT0]

30.4.12.10 Read Once Command

The Read Once command provides read access to a reserved 64-byte field located in the program flash 0 IFR (see Program flash 0 IFR map and Program Once field). Access to this field is via 8 records, each 8 bytes long. The Read Once field is programmed using the Program Once command described in Program Once command.

Table 30-57. Read Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x41 (RDONCE)
1	Read Once record index (0x00 - 0x07)
Returned Values	
4	Read Once byte 0 value
5	Read Once byte 1 value
6	Read Once byte 2 value
7	Read Once byte 3 value
8	Read Once byte 4 value
9	Read Once byte 5 value
A	Read Once byte 6 value
B	Read Once byte 7 value

After clearing CCIF to launch the Read Once command, an 8-byte Read Once record is read from the program flash IFR and stored in the FCCOB register. The CCIF flag is set after the Read Once operation completes. Valid record index values for the Read Once command range from 0x00 to 0x07. During execution of the Read Once command, any attempt to read addresses within the program flash block containing this 64-byte field returns invalid data. The Read Once command can be executed any number of times.

Table 30-58. Read Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]

30.4.12.11 Program Once command

The Program Once command enables programming to a reserved 64-byte field in the program flash 0 IFR (see Program flash 0 IFR map and Program Once field). Access to the Program Once field is via 8 records, each 8 bytes long. The Program Once field can be read using the Read Once command (see Read Once Command) or using the Read Resource command (see Read Resource Command). Each Program Once record can be programmed only once since the program flash 0 IFR cannot be erased.

Table 30-59. Program Once Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x43 (PGMONCE)
1	Program Once record index (0x00 - 0x0F)
2	Not Used
3	Not Used
4	Program Once Byte 0 value
5	Program Once Byte 1 value
6	Program Once Byte 2 value
7	Program Once Byte 3 value
8	Program Once Byte 4 value
9	Program Once Byte 5 value
A	Program Once Byte 6 value
B	Program Once Byte 7 value

After clearing CCIF to launch the Program Once command, the FTFE first verifies that the selected record is erased. If erased, then the selected record is programmed using the values provided. The Program Once command also verifies that the programmed values read back correctly. The CCIF flag is set after the Program Once operation has completed.

The reserved program flash 0 IFR location accessed by the Program Once command cannot be erased and any attempt to program one of these records when the existing value is not Fs (erased) is not allowed. Valid record index values for the Program Once command range from 0x00 to 0x07. During execution of the Program Once command, any attempt to read addresses within program flash 0 returns invalid data.

Table 30-60. Program Once Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
An invalid record index is supplied	FSTAT[ACCERR]
The requested record has already been programmed to a non-erased value[1]	FSTAT[ACCERR]
Any errors have been encountered during the verify operation.	FSTAT[MGSTAT0]

[1] If a Program Once record is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command is allowed to execute again on that same record.

30.4.12.12 Erase All Blocks Command

The Erase All Blocks operation erases all flash memory, initializes the FlexRAM, verifies all memory contents, and releases MCU security.

Table 30-61. Erase All Blocks Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x44 (ERSALL)

After clearing CCIF to launch the Erase All Blocks command, the FTFE erases all program flash memory, the program flash IFR containing the swap indicator address, data flash memory, data flash IFR space, EEPROM backup memory, and FlexRAM, then verifies that all are erased.

If the FTFE verifies that all flash memories and the FlexRAM were properly erased, security is released by setting the FSEC[SEC] field to the unsecure state and the FCNFG[RAMRDY] bit is set. The Erase All Blocks command aborts if any flash or FlexRAM region is protected. The swap indicator address in the program flash blocks are not implicitly protected from the erase operation. The security byte and all other contents of the flash configuration field (see Flash configuration field description) are erased by the Erase All Blocks command. If the erase-verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks operation completes.

Table 30-62. Erase All Blocks Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
Any region of the program flash memory, data flash memory, or FlexRAM is protected	FSTAT[FPVIOL]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

30.4.12.12.1 Triggering an erase all external to the flash module

The functionality of the Erase All Blocks command is also available in an uncommanded fashion outside of the flash memory. Refer to the device's Chip Configuration details for information on this functionality.

Before invoking the external erase all function, the FSTAT[ACCERR and FPVIOL] flags must be cleared and the FCCOB0 register must not contain 0x44. When invoked, the erase-all function erases all program flash memory, the program flash IFR containing the swap indicator address, data flash memory, data flash IFR space, EEPROM backup, and FlexRAM regardless of the protection settings. If the post-erase verify passes, the routine then releases security by setting the FSEC[SEC] field register to the unsecure state and the FCNFG[RAMRDY] bit sets. The security byte in the Flash Configuration Field is also programmed to the unsecure state. The status of the erase-all request is reflected in the FCNFG[ERSAREQ] bit. The FCNFG[ERSAREQ] bit is cleared once the operation completes and the normal FSTAT error reporting is available as described in Erase All Blocks Command.

30.4.12.13 Verify Backdoor Access Key command

The Verify Backdoor Access Key command only executes if the mode and security conditions are satisfied (see Flash commands by mode). Execution of the Verify Backdoor Access Key command is further qualified by the FSEC[KEYEN] bits. The Verify Backdoor Access Key command releases security if user-supplied keys in the FCCOB match those stored in the Backdoor Comparison Key bytes of the Flash Configuration Field. The column labeled Flash Configuration Field offset address shows the location of the matching byte in the Flash Configuration Field.

Table 30-63. Verify Backdoor Access Key Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]	Flash Configuration Field Offset Address
0	0x45 (VFYKEY)	
1-3	Not Used	
4	Key Byte 0	0x0_0000
5	Key Byte 1	0x0_0001
6	Key Byte 2	0x0_0002
7	Key Byte 3	0x0_0003
8	Key Byte 4	0x0_0004
9	Key Byte 5	0x0_0005
A	Key Byte 6	0x0_0006
B	Key Byte 7	0x0_0007

After clearing CCIF to launch the Verify Backdoor Access Key command, the FTFE checks the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the FTFE sets the FSTAT[ACCERR] bit and terminates. If the command is enabled, the FTFE compares the key provided in FCCOB to the backdoor comparison key in the Flash Configuration Field. If the backdoor keys match, the FSEC[SEC] field is changed to the unsecure state and security is released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are immediately aborted and the FSTAT[ACCERR] bit is (again) set to 1 until a reset of the FTFE module occurs. If the entire 8-byte key is all zeros or all ones, the Verify Backdoor Access Key command fails with an access error. The CCIF flag is set after the Verify Backdoor Access Key operation completes.

Table 30-64. Verify Backdoor Access Key Command Error Handling

Error Condition	Error Bit
The supplied key is all-0s or all-Fs	FSTAT[ACCERR]
An incorrect backdoor key is supplied	FSTAT[ACCERR]
Backdoor key access has not been enabled (see the description of the FSEC register)	FSTAT[ACCERR]
This command is launched and the backdoor key has mismatched since the last power down reset	FSTAT[ACCERR]

30.4.12.14 Swap Control command

The Swap Control command handles specific activities associated with swapping the two halves of program flash memory within the memory map.

Table 30-65. Swap Control Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x46 (SWAP)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0][1]
4	Swap Control Code: 0x01 - Initialize Swap System 0x02 - Set Swap in Update State 0x04 - Set Swap in Complete State 0x08 - Report Swap Status
Returned values	
5	Current Swap Mode: 0x00 - Uninitialized 0x01 - Ready 0x02 - Update 0x03 - Update-Erased 0x04 - Complete
6	Current Swap Block Status: For devices with FlexNVM: 0x00 - Program flash block 0 at 0x0_0000 0x01 - Program flash block 1 at 0x0_0000 For devices with program flash only: 0x00 - Program flash block 0/1 at 0x0_0000 0x01 - Program flash block 2/3 at 0x0_0000

Table continues on the next page...

Table 30-65. Swap Control Command FCCOB Requirements (continued)

FCCOB Number	FCCOB Contents [7:0]
7	Next Swap Block Status (after any reset): For devices with FlexNVM: 0x00 - Program flash block 0 at 0x0_0000 0x01 - Program flash block 1 at 0x0_0000 For devices with program flash only: 0x00 - Program flash block 0/1 at 0x0_0000 0x01 - Program flash block 2/3 at 0x0_0000

1. Must be 128-bit aligned (Flash address [3:0] = 0000).

Upon clearing CCIF to launch the Swap Control command, the MGATE will handle swap-related activities based on the Swap Control code provided in FCCOB4 as follows:

- 0x01 (Initialize Swap System to UPDATE-ERASED State) - After verifying that the current swap state is UNINITIALIZED and that the flash address provided is in Program flash block 0 but not in the Flash Configuration Field, the flash address (shifted with bit 0 removed) will be programmed into the IFR Swap Field found in a program flash IFR. After the swap indicator address has been programmed into the IFR Swap Field, the swap enable word will be programmed to 0x0000. After the swap enable word has been programmed, the swap indicator, located within the Program flash block 0 address provided, will be programmed to 0xFF00.
- 0x02 (Progress Swap to UPDATE State) - After verifying that the current swap state is READY and that the aligned flash address provided matches the one stored in the IFR Swap Field, the swap indicator located within bits [15:0] of the flash address in the currently active program flash block will be programmed to 0xFF00.
- 0x04 (Progress Swap to COMPLETE State) - After verifying that the current swap state is UPDATE-ERASED and that the aligned flash address provided matches the one stored in the IFR Swap Field, the swap indicator located within bits [15:0] of the flash address in the currently active program flash block will be programmed to 0x0000. Before executing with this Swap Control code, the user must erase the non-active swap indicator using the Erase Flash Block or Erase Flash Sector commands and update the application code or data as needed. The non-active swap indicator will be checked at the erase verify level and if the check fails, the current swap state will be changed to UPDATE with ACCERR set.
- 0x08 (Report Swap Status) - After verifying that the aligned flash address provided is in program flash block 0 but not in the Flash Configuration Field, the status of the swap system will be reported as follows:
  - FCCOB5 (Current Swap State) - indicates the current swap state based on the status of the swap enable phrase and the swap indicators. If the MGSTAT0 flag is set after command completion, the swap state returned was not successfully transitioned from and the appropriate swap command code must be attempted

again. If the current swap state is UPDATE and the non-active swap indicator is 0xFFFF, the current swap state is changed to UPDATE-ERASED.

- FCCOB6 (Current Swap Block Status) - indicates which program flash block is currently located at relative flash address 0x0_0000.
- FCCOB7 (Next Swap Block Status) - indicates which program flash block will be located at relative flash address 0x0_0000 after the next reset of the FTFE module.

NOTE

It is recommended that the user execute the Swap Control command to report swap status (code 0x08) after any reset to determine if issues with the swap system were detected during the swap state determination procedure.

NOTE

It is recommended that the user write 0xFF to FCCOB5, FCCOB6, and FCCOB7 since the Swap Control command will not always return the swap state and status fields when an ACCERR is detected.

The CCIF flag is set after the Swap Control operation has completed.

The swap indicators are implicitly protected from being programmed during Program Phrase or Program Section command operations and are implicitly unprotected during Swap Control command operations. The swap indicators are implicitly protected from being erased during Erase Flash Block and Erase Flash Sector command operations unless the swap indicator being erased is in the non-active program flash block and the swap system is in the UPDATE or UPDATE-ERASED state. Once the swap system has been initialized, the Erase All Blocks command can be used to uninitialize the swap system.

Table 30-66. Swap Control Command Error Handling

Error Condition	Swap Control Code	Error Bit
Command not available in current mode/security[1]	All	FSTAT[ACCERR]
Flash address is not in program flash block 0	All	FSTAT[ACCERR]
Flash address is in the Flash Configuration Field	All	FSTAT[ACCERR]
Flash address is not 128-bit aligned	All	FSTAT[ACCERR]
Flash address does not match the swap indicator address in the IFR	2, 4	FSTAT[ACCERR]
Swap initialize requested when swap system is not in the uninitialized state	1	FSTAT[ACCERR]
Swap update requested when swap system is not in the ready state	2	FSTAT[ACCERR]

Table continues on the next page...

Table 30-66. Swap Control Command Error Handling (continued)

Error Condition	Swap Control Code	Error Bit
Swap complete requested when swap system is not in the update-erased state	4	FSTAT[ACCERR]
An undefined swap control code is provided	-	FSTAT[ACCERR]
Any errors have been encountered during the swap determination and program-verify operations	1, 2, 4	FSTAT[MGSTAT0]
Any brownouts were detected during the swap determination procedure	8	FSTAT[MGSTAT0]

1. Returned fields will not be updated, i.e. no swap state or status reporting

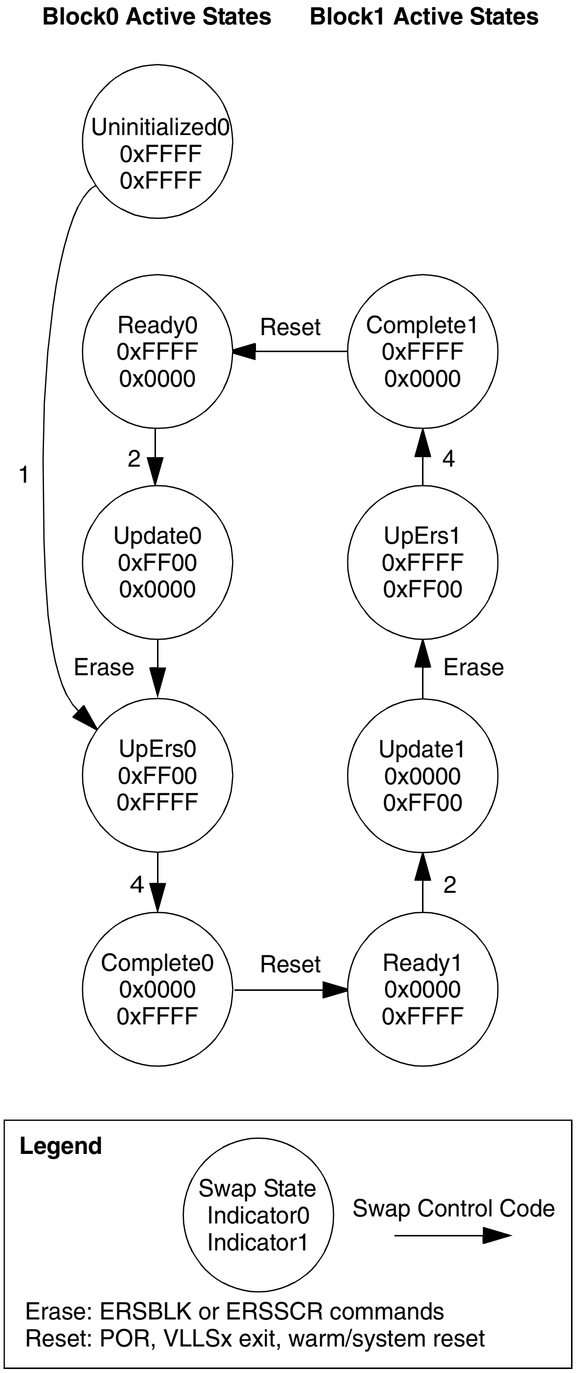


Figure 30-35. Valid Swap State Sequencing

Table 30-67. Swap State Report Mapping

Case	Swap Enable Field[1]	Swap Indicator 0[1]	Swap Indicator 1[1]	Swap State[2]	State Code	MGST AT0	Active Block
1	0xFFFF	-	-	Uninitialized	0	0	0
2	0x0000	0xFF00	0x0000	Update	2	0	0
3	0x0000	0xFF00-	0xFFFF	Update-Erased	3	0	0
4	0x0000	0x0000	0xFFFF[3]	Complete[4]	4	0	0
5	0x0000	0x0000	0xFFFF	Ready[5]	1	0	1
6	0x0000	0x0000	0xFF00	Update	2	0	1
7	0x0000	0xFFFF	0xFF00	Update-Erased	3	0	1
8	0x0000	0xFFFF[3]	0x0000	Complete[4]	4	0	1
9	0x0000	0xFFFF	0x0000	Ready[5]	1	0	0
10	0xXXXX	-	-	Uninitialized	0	1	0
11	0x0000	0xFFFF	0xFFFF	Uninitialized	0	1	0
12	0x0000	0xFFXX	0xFFFF	Ready	1	1	0
13	0x0000	0xFFXX	0x0000	Ready	1	1	0
14[6]	0x0000	0xXXXX	0x0000	Ready	1	1	0
15[6]	0x0000	0xFFFF	0xFFXX	Ready	1	1	1
16	0x0000	0x0000	0xFFXX	Ready	1	1	1
17[6]	0x0000	0x0000	0xXXXX	Ready	1	1	1
18	0x0000	0xFF00	0xFFFF[7]	Update	2	1	0
19	0x0000	0xFF00	0xXXXX	Update	2	1	0
20	0x0000	0xFF(00)	0xFFXX	Update	2	1	0
21[6]	0x0000	0x0000	0x0000	Update	2	1	0
22[6]	0x0000	0xXXXX	0xXXXX	Update	2	1	0
23	0x0000	0xFFFF[7]	0xFF00	Update	2	1	1
24	0x0000	0xXXXX	0xFF00	Update	2	1	1
25	0x0000	0xFFXX	0xFF(00)	Update	2	1	1
26	0x0000	0xXX00	0xFFFF	Update-Erased	3	1	0
27	0x0000	0xXXXX	0xFFFF	Update-Erased	3	1	0
28	0x0000	0xFFFF	0xXX00	Update-Erased	3	1	1
29	0x0000	0xFFFF	0xXXXX	Update-Erased	3	1	1

[1] 0xXXXX, 0xFFXX, 0xXX00 indicates a non-valid value was read; 0xFF(00) indicates more 0's than other indicator (if same number of 0's, then swap system defaults to block 0 active)
[2] Cases 10-29 due to brownout (abort) detected during program or erase steps related to swap
[3] Must read 0xFFFF with erase verify level before transition to Complete allowed
[4] No reset since successful Swap Complete execution
[5] Reset after successful Swap Complete execution
[6] Not a valid case
[7] Fails to read 0xFFFF at erase verify level

30.4.12.14.1 Swap state determination

During the reset sequence, the state of the swap system is determined by evaluating the IFR Swap Field in the program flash 1 IFR and both swap indicators located in the program flash blocks at the swap indicator address stored in the IFR Swap Field.

Table 30-68. Program Flash 1 IFR Swap Field

Address Range	Size (Bytes)	Field Description
0x000 – 0x005	6	Reserved
0x006 – 0x007	2	Swap Enable Word
0x008 – 0x00C	6	Reserved
0x00D – 0x00F	2	Swap Indicator Address
0x010 – 0x3FF	1008	Reserved

30.4.12.15 Program Partition command

The Program Partition command prepares the FlexNVM block for use as data flash, EEPROM backup, or a combination of both and initializes the FlexRAM. The Program Partition command must not be launched from flash memory, since flash memory resources are not accessible during Program Partition command execution.

CAUTION

> While different partitions of the FlexNVM are available, the intention is that a single partition choice is used throughout the entire lifetime of a given application. The FlexNVM Partition Code choices affect the endurance and data retention characteristics of the device.

Table 30-69. Program Partition Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x80 (PGMPART)
1	Not Used
2	Not Used
3	Not Used
4	EEPROM Data Size Code[1]
5	FlexNVM Partition Code[2]

1. See Table 30-70 and EEPROM Data Set Size
2. See Table 30-71 and

Table 30-70. Valid EEPROM Data Set Size Codes

EEPROM Data Set Size Code (FCCOB4)[1]		EEPROM Data Set Size (Bytes) Subsystem A + B
EEESPLIT (FCCOB4[5:4)]	EEESIZE (FCCOB4[3:0])	
11	0xF	0[2]
00	0x9	4 + 28
01		8 + 24
10		16 + 16
11		16 + 16
00	0x8	8 + 56
01		16 + 48
10		32 + 32
11		32 + 32
00	0x7	16 + 112
01		32 + 96
10		64 + 64
11		64 + 64
00	0x6	32 + 224
01		64 + 192
10		128 + 128
11		128 + 128
00	0x5	64 + 448
01		128 + 384
10		256 + 256
11		256 + 256
00	0x4	128 + 896
01		256 + 768
10		512 + 512
11		512 + 512
00	0x3	256 + 1,792
01		512 + 1,536
10		1,024 + 1,024
11		1,024 + 1,024
00	0x2	512 + 3,584
01		1,024 + 3,072
10		2,048 + 2,048
11		2,048 + 2,048

Table continues on the next page...

Table 30-70. Valid EEPROM Data Set Size Codes (continued)

EEPROM Data Set Size Code (FCCOB4)[1]		EEPROM Data Set Size (Bytes) Subsystem A + B
EEESPLIT (FCCOB4[5:4)]	EEESIZE (FCCOB4[3:0])	
00	0x1	1,024 + 7,168
01		2,048 + 6,142
10		4,096 + 4,096
11		4,096 + 4,096
00	0x0	2,048 + 14,336
01		4,096 + 12,284
10		8,192 + 8,192
11		8,192 + 8,192

1. FCCOB4[7:6] = 00
2. EEE Data Set Size must be set to 0 Bytes when the FlexNVM Partition Code is set for no EEPROM.

Table 30-71. Valid FlexNVM Partition Codes

FlexNVM Partition Code DEPART (FCCOB5[3:0)[1]	Data flash Size (Kbytes)	EEPROM-backup Size (Kbytes)
0000	512	0
0100	448	64
0101	384	128
0110	256	256
0111	0	512
1000	0	512
1100	64	448
1101	128	384
1110	256	256
1111	512	0

1. FCCOB5[7:4] = 0000

After clearing CCIF to launch the Program Partition command, the FTFE first verifies that the EEPROM Data Size Code and FlexNVM Partition Code in the data flash IFR are erased. If erased, the Program Partition command erases the contents of the FlexNVM memory. If the FlexNVM is to be partitioned for EEPROM backup, the allocated EEPROM backup sectors are formatted for EEPROM use. Finally, the partition codes are programmed into the data flash IFR using the values provided. The Program Partition command also verifies that the partition codes read back correctly after programming. If the FlexNVM is partitioned for EEPROM, the allocated EEPROM backup sectors are formatted for EEPROM use. The CCIF flag is set after the Program Partition operation completes.

Prior to launching the Program Partition command, the data flash IFR must be in an erased state, which can be accomplished by executing the Erase All Blocks command or by an external request (see Erase All Blocks Command). The EEPROM Data Size Code and FlexNVM Partition Code are read using the Read Resource command (see Read Resource Command).

Table 30-72. Program Partition Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
The EEPROM data size and FlexNVM partition code bytes are not initially 0xFFFF	FSTAT[ACCERR]
Invalid EEPROM Data Size Code is entered (see Table 30-70 for valid codes)	FSTAT[ACCERR]
Invalid FlexNVM Partition Code is entered (see Table 30-71 for valid codes)	FSTAT[ACCERR]
FlexNVM Partition Code = full data flash (no EEPROM) and EEPROM Data Size Code allocates FlexRAM for EEPROM	FSTAT[ACCERR]
FlexNVM Partition Code allocates space for EEPROM backup, but EEPROM Data Size Code allocates no FlexRAM for EEPROM	FSTAT[ACCERR]
FCCOB4[7:6] != 00	FSTAT[ACCERR]
FCCOB5[7:4] != 0000	FSTAT[ACCERR]
Any errors have been encountered during the verify operation	FSTAT[MGSTAT0]

30.4.12.16 Set FlexRAM Function command

The Set FlexRAM Function command changes the function of the FlexRAM:

- When not partitioned for EEPROM, the FlexRAM is typically used as traditional RAM.
- When partitioned for EEPROM, the FlexRAM is typically used to store EEPROM data.

Table 30-73. Set FlexRAM Function Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x81 (SETRAM)
1	FlexRAM Function Control Code (see Table 30-74)

Table 30-74. FlexRAM Function Control

FlexRAM Function Control Code	Action
0xFF	Make FlexRAM available as RAM: • Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags • Write a background of ones to all FlexRAM locations • Set the FCNFG[RAMRDY] flag
0x00	Make FlexRAM available for EEPROM: • Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags • Write a background of ones to all FlexRAM locations • Copy-down existing EEPROM data to FlexRAM • Set the FCNFG[EEERDY] flag

After clearing CCIF to launch the Set FlexRAM Function command, the FTFE sets the function of the FlexRAM based on the FlexRAM Function Control Code.

When making the FlexRAM available as traditional RAM, the FTFE clears the FCNFG[EEERDY] and FCNFG[RAMRDY] flags, overwrites the contents of the entire FlexRAM with a background pattern of all ones, and sets the FCNFG[RAMRDY] flag. The state of the EPROT register does not prevent the FlexRAM from being overwritten. When the FlexRAM is set to function as a RAM, normal read and write accesses to the FlexRAM are available. When large sections of flash memory need to be programmed, e.g. during factory programming, the FlexRAM can be used as the Section Program Buffer for the Program Section command (see Program Section command).

When making the FlexRAM available for EEPROM, the FTFE clears the FCNFG[RAMRDY] and FCNFG[EEERDY] flags, overwrites the contents of the FlexRAM allocated for EEPROM with a background pattern of all ones, and copies the existing EEPROM data from the EEPROM backup record space to the FlexRAM. After completion of the EEPROM copy-down, the FCNFG[EEERDY] flag is set. When the FlexRAM is set to function as EEPROM, normal read and write access to the FlexRAM is available, but writes to the FlexRAM also invoke EEPROM activity.

The CCIF flag will be set after the Set FlexRAM Function operation has completed.

Table 30-75. Set FlexRAM Function Command Error Handling

Error Condition	Error Bit
Command not available in current mode/security	FSTAT[ACCERR]
FlexRAM Function Control Code is not defined	FSTAT[ACCERR]
FlexRAM Function Control Code is set to make the FlexRAM available for EEPROM, but FlexNVM is not partitioned for EEPROM	FSTAT[ACCERR]

30.4.13 Security

The FTFE module provides security information to the MCU based on contents of the FSEC security register. The MCU then limits access to FTFE resources as defined in the device's Chip Configuration details. During reset, the FTFE module initializes the FSEC register using data read from the security byte of the Flash Configuration Field (see Flash configuration field description).

The following fields are available in the FSEC register. Details of the settings are described in the FSEC register description.

Table 30-76. FSEC fields

FSEC field	Description
KEYEN	Backdoor Key Access
MEEN	Mass Erase Capability
FSLACC	Freescale Factory Access
SEC	MCU security

30.4.13.1 FTFE Access by Mode and Security

The following table summarizes how access to the FTFE module is affected by security and operating mode.

Table 30-77. FTFE Access Summary

Operating Mode	MCU Security State	
	Unsecure	Secure
NVM Normal	Full command set	
NVM Special	Full command set	Only the Erase All Blocks and Read 1s All Blocks commands.

30.4.13.2 Changing the Security State

The security state out of reset can be permanently changed by programming the security byte of the flash configuration field. This assumes that you are starting from a mode where the necessary program flash erase and program commands are available and that the region of the program flash containing the flash configuration field is unprotected. If the flash security byte is successfully programmed, its new value takes affect after the next MCU reset.

30.4.13.2.1 Unsecuring the MCU Using Backdoor Key Access

The MCU can be unsecured by using the backdoor key access feature which requires knowledge of the contents of the 8-byte backdoor key value stored in the Flash Configuration Field (see Flash configuration field description). If the FSEC[KEYEN] bits are in the enabled state, the Verify Backdoor Access Key command (see Verify Backdoor Access Key command) can be run which allows the user to present prospective keys for comparison to the stored keys. If the keys match, the FSEC[SEC] bits are changed to unsecure the MCU. The entire 8-byte key cannot be all 0s or all 1s, i.e. 0x0000_0000_0000_0000 and 0xFFFF_FFFF_FFFF_FFFF are not accepted by the Verify Backdoor Access Key command as valid comparison values. While the Verify Backdoor Access Key command is active, program flash memory is not available for read access and returns invalid data.

The user code stored in the program flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN bits are in the enabled state, the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Verify Backdoor Access Key command
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the FSEC[SEC] bits are forced to the unsecure state

An illegal key provided to the Verify Backdoor Access Key command prohibits future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command when a comparison fails.

After the backdoor keys have been correctly matched, the MCU is unsecured by changing the FSEC[SEC] bits. A successful execution of the Verify Backdoor Access Key command changes the security in the FSEC register only. It does not alter the security byte or the keys stored in the Flash Configuration Field (Flash configuration field description). After the next reset of the MCU, the security state of the FTFE module reverts back to the Flash security byte in the Flash Configuration Field. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the program flash protection registers.

If the backdoor keys successfully match, the unsecured MCU has full control of the contents of the Flash Configuration Field. The MCU may erase the sector containing the Flash Configuration Field and reprogram the flash security byte to the unsecure state and change the backdoor keys to any desired value.

30.4.14 Reset Sequence

On each system reset the FTFE module executes a sequence which establishes initial values for the flash block configuration parameters, FPROT, FDPROT, FEPROT, FOPT, and FSEC registers and the FCNFG[SWAP, PFLSH, RAMRDY, EEERDY] bits.

CCIF is cleared throughout the reset sequence. The FTFE module holds off all CPU access for a portion of the reset sequence. Flash reads are possible when the hold is removed. Completion of the reset sequence is marked by setting CCIF which enables flash user commands.

If a reset occurs while any FTFE command is in progress, that command is immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed. Commands and operations do not automatically resume after exiting reset.

Chapter 31
EzPort

31.1 Overview

NOTE

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

EzPort is a serial flash programming interface that allows In-System Programming (ISP) of flash memory contents on a 32 bit general purpose micro-controller. Memory contents can be read, erased and programmed from off-chip in a compatible format to many stand-alone flash memory chips, without necessitating the removal of the micro-controller from the system.

31.1.1 Introduction

The block diagram of the EzPort is as following.

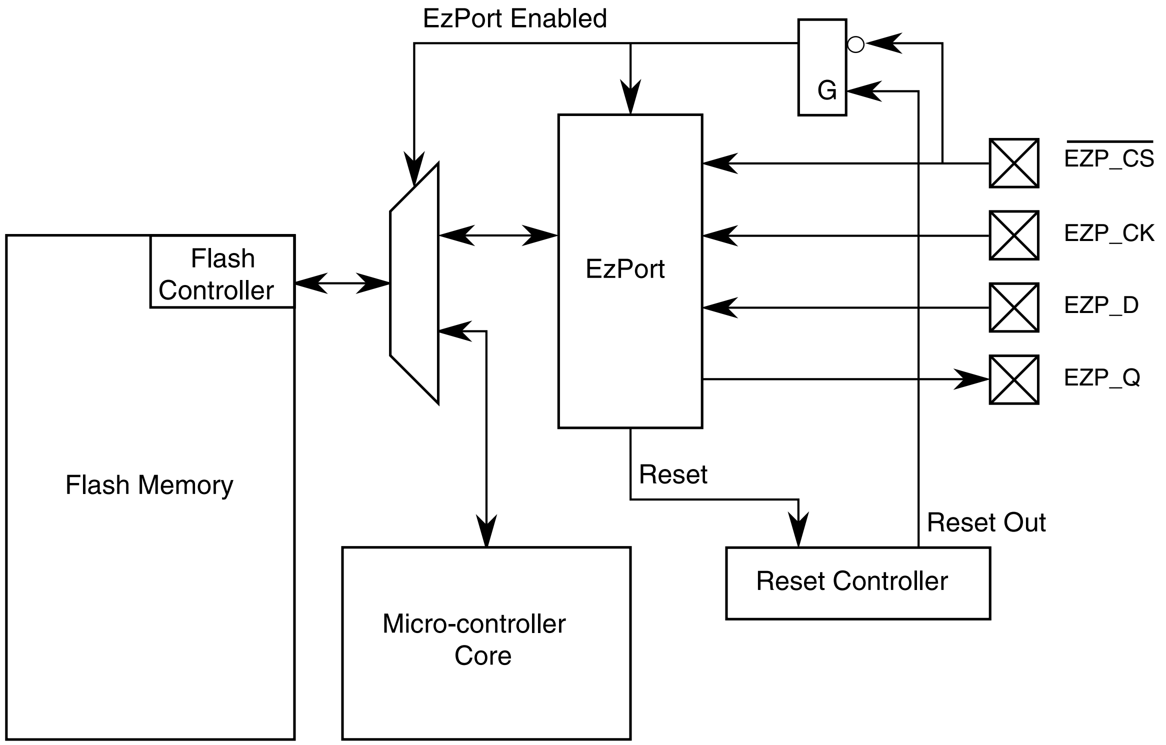


Figure 31-1. EzPort Block Diagram

31.1.2 Features

The EzPort includes the following features:

- Serial interface that is compatible with a subset of the SPI format.
- Able to read, erase and program flash memory.
- Able to reset the micro-controller, allowing it to boot from the flash memory after the memory has been configured.

31.1.3 Modes of Operation

The EzPort can operate in one of two different modes, enabled or disabled.

- Enabled — When enabled, the EzPort steals access to the flash memory, preventing access from other cores or peripherals. The rest of the microcontroller is disabled to avoid conflicts. The flash is configured for NVM Special Mode.
- Disabled — When the EzPort is disabled, the rest of the micro-controller can access flash memory as normal.

The EzPort provides a simple interface to connect an external device to the flash memory on board a 32 bit micro-controller. The interface itself is compatible with the SPI interface (with the EzPort operating as a slave) running in either of the two following modes with data transmitted most significant bit first:

- CPOL = 0, CPHA = 0
- CPOL = 1, CPHA = 1

Commands are issued by the external device to erase, program or read the contents of the flash memory. The serial data out from the EzPort is tri-stated unless data is being driven, allowing the signal to be shared among several different EzPort (or compatible) devices in parallel, provided they have different chip selects.

31.2 External Signal Description

The following table contains a list of EzPort external signals, and the following sections explain them in detail.

Table 31-1. EzPort External Signal Descriptions

Name	Description	I/O
EZP_CK	EzPort Clock	Input
$\overline{\text{EZP\_CS}}$	EzPort Chip Select	Input
EZP_D	EzPort Serial Data In	Input
EZP_Q	EzPort Serial Data Out	Output

31.2.1 EzPort Clock (EZP_CK)

Serial clock for data transfers. The serial data in (EZP_D) and chip select ($\overline{\text{EZP\_CS}}$) are registered on the rising edge of EZP_CK while serial data out (EZP_Q) is driven on the falling edge of EZP_CK.

The maximum frequency of the EzPort clock is half the system clock frequency for all commands except when executing the Read Data or Read FlexRAM commands. When executing these commands, the EzPort clock has a maximum frequency of one-eighth the system clock frequency.

31.2.2 EzPort Chip Select ($\overline{\text{EZP\_CS}}$)

Chip select for signalling the start and end of serial transfers. If $\overline{\text{EZP\_CS}}$ is asserted during and when the micro-controller's reset out signal is negated, then EzPort is enabled out of reset; otherwise it is disabled. After EzPort is enabled, asserting $\overline{\text{EZP\_CS}}$ commences a serial data transfer, which continues until $\overline{\text{EZP\_CS}}$ is negated again. The negation of $\overline{\text{EZP\_CS}}$ indicates the current command is finished and resets the EzPort state machine so that it is ready to receive the next command.

31.2.3 EzPort Serial Data In (EZP_D)

Serial data in for data transfers. EZP_D is registered on the rising edge of EZP_CK. All commands, addresses, and data are shifted in most significant bit first. When the EzPort is driving output data on EZP_Q, the data shifted in EZP_D is ignored.

31.2.4 EzPort Serial Data Out (EZP_Q)

Serial data out for data transfers. EZP_Q is driven on the falling edge of EZP_CK. It is tri-stated unless $\overline{\text{EZP\_CS}}$ is asserted and the EzPort is driving data out. All data is shifted out most significant bit first.

31.3 Command definition

The EzPort receives commands from an external device and translates the commands into flash memory accesses. The following table lists the supported commands.

Table 31-2. EzPort commands

Command	Description	Code	Address Bytes	Data Bytes	Accepted when secure?
WREN	Write Enable	0x06	0	0	Yes
WRDI	Write Disable	0x04	0	0	Yes
RDSR	Read Status Register	0x05	0	1	Yes
READ	Flash Read Data	0x03	3[1]	1+	No
FAST_READ	Flash Read Data at High Speed	0x0B	3[1]	1+[2]	No
SP	Flash Section Program	0x02	3[3]	8 - SECTION[4]	No
SE	Flash Sector Erase	0xD8	3[3]	0	No
BE	Flash Bulk Erase	0xC7	0	0	Yes[5]

Table continues on the next page...

Table 31-2. EzPort commands (continued)

Command	Description	Code	Address Bytes	Data Bytes	Accepted when secure?
RESET	Reset Chip	0xB9	0	0	Yes
WRFCCOB	Write FCCOB Registers	0xBA	0	12	Yes[6]
FAST_RDFCCOB	Read FCCOB registers at high speed	0xBB	0	1 - 12[2]	No
WRFLEXRAM	Write FlexRAM	0xBC	3[1]	4	No
RDFLEXRAM	Read FlexRAM	0xBD	3[1]	1+	No
FAST_RDFLEXRAM	Read FlexRAM at high speed	0xBE	3[1]	1+[2]	No

1. Address must be 32-bit aligned (two LSBs must be zero).
2. One byte of dummy data must be shifted in before valid data is shifted out.
3. Address must be 64-bit aligned (three LSBs must be zero).
4. A section is defined as the smaller of either half the size of FlexRAM or the flash sector size. Total number of data bytes programmed must be a multiple of 8.
5. Bulk Erase is accepted when security is set and only when the BEDIS status field is not set.
6. The flash will be in NVM Special mode, restricting the type of commands that can be executed through WRITE_FCCOB when security is enabled.

31.3.1 Command Descriptions

This section describes the module commands.

31.3.1.1 Write Enable

The Write Enable command (WREN) sets the write enable register bit in the EzPort status register. The write enable bit must be set for a write command (SP, SE, BE, WRFCCOB or WRFLEXRAM) to be accepted. The write enable register bit clears on reset, on a Write Disable command, and at the completion of write command. This command should not be used if a write is already in progress.

31.3.1.2 Write Disable

The Write Disable command (WRDI) clears the write enable register bit in the status register. This command should not be used if a write is already in progress.

31.3.1.3 Read Status Register

The Read Status Register command (RDSR) returns the contents of the EzPort status register.

Table 31-3. EzPort Status Register

	7	6	5	4	3	2	1	0
R	FS	WEF			FLEXRAM	BEDIS	WEN	WIP
W								
Reset:	0/1[1]	0	0	0	0/1[2]	0/1[3]	0	1[4]

1. Reset value reflects the status of flash security out of reset.
2. Reset value reflects FlexNVM flash partitioning. If FlexNVM flash has been paritioned for EEPROM, this bit is set immediately after reset. Note that FLEXRAM is cleared after the EzPort initialization sequence completes, as indicated by clearing of WIP.
3. Reset value reflects if bulk erase is enabled or disabled out of reset
4. Initial value of WIP is 1, but the value clears to 0 after EzPort initialization is complete

Table 31-4. EzPort Status Register Field Descriptions

Field	Description
0 WIP	Write in progress. Status flag that sets after a write command (SP, SE, BE, WRFCCOB, or WRFLEXRAM) is accepted and clears once the flash memory has completed all operations associated with that command as indicated by the Command Complete Interrupt Flag (CCIF) inside the Flash. Also asserted on reset and clears when EzPort initialization is complete. Only the Read Status Register (RDSR) command is accepted while a write is in progress. 0 = Write is not in progress. Accept any command. 1 = Write is in progress. Only accept RDSR command.
1 WEN	Write enable Control bit that must be set before a write command (SP, SE, BE, WRFCCOB, or WRFLEXRAM) is accepted. Is set by the Write Enable (WREN) command and cleared by reset or a Write Disable (WRDI) command. It also clears when the flash memory has completed all operations associated with the command. 0 = Disables the following write command. 1 = Enables the following write command.
2 BEDIS	Bulk erase disable Status flag which indicates if bulk erase (BE) is disabled when Flash is secure. 0 = Bulk Erase is enabled. 1 = Bulk Erase is disabled if the FS bit is also set. Attempts to issue a BE command will result in the WEF flag being set.
3 FLEXRAM	For devices with FlexRAM: FlexRAM mode Status flag that indicates the current mode of the FlexRAM. Only valid when the WIP bit is cleared. 0 = FlexRAM is in RAM mode. RD/WRFLEXRAM command can be used to read/write data in FlexRAM. 1 = FlexRAM is in EEPROM mode. SP command is not accepted. RD/WRFLEXRAM command can be used to read/write data in the FlexRAM.

Table continues on the next page...

Table 31-4. EzPort Status Register Field Descriptions (continued)

Field	Description
6 WEF	Write error flag Status flag that indicates if there has been an error while executing a write command (SP, SE, BE, WRFCCOB, or WRFLEXRAM). The WEF flag will set if either the Flash Access Error Flag (ACCERR) or the Flash Protection Violation Flag (FPVIOL) or the Memory Controller Command Completion Status Flag (MGSTAT0) inside the flash memory is set at the completion of the write command. See the flash memory chapter for further description of these flags and their sources. The WEF flag clears after a Read Status Register (RDSR) command. 0 = No error on previous write command. 1 = Error on previous write command.
7 FS	Flash security Status flag that indicates if the flash is secure. See Table 31-2 for the list of commands which will be accepted when flash is secure. Flash security can be disabled by performing a Bulk Erase (BE) command. 0 = Flash is not secure 1 = Flash is secure.

31.3.1.4 Read Data

The Read Data (READ) command returns data from the flash memory or FlexNVM, depending on the initial address specified in the command word. The initial address must be 32-bit aligned (the two LSBs must be zero).

Data continues being returned for as long as the EzPort chip select ($\overline{\text{EZP\_CS}}$) is asserted, with the address automatically incrementing. In this way, the entire contents of flash can be returned by one command. Attempts to read from an address which does not fall within the valid address range (see Flash memory map for EzPort access) for the flash memory regions returns junk data.

For this command to return the correct data, the EzPort clock (EZP_CK) must run at the internal system clock divided by eight or slower. This command is not accepted if the WEF, WIP, or FS bit in the EzPort status register is set.

31.3.1.5 Read Data at High Speed

The Read Data at High Speed command (FAST_READ) is identical to the READ command, except for the inclusion of a dummy byte following the address bytes and before the first data byte is returned.

This command can be run with an EzPort clock (EZP_CK) frequency of half the internal system clock frequency of the micro-controller or slower. This command is not accepted if the WEF, WIP, or FS bit in the EzPort status register is set.

31.3.1.6 Section Program

The Section Program (SP) command programs up to one section of flash memory which has previously been erased. A section is defined as the smaller of the flash sector size or half the size of the FlexRAM/Programming Acceleration RAM. The starting address of the memory to program is sent after the command word and must be a 64-bit aligned address (the three LSBs must be zero).

As data is shifted in, the EzPort buffers the data in FlexRAM/Programming Acceleration RAM before executing a 'Program Section' command within the flash (see Flash Block Guide for more detail). For this reason, the number of bytes to program must be a multiple of 8 and up to one flash section can be programmed at a time.

Attempts to program more than one section, across a sector boundary or from an initial address which does not fall within the valid address range (see Flash memory map for EzPort access) for the flash causes the WEF flag to set.

For devices with FlexRAM: This command requires the FlexRAM to be configured for traditional RAM operation. By default, after entering EzPort mode, the FlexRAM is configured for traditional RAM operation. If the user reconfigures FlexRAM for EEPROM operation (see Flash Memory chapter for details on how FlexRAM function is modified), then the user should use the WRFCCOB command to configure FlexRAM back to traditional RAM operation before issuing a SP command.

This command is not accepted if the WEF, WIP, FLEXRAM, or FS bit is set or if the WEN bit is not set in the EzPort status register.

31.3.1.7 Sector Erase

The Sector Erase (SE) command erases the contents of one sector of flash memory. The three byte address sent after the command byte can be any address within the sector to erase, but must be a 64-bit aligned address (the three LSBs must be zero). Attempts to erase from an initial address which does not fall within the valid address range (see Flash memory map for EzPort access) for the flash results in the WEF flag being set.

This command is not accepted if the WEF, WIP or FS bit is set or if the WEN bit is not set in the EzPort status register.

31.3.1.8 Bulk Erase

The Bulk Erase (BE) command erases the entire contents of flash memory, ignoring any protected sectors or flash security. Flash security is disabled upon successful completion of the BE command.

Attempts to issue a BE command while the BEDIS and FS bits are set results in the WEF flag being set in the EzPort status register. Also, this command is not accepted if the WEF or WIP bit is set or if the WEN bit is not set in the EzPort status register.

31.3.1.9 EzPort Reset Chip

The Reset Chip (RESET) command forces the chip into the reset state. If the EzPort chip select (EZP_CS) pin is asserted at the end of the reset period then EzPort is enabled; otherwise, it is disabled. This command allows the chip to boot up from flash memory after it has been programmed by an external source.

This command is not accepted if the WIP bit is set in the EzPort status register.

31.3.1.10 Write FCCOB Registers

The Write FCCOB Registers (WRFCCOB) command allows the user to write to the flash common command object registers and execute any command allowed by the flash.

NOTE

> The flash is configured in NVM special mode, restricting which commands can be executed by the flash when security is enabled.

After receiving 12 bytes of data, EzPort writes the data to the FCCOB 0-B registers in the flash and then automatically launches the command within the flash. If greater or less than 12 bytes of data is received, this command has unexpected results and may result in the WEF flag being set.

This command is not accepted if the WEF or WIP bit is set or if the WEN bit is not set in the EzPort status register.

31.3.1.11 Read FCCOB Registers at High Speed

The Read FCCOB Registers at High Speed (FAST_RDFCCOB) command allows the user to read the contents of the flash common command object registers. After receiving the command, EzPort waits for one dummy byte of data before returning FCCOB register data starting at FCCOB 0 and ending with FCCOB B.

This command can be run with an EzPort clock (EZP_CK) frequency half the internal system clock frequency of the microcontroller or slower. Attempts to read greater than 12 bytes of data returns unknown data. This command is not accepted if the WEF, WIP, or FS fields in the EzPort status register are 1.

31.3.1.12 Write FlexRAM

This command is only applicable for devices with FlexRAM.

The Write FlexRAM (WRFLEXRAM) command allows the user to write four bytes of data to the FlexRAM. If the FlexRAM is configured for EEPROM operation, the WRFLEXRAM command can effectively be used to create data records in the EEPROM flash memory.

By default, after entering EzPort mode, the FlexRAM is configured for traditional RAM operation and functions as direct RAM. The user can alter the FlexRAM configuration by using WRFCCOB to execute a Set FlexRAM or Program Partition command within the flash.

The address of the FlexRAM location to be written is sent after the command word and must be a 32-bit aligned address (the two LSBs must be zero). Attempts to write an address which does not fall within the valid address range for the FlexRAM results in the value of the WEF flag being 1. See Flash memory map for EzPort access for more information.

After receiving four bytes of data, EzPort writes the data to the FlexRAM. If greater or less than four bytes of data is received, this command has unexpected results and may result in the value of the WEF flag being 1.

This command is not accepted if the WEF, WIP or FS fields are 1 or if the WEN field is 0 in the EzPort status register.

31.3.1.13 Read FlexRAM

This command is only applicable for devices with FlexRAM.

The Read FlexRAM (RDFLEXRAM) command returns data from the FlexRAM. If the FlexRAM is configured for EEPROM operation, the RDFLEXRAM command can effectively be used to read data from EEPROM flash memory.

Data continues being returned for as long as the EzPort chip select ($\overline{\text{EZP\_CS}}$) is asserted, with the address automatically incrementing. In this way, the entire contents of FlexRAM can be returned by one command.

The initial address must be 32-bit aligned (the two LSBs must be zero). Attempts to read from an address which does not fall within the valid address range for the FlexRAM returns unknown data. See Flash memory map for EzPort access for more information.

For this command to return the correct data, the EzPort clock (EZP_CK) must run at the internal system clock divided by eight or slower. This command is not accepted if the WEF, WIP, or FS fields in the EzPort status register are set.

31.3.1.14 Read FlexRAM at High Speed

This command is only applicable for devices with FlexRAM.

The Read FlexRAM at High Speed (FAST_RDFLEXRAM) command is identical to the RDFLEXRAM command, except for the inclusion of a dummy byte following the address bytes and before the first data byte is returned.

This command can be run with an EzPort clock (EZP_CK) frequency up to and including half the internal system clock frequency of the microcontroller. This command is not accepted if the WEF, WIP, or FS fields in the EzPort status register are set.

31.4 Flash memory map for EzPort access

The following table shows the flash memory map for access through EzPort.

NOTE

The flash block address map for access through EzPort may not conform to the system memory map. Changes are made to allow the EzPort address width to remain 24 bits.

Table 31-5. Flash Memory Map for EzPort Access

Valid start address	Size	Flash block	Valid commands
0x0000_0000	See device's chip configuration details	Flash	READ, FAST_READ, SP, SE, BE

Table continues on the next page...

Table 31-5. Flash Memory Map for EzPort Access (continued)

Valid start address	Size	Flash block	Valid commands
0x0080_0000	See device's chip configuration details	FlexNVM	READ, FAST_READ, SP, SE, BE
0x0000_0000	See device's chip configuration details	FlexRAM	RDFLEXRAM, FAST_RDFLEXRAM, WRFLEXRAM, BE

Chapter 32
NAND Flash Controller (NFC)

32.1 Introduction

NOTE

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

The NAND flash controller (NFC) interfaces to standard NAND flash memory devices. It is composed of various control logic units and a 9 KB SRAM buffer. The NFC provides a glueless interface to 8- and 16-bit NAND flash devices with page sizes of 512 bytes, 2 KB, 4 KB, and 8 KB.

Throughout this chapter the following terms are used:

- Block — (specified by device) smallest erasable unit in a NAND device, consisting of multiple pages
- Page — (specified by device) unit of flash data containing main and spare areas
- Main area of a page — stores data
- Spare area of a page — stores ECC and other software information
- Sector — an elementary transfer unit
    - For devices with pages of 2KB and smaller, this is the same size of the page
    - For devices with pages larger than 2KB, the pages are split into multiple virtual pages. In this case, the sector size is the size of the virtual page
- Virtual page — is the physical page size divided by the splitting factor, NFC_CFG[PAGECNT]

- ECC — error-correcting code
- BCH (Bose Chaudhuri Hocquenghem) — cyclic error-correcting code that corrects multi-bit errors

32.1.1 Block Diagram

The following is a block diagram of the NAND flash controller.

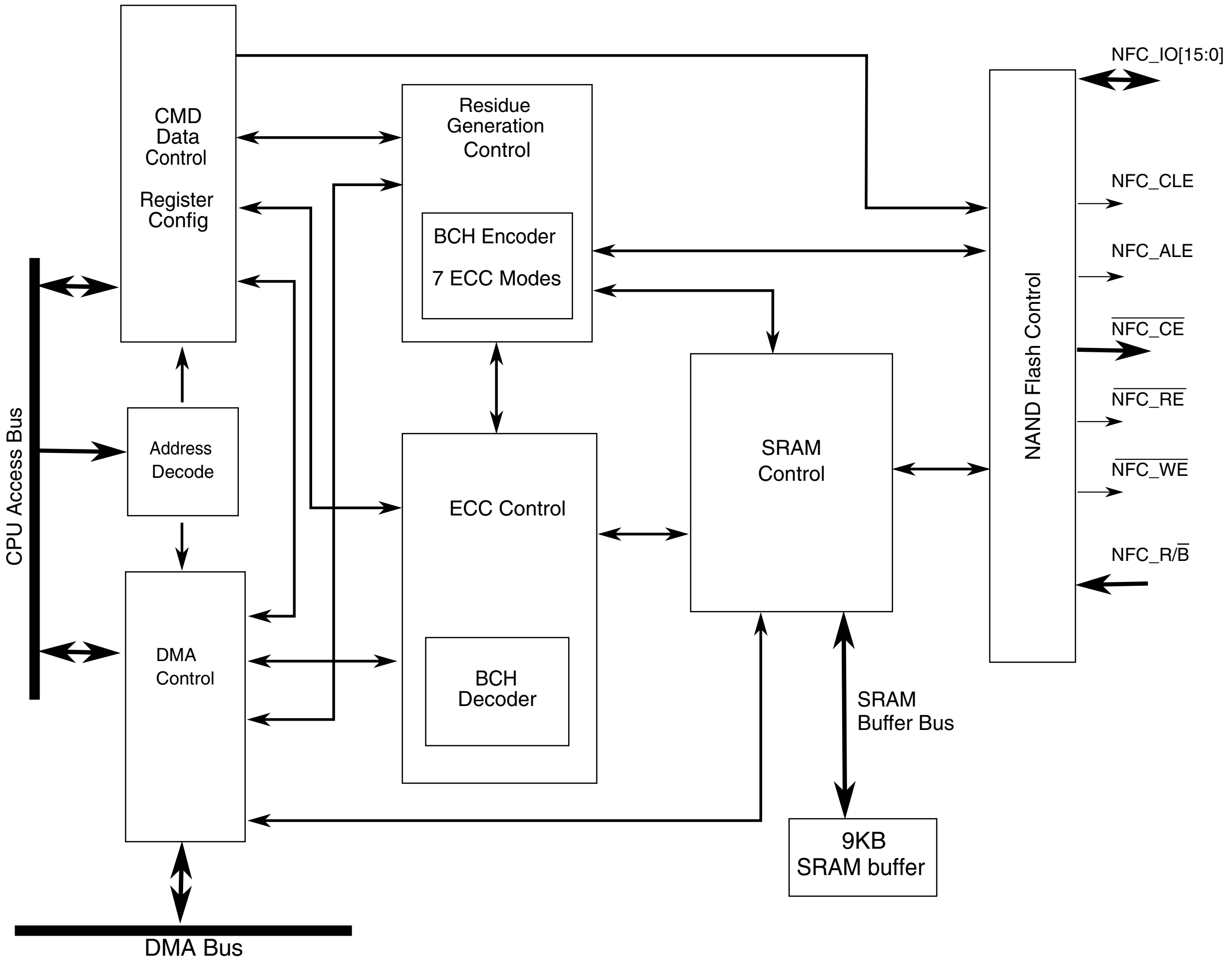


Figure 32-1. NAND Flash Controller Block Diagram

32.1.2 Features

The NAND flash controller includes the following features:

- 8- and 16-bit NAND flash interface

- 9 KB RAM buffer
  - Memory-mapped registers and SRAM buffer
- Supports all NAND flash products regardless of density/organization
- Supports flash device commands, such as page read, page program, reset, block erase, read status, read ID, copy-back, multiplane read/program, interleaved read/program, random input/output, read in EDO mode.
- Integrated DMA engine
  - Two configurable DMA channels
    - Use DMA channel 1 only to read/write a page for main and spare area of a page
    - Use DMA channel 1 to read/write the main area of a page, and DMA channel 2 for the spare area
- ECC mode
  - In ECC mode, NFC supports 4/6/8/12/16/24/32-bit error correction.
  - ECC mode can be bypassed.

32.2 External Signal Description

The signals shown in the next table are used to control NAND flash device.

Table 32-1. NFC Signal Properties

Name	Function	I/O	Reset
NFC_ALE	Flash address latch enable	O	1
$\overline{\text{NFC\_CE}}$	Flash chip enable	O	1
NFC_CLE	Flash command latch enable	O	1
NFC_R/$\overline{\text{B}}$	Flash ready/busy	I	Pull up[1]
$\overline{\text{NFC\_RE}}$	Flash read enable	O	1
$\overline{\text{NFC\_WE}}$	Flash write enable	O	1
NFC_IO[15:0]	Flash data bus	I/O	—

[1] Need to configure both PE and PS bit to 1'b1 of pin control register PORTC_PCR18 to make NFC_R/B pull up, when PTC18 is configured to MUX=6 (NFC_R/B).

32.3 Memory Map/Register Definition

This section defines the NAND flash controller's registers.

NFC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400A_BF00	Flash command 1 (NFC_CMD1)	32	R/W	30FF_0000h	32.3.1/ 821
400A_BF04	Flash command 2 (NFC_CMD2)	32	R/W	007E_E000h	32.3.2/ 821
400A_BF08	Column address (NFC_CAR)	32	R/W	0000_0000h	32.3.3/ 822
400A_BF0C	Row address (NFC_RAR)	32	R/W	1100_0000h	32.3.4/ 823
400A_BF10	Flash command repeat (NFC_RPT)	32	R/W	0000_0000h	32.3.5/ 824
400A_BF14	Row address increment (NFC_RAI)	32	R/W	0000_0001h	32.3.6/ 824
400A_BF18	Flash status 1 (NFC_SR1)	32	R	0000_0000h	32.3.7/ 825
400A_BF1C	Flash status 2 (NFC_SR2)	32	R	0000_0000h	32.3.8/ 825
400A_BF20	DMA channel 1 address (NFC_DMA1)	32	R/W	0000_0000h	32.3.9/ 826
400A_BF24	DMA configuration (NFC_DMACFG)	32	R/W	0000_0000h	32.3.10/ 826
400A_BF28	Cach swap (NFC_SWAP)	32	R/W	0FFE_ 0FFEh	32.3.11/ 827
400A_BF2C	Sector size (NFC_SECSZ)	32	R/W	0000_0420h	32.3.12/ 828
400A_BF30	Flash configuration (NFC_CFG)	32	R/W	000E_A631h	32.3.13/ 828
400A_BF34	DMA channel 2 address (NFC_DMA2)	32	R/W	0000_0000h	32.3.14/ 830
400A_BF38	Interrupt status (NFC_ISR)	32	R/W	6000_0000h	32.3.15/ 830

32.3.1 Flash command 1 (NFC_CMD1)

Address: NFC_CMD1 is 400A_8000h base + 3F00h offset = 400A_BF00h

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	BYTE2								BYTE3								0															
W																																
Reset	0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NFC_CMD1 field descriptions

Field	Description
31–24 BYTE2	Second command byte that may be sent to the flash device
23–16 BYTE3	Third command byte that may be sent to the flash device
15–0 Reserved	This read-only field is reserved and always has the value zero.

32.3.2 Flash command 2 (NFC_CMD2)

Address: NFC_CMD2 is 400A_8000h base + 3F04h offset = 400A_BF04h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BYTE1								CODE[0:8]							
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CODE[7:0]								0					BUFNO		BUSY_ START
W																
Reset	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

NFC_CMD2 field descriptions

Field	Description
31–24 BYTE1	First command byte that may be sent to the flash device
23–8 CODE	User-defined flash operation sequencer

Table continues on the next page...

NFC_CMD2 field descriptions (continued)

Field	Description
	Each bit indicates a certain action. If the bit is set, the corresponding action is executed after writing 1 to START. The following are some configuration examples (other sequences are possible): 0111_1110_1110_0000 Read data (BYTE1, 5x Address, BYTE2, R/$\overline{B}$, read data) 1111_1111_1101_1000 Write page (DMA,BYTE1, 5x Address, write data, BYTE2, R/$\overline{B}$, BYTE3, read status) 0100_1110_1101_1000 Block erase (BYTE1, 3x Address, BYTE2, R/$\overline{B}$, BYTE3, read status) 0100_1000_0000_0100 Read ID (BYTE1, 1x Address, read ID) 0100_0000_0100_0000 Reset (BYTE1, R/$\overline{B}$) 0111_1110_0000_0000 CMD+address (BYTE1, 5xaddress) 1111_1111_1100_0000 Write page burst (DMA,BYTE1,5xAddress, write data, BYTE2,R/$\overline{B}$)
7–3 Reserved	This read-only field is reserved and always has the value zero.
2–1 BUFNO	Internal buffer number used for this command
0 BUSY_START	Busy indicator and start command This busy indicator is repeated in the NFC_ISR register. **NOTE:** Read to this bitfield indicates BUSY whereas write indicates START. 0 During reads, flash controller is idle and it is okay to send next command. During writes, no action. 1 During reads, command execution is busy. During writes, start command execution.

32.3.3 Column address (NFC_CAR)

Address: NFC_CAR is 400A_8000h base + 3F08h offset = 400A_BF08h

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																BYTE2								BYTE1							
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

NFC_CAR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–8 BYTE2	Second byte of column address
7–0 BYTE1	First byte of column address

32.3.4 Row address (NFC_RAR)

Address: NFC_RAR is 400A_8000h base + 3F0Ch offset = 400A_BF0Ch

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		CS1	CS0	0		RB1	RB0	BYTE3								BYTE2								BYTE1							
W	▨				▨																											
Reset	0	0	0	1	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	0	0	0

NFC_RAR field descriptions

Field	Description
31–30 Reserved	This read-only field is reserved and always has the value zero.
29 CS1	Chip select 1 enable 0 $\overline{\text{NFC\_CE1}}$ is disabled 1 $\overline{\text{NFC\_CE1}}$ is enabled
28 CS0	Chip select 0 enable 0 $\overline{\text{NFC\_CE0}}$ is disabled 1 $\overline{\text{NFC\_CE0}}$ is enabled
27–26 Reserved	This read-only field is reserved and always has the value zero.
25 RB1	Ready/busy 1 enable Determines if NFC_R/ $\overline{\text{B}}$ 1 is waited on a wait for R/ $\overline{\text{B}}$ command. If an equal number of $\overline{\text{NFC\_CE}}$ and NFC_R/ $\overline{\text{B}}$ lines are used, the CS *n* and RB *n* fields must contain identical values. If only one NFC_R/ $\overline{\text{B}}$ is used, then CS *n* determines the true chip select, and this field is always 0. 0 NFC_R/ $\overline{\text{B}}$ 1 is disabled 1 NFC_R/ $\overline{\text{B}}$ 1 is enabled
24 RB0	Ready/busy 0 enable Determines if NFC_R/ $\overline{\text{B}}$ 0 is waited on a wait for R/ $\overline{\text{B}}$ command. If an equal number of $\overline{\text{NFC\_CE}}$ and NFC_R/ $\overline{\text{B}}$ lines are used, the CS *n* and RB *n* fields must contain identical values. If only one NFC_R/ $\overline{\text{B}}$ is used, then CS *n* determines the true chip select, and this field is always 1. 0 NFC_R/ $\overline{\text{B}}$ 0 is disabled 1 NFC_R/ $\overline{\text{B}}$ 0 is enabled
23–16 BYTE3	Third byte of row address
15–8 BYTE2	Second byte of row address
7–0 BYTE1	First byte of row address

32.3.5 Flash command repeat (NFC_RPT)

Address: NFC_RPT is 400A_8000h base + 3F10h offset = 400A_BF10h

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																COUNT															
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

NFC_RPT field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 COUNT	16-bit repeat count Determines how many times NFC_CMD2[CODE] is executed. If 0 or 1, the flash command is executed once.

32.3.6 Row address increment (NFC_RAI)

When auto-increment of row address is enabled (NFC_CFG[AIAD] = 1), the row address is incremented as follows:

```
new{NFC_RAR[BYTE3, BYTE2, BYTE1]} =
                                {NFC_RAR[BYTE3], NFC_RAR[BYTE2], NFC_RAR[BYTE1]} +
{NFC_RAI[INC3, INC2, INC1]}
```

Address: NFC_RAI is 400A_8000h base + 3F14h offset = 400A_BF14h

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								INC3								INC2								INC1							
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	1

NFC_RAI field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value zero.
23–16 INC3	Increment for the third byte of row address
15–8 INC2	Increment for the second byte of row address
7–0 INC1	Increment for the first byte of row address

32.3.7 Flash status 1 (NFC_SR1)

Address: NFC_SR1 is 400A_8000h base + 3F18h offset = 400A_BF18h

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	ID1								ID2								ID3								ID4							
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

NFC_SR1 field descriptions

Field	Description
31–24 ID1	First byte returned by read ID command
23–16 ID2	Second byte returned by read ID command
15–8 ID3	Third byte returned by read ID command
7–0 ID4	Fourth byte returned by read ID command

32.3.8 Flash status 2 (NFC_SR2)

Address: NFC_SR2 is 400A_8000h base + 3F1Ch offset = 400A_BF1Ch

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	ID5								0																STATUS1							
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

NFC_SR2 field descriptions

Field	Description
31–24 ID5	Fifth byte returned by read ID command
23–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 STATUS1	Byte returned by read status command

32.3.9 DMA channel 1 address (NFC_DMA1)

Address: NFC_DMA1 is 400A_8000h base + 3F20h offset = 400A_BF20h

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	ADDRESS																															
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

NFC_DMA1 field descriptions

Field	Description
31–0 ADDRESS	DMA channel 1 address. DMA channel 1 address, it is 8-byte aligned.

32.3.10 DMA configuration (NFC_DMACFG)

Address: NFC_DMACFG is 400A_8000h base + 3F24h offset = 400A_BF24h

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	COUNT1												COUNT2							OFFSET2				0							ACT1	ACT2
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

NFC_DMACFG field descriptions

Field	Description
31–20 COUNT1	Number of bytes to be transferred by DMA channel 1. It should be multiple of 8 bytes.
19–13 COUNT2	Number of bytes to be transferred by DMA channel 2. It should be multiple of 8 bytes.
12–9 OFFSET2	256-byte offset for DMA channel 2. DMA channel 2 transfer starts at this offset count x 256 bytes. For example, if OFFSET2 = 0x2, DMA channel 2 transfer starts at 0x200.
8–2 Reserved	This read-only field is reserved and always has the value zero.
1 ACT1	DMA channel 1 status 0 Inactive 1 Active, and transfers to memory when triggered
0 ACT2	DMA channel 2 status 0 Inactive 1 Active, and transfers to memory when triggered

32.3.11 Cach swap (NFC_SWAP)

When DMA transfers data to/from the NFC cache (NFC SRAM buffer), or when the CPU reads or writes data to/from the NFC cache via the internal bus, all accesses that go to NFC_SWAP[ADDR1] are directed to NFC_SWAP[ADDR2]. Likewise, all accesses that go to NFC_SWAP[ADDR2] are directed to NFC_SWAP[ADDR1].

The feature allows the bad block marker in the first position of the spare area of a page. Because of the way the flash controller interleaves data and ECC bytes on flash devices with page sizes larger than 2 KB, the position of the bad block marker is shifted, and does not appear in the first position of the spare area of the page. The cache swap feature allows consistent swapping of the actual bad block line with the expected bad block line, and causes the operating system to get the bad block marker in the position where it is expected.

Address: NFC_SWAP is 400A_8000h base + 3F28h offset = 400A_BF28h

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				ADDR1											0					ADDR2											0
W																																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0

NFC_SWAP field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value zero.
27–17 ADDR1	Lower swap address
16–12 Reserved	This read-only field is reserved and always has the value zero.
11–1 ADDR2	Upper swap address
0 Reserved	This read-only field is reserved and always has the value zero.

32.3.12 Sector size (NFC_SECSZ)

Address: NFC_SECSZ is 400A_8000h base + 3F2Ch offset = 400A_BF2Ch

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																			SIZE												
W																																
Reset	0	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	1	0	0	0	0	0

NFC_SECSZ field descriptions

Field	Description
31–13 Reserved	This read-only field is reserved and always has the value zero.
12–0 SIZE	Size in bytes of one elementary transfer unit For devices with pages of 2KB and smaller, this is the physical size of the page in bytes (data bytes + ECC bytes) transferred in one page. When pages are larger than 2KB, they must be split in multiple virtual pages. In this case, the sector size is the size of the virtual page. The virtual page size is the physical size divided by the splitting factor, NFC_CFG[PAGECNT]. **NOTE:** If only a part of a page to be programmed or read, SIZE can be set to the number of affected bytes, not the page size. Then, ECC and DMA (data bytes) are all performed on the number of bytes, indicated by SIZE. **NOTE:** For 16-bit data width flash devices, only odd SIZE is supported. If SIZE is even number, the real implemented size is SIZE – 1. So, write size + 1 to this field. For example, if SIZE = 1, no data is written or read. **NOTE:** When programming NAND memory using the ECC feature, ensure that SIZE is equal to the default value (data + ECC bytes).

32.3.13 Flash configuration (NFC_CFG)

Address: NFC_CFG is 400A_8000h base + 3F30h offset = 400A_BF30h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STOPWERR	ECCAD[11:3]									ECCSRAM	DMAREQ	ECCMODE			FAST
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDCNT			TIMEOUT					BITWIDTH	0	AIAD	AIBN	PAGECNT			
W																
Reset	1	0	1	0	0	1	1	0	0	0	1	1	0	0	0	1

NFC_CFG field descriptions

Field	Description
31 STOPWERR	0 No stop on write error 1 Auto-sequencer stops on a write error
30–22 ECCAD[11:3]	Byte address in SRAM where ECC status is written.
21 ECCSRAM	0 Do not write ECC status to SRAM 1 Write ECC status to SRAM
20 DMAREQ	0 Do not transfer sector after ECC done 1 After ECC done, transfer sector using DMA
19–17 ECCMODE	000 No correction, ECC bypass 001 4-error correction (8 ECC bytes) 010 6-error correction (12 ECC bytes) 011 8-error correction (15 ECC bytes) 100 12-error correction (23 ECC bytes) 101 16-error correction (30 ECC bytes) 110 24-error correction (45 ECC bytes) 111 32-error correction (60 ECC bytes)
16 FAST	See the "Fast Flash Configuration for EDO" section for more details. 0 Slow flash timing. Clock in read data on rising edge of read strobe 1 Fast flash timing. Clock in read data a half clock later than rising edge of read strobe
15–13 IDCNT	Number of bytes that are read for the read id command.
12–8 TIMEOUT	The number of flash_clk cycles from NFC_WE high to either: • NAND flash busy (t_{WB}), or • NFC_RE low (t_{WHR}) After the last command is issued to flash, before sampling NFC_R/ $\overline{B}$, the NFC must wait t_{WB} clocks. After t_{WB} clocks: • if NFC_R/ $\overline{B}$ is sampled as high, the NFC considers the command to be a timeout, and the flash memory is idle. The NFC can issue new commands to the flash memory. • if NFC_R/ $\overline{B}$ is sampled as low, the NAND flash memory is busy. When reading the status or ID from the NAND flash memory, after the last command is issued to flash, the NFC must wait for t_{WHR} cycles. The NFC then negates NFC_RE to low to read the valid status or ID. **NOTE:** t_{WB} exists in page program/read, block erase, etc. Refer to the NAND flash datasheet for details of t_{WB} and t_{WHR} .
7 BITWIDTH	0 8-bit wide flash mode 1 16-bit wide flash mode
6 Reserved	This read-only field is reserved and always has the value zero.
5 AIAD	0 Do not auto-increment flash row address 1 Auto-increment flash row address
4 AIBN	0 Do not auto-increment buffer number 1 Auto-increment buffer number

Table continues on the next page...

NFC_CFG field descriptions (continued)

Field	Description
3–0 PAGECNT	Number of virtual pages (in one physical flash page) to be programmed or read, etc.

32.3.14 DMA channel 2 address (NFC_DMA2)

Address: NFC_DMA2 is 400A_8000h base + 3F34h offset = 400A_BF34h

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	ADDRESS																															
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

NFC_DMA2 field descriptions

Field	Description
31–0 ADDRESS	DMA channel 2 address. DMA channel 2 address, it is 8-byte aligned.

32.3.15 Interrupt status (NFC_ISR)

Address: NFC_ISR is 400A_8000h base + 3F38h offset = 400A_BF38h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WERR	DONE	IDLE	0	WERRNS	CMDBUSY	RESBUSY	ECCBUSY	DMABUSY	WERREN	DONEEN	IDLEEN	WERRCLR	DONECLR	IDLECLR	0
W													w1c	w1c	w1c	
Reset	0	1	1	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										RESBN		ECCBN		DMABN	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NFC_ISR field descriptions

Field	Description
31 WERR	Write error interrupt Set if an error condition is detected during a flash read status command. Sticky bit.
30 DONE	Done interrupt Set if command processing is done.
29 IDLE	Command idle interrupt Set if command done, and residue engine, ECC engine and DMA engine are idle.
28 Reserved	This read-only field is reserved and always has the value zero.
27 WERRNS	Write error status Set if an error condition was detected during the last flash read status command. Non-sticky bit.
26 CMDBUSY	Command busy Set it command execution busy, cleared otherwise.
25 RESBUSY	Residue engine busy Set if residue engine busy, cleared otherwise.
24 ECCBUSY	ECC engine busy Set if ECC engine busy, cleared otherwise.
23 DMABUSY	DMA engine busy Set if DMA engine busy, cleared otherwise.
22 WERREN	Enable bit for NFC_ISR[WERR]
21 DONEEN	Enable bit for NFC_ISR[DONE]
20 IDLEEN	Enable bit for NFC_ISR[IDLE]
19 WERRCLR	Clear bit for NFC_ISR[WERR]. Writing 1 to this bit clears NFC_ISR[WERR].
18 DONECLR	Clear bit for NFC_ISR[DONE]. Writing 1 to this bit clears NFC_ISR[DONE].
17 IDLECLR	Clear bit for NFC_ISR[IDLE]. Writing 1 to this bit clears NFC_ISR[IDLE].
16–6 Reserved	This read-only field is reserved and always has the value zero.
5–4 RESBN	Residue buffer number Buffer number corresponding with the current residue block task.
3–2 ECCBN	ECC buffer number Buffer number corresponding with the current ECC task.

Table continues on the next page...

NFC_ISR field descriptions (continued)

Field	Description
1–0 DMABN	DMA buffer number Buffer number corresponding with the current DMA task.

32.4 Functional Description

The NFC executes commands on a single or bank of external NAND flash chips. The NFC supports commands such as read, program, reset, erase, status read, read ID.

The NFC block contains a DMA engine and built-in ECC logic. For each read or write, the NFC performs ECC calculations on-the-fly. Two DMA channels are organized for each read or write: one for the main area, and one for the spare area. It is possible to disable the second DMA channel, and transfer main and spare data with only the first DMA channel.

Page size supported is 512, 2K, 4K and 8K bytes. There are 8 different ECC settings provided: 0, 4, 6, 8, 12, 16, 24 and 32 bits errors. They use 0, 8, 12, 15, 23, 30, 45 and 60 ECC bytes. The ECC works on page sizes of 512+spares bytes, 1K+spares bytes, 2K +spares bytes. The ECC algorithm used is a BCH code.

The error corrector can write ECC status to the spare area, since the read is pipelined. This means, while the current page is transferred from flash to buffer, the previous page is ECC corrected, and the page before that is transferred using DMA. Because of the pipelining, it is difficult to inform the CPU in the foreground of ECC errors. To solve this, ECC status is written to the auxiliary area of the sector, and transferred to memory. See Error Corrector Status for more information. It's up to the CPU to inspect the ECC result in memory, and act appropriately.

As described, reads are pipelined. However, writes are flow-through; no advance operations are done during a write. If there is a problem found during a write, the command sequence may be interrupted, and the CPU is informed.

Each page read, page write, page erase, read ID, or read status command sequence needs CPU attention only once. The CPU needs to prepare the DMA to point to the data, write correct values to all registers, and start the command. After command completion, the NFC block may interrupt the CPU.

The block allows command repeat, which is useful for write, read and erase, and allows processing multiple pages with just one command given by the CPU. No bank interleaving is supported during command repeat.

32.4.1 NFC Buffer Memory Space

The next figure shows the organization of the buffer memory space in the NFC. The memory's size is 1152 × 64 bit, and is separated into four buffers, each with incontinuous physical address. For example, buffer 0's physical address is (0x000 + 0x20×*i*) – (0x007+0x20×*i*).

However, when the CPU writes or reads a buffer in non-boot mode, the CPU address is continuous, since there's an address transition inside NFC: sram_physical_addr[13:3] = {cpu_addr[11:3],cpu_addr[13:12]}

So, in non-boot mode, the address ranges are:

- Buffer 0: 0x0000 – 0x08FF
- Buffer 1: 0x1000 – 0x18FF
- Buffer 2: 0x2000 – 0x28FF
- Buffer 3: 0x3000 – 0x38FF

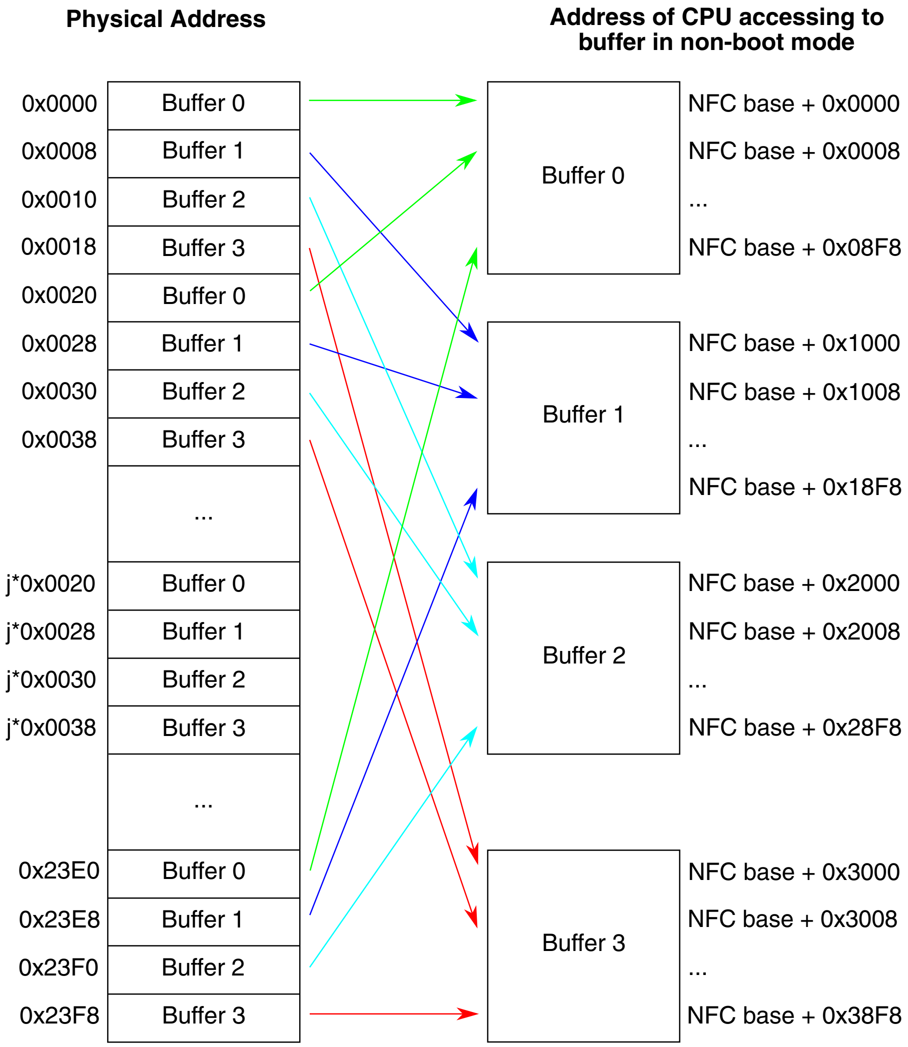


Figure 32-17. NFC Buffer Memory Space

32.4.2 Error Corrector Status

The ECC engine determines if a page is correctable. If correctable, it corrects error bits, and indicates error number. Otherwise, CORFAIL bit is asserted as shown in the next table. For a bad block management strategy to work, it may be necessary for the processor to obtain this information.

The error corrector writes the status word to a byte location to the SRAM buffer, defined by NFC_CFG[ECCAD[11:3]]. It is selectable if the status is written or not with NFC_CFG[ECCSRAM]. If the status is written to the SRAM buffer, it becomes effectively part of the flash data, and is processed like the flash data. Most likely, the status byte is written to memory as part of the page header. Once in memory, the ECC status is visible to the CPU, while CPU parses the rest of the flash header. No interrupt on error or status is available because this increases the interrupt load on the CPU. (The interrupt would be independent of the command done interrupt.) It is not possible to stop reading when ECC failed.

The organization of the status byte is shown here.

Table 32-18. ECC Status Word

Field	Definition
7 CORFAIL	0 Page has been successfully corrected 1 Page is uncorrectable
5–0 ERROR_COUNT	Number of errors that have been corrected in this page

NOTE

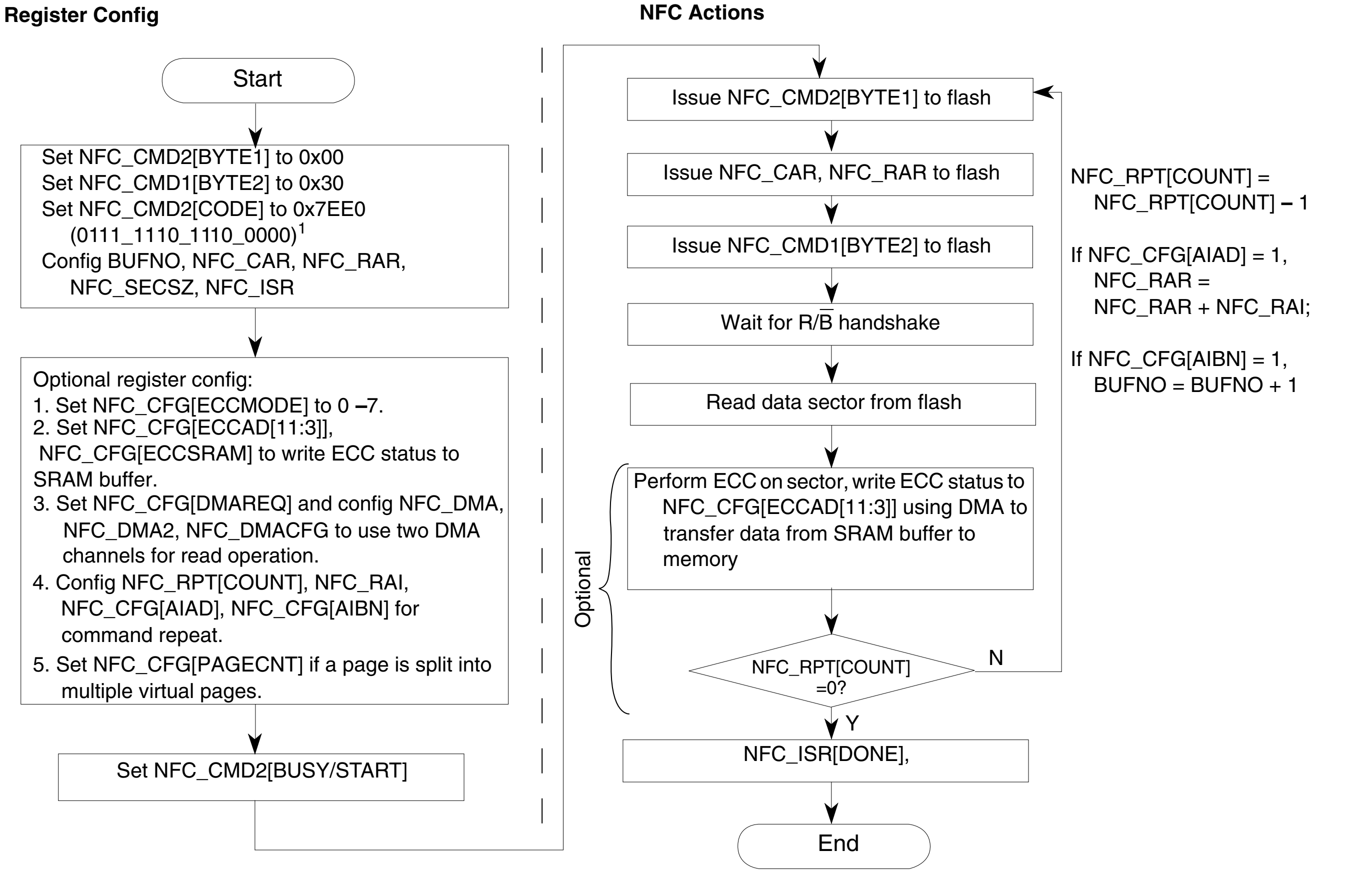
The address of the ECC status byte=ECCAD[11:3]+7.

32.4.3 NFC Basic Commands

NFC basic commands include Page Read, Page Program, Block Erase, Read ID, and Reset.

32.4.3.1 Page Read

This command reads pages from the NAND flash. This figure shows the general flowchart of a read operation.


Register Config
NFC Actions
Start
Set NFC_CMD2[BYTE1] to 0x00
Set NFC_CMD1[BYTE2] to 0x30
Set NFC_CMD2[CODE] to 0x7EE0
(0111_1110_1110_0000)1
Config BUFNO, NFC_CAR, NFC_RAR,
NFC_SECSZ, NFC_ISR
Optional register config:
1. Set NFC_CFG[ECCMODE] to 0 –7.
2. Set NFC_CFG[ECCAD[11:3]],
NFC_CFG[ECCSRAM] to write ECC status to SRAM buffer.
3. Set NFC_CFG[DMAREQ] and config NFC_DMA, NFC_DMA2, NFC_DMACFG to use two DMA channels for read operation.
4. Config NFC_RPT[COUNT], NFC_RAI, NFC_CFG[AIAD], NFC_CFG[AIBN] for command repeat.
5. Set NFC_CFG[PAGECNT] if a page is split into multiple virtual pages.
Set NFC_CMD2[BUSY/START]
Issue NFC_CMD2[BYTE1] to flash
Issue NFC_CAR, NFC_RAR to flash
Issue NFC_CMD1[BYTE2] to flash
Wait for R/B handshake
Read data sector from flash
Perform ECC on sector, write ECC status to NFC_CFG[ECCAD[11:3]] using DMA to transfer data from SRAM buffer to memory
Optional
NFC_RPT[COUNT] =0?
N
Y
NFC_ISR[DONE],
End
NFC_RPT[COUNT] = NFC_RPT[COUNT] – 1
If NFC_CFG[AIAD] = 1, NFC_RAR = NFC_RAR + NFC_RAI;
If NFC_CFG[AIBN] = 1, BUFNO = BUFNO + 1

Note:

[1] COL_ADDR2, NFC_RAR[BYTE3], and NFC_CMD1[BYTE2] (bold) are not necessary for some flash devices. See their data sheets for detail. For example:

If the flash only has one column address, then NFC_CMD2[CODE] = 0110_1110_1110_0000;
If the flash only has two row addresses, then NFC_CMD2[CODE] = 0111_1100_1110_0000;
If flash does not need the second command 0x30 for read, then NFC_CMD2[CODE] = 0110_1110_0110_0000.

Figure 32-18. Flow Chart of Read Operation

The next figure shows a particular case: one page is split into 8 virtual pages (see Organization of the Data in the NAND Flash), and DMA is not used. The SRAM buffer can hold data for four (virtual) pages at most. The CPU must transfer data out of the SRAM buffer after the first four virtual pages are read from flash. Otherwise, the next four virtual pages data overwrite the buffer. So, the read operation has following steps:

- Configure registers as shown in the preceding figure. NFC_CFG[PAGECNT] = 4, start commands, wait for NFC_ISR[DONE]
- CPU reads data from buffer, set NFC_CMD2[CODE] = 0x20 (only enable read data)
- Start commands to read out the next 4 virtual pages, wait for NFC_ISR[DONE]

If DMA is used to transfer data from SRAM buffer to memory instead of CPU, the flow in the preceding figure is used: set NFC_CFG[PAGECNT] = 8, set NFC_CFG[DMAREQ] = 1, configure DMA registers, start commands. A pipeline (Functional Description) controls the read operation.

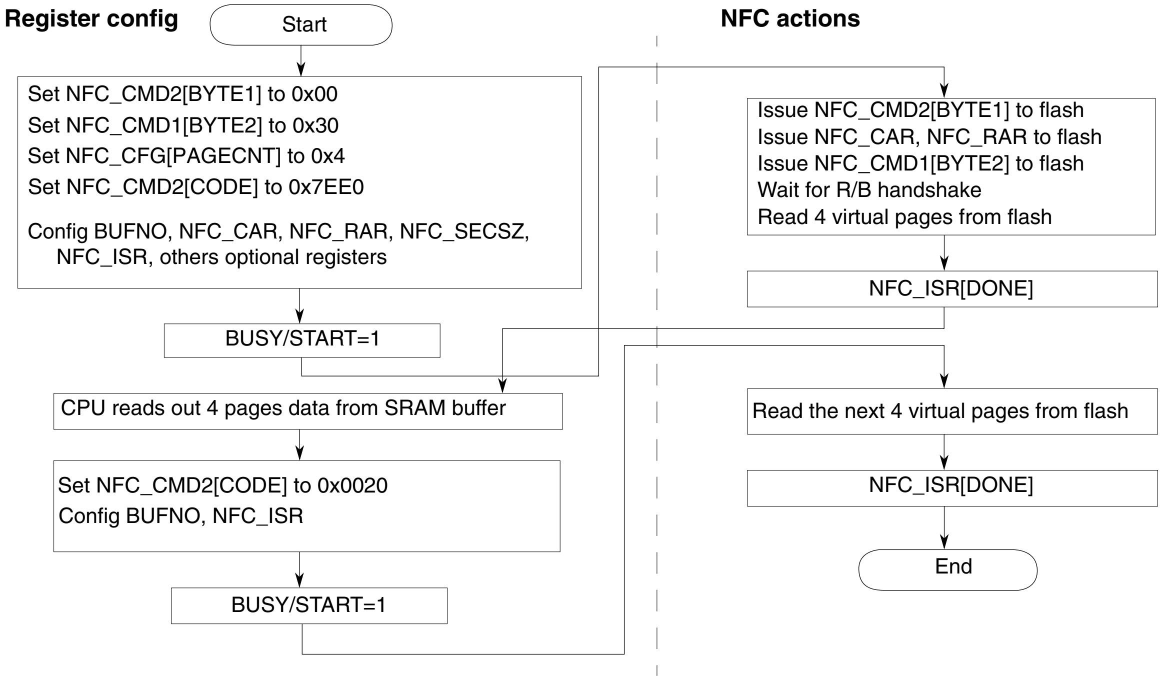

Register config
Start
Set NFC_CMD2[BYTE1] to 0x00
Set NFC_CMD1[BYTE2] to 0x30
Set NFC_CFG[PAGECNT] to 0x4
Set NFC_CMD2[CODE] to 0x7EE0
Config BUFNO, NFC_CAR, NFC_RAR, NFC_SECSZ, NFC_ISR, others optional registers
BUSY/START=1
CPU reads out 4 pages data from SRAM buffer
Set NFC_CMD2[CODE] to 0x0020
Config BUFNO, NFC_ISR
BUSY/START=1
NFC actions
Issue NFC_CMD2[BYTE1] to flash
Issue NFC_CAR, NFC_RAR to flash
Issue NFC_CMD1[BYTE2] to flash
Wait for R/B handshake
Read 4 virtual pages from flash
NFC_ISR[DONE]
Read the next 4 virtual pages from flash
NFC_ISR[DONE]
End

Figure 32-19. Flow Chart of Read Operation, NFC_CFG[PAGECNT] = 8, No DMA

Note

See footnote in Figure 32-18.

32.4.3.2 Page Program

This command programs pages to the NAND flash. The next figure is the general flow of page program operation.

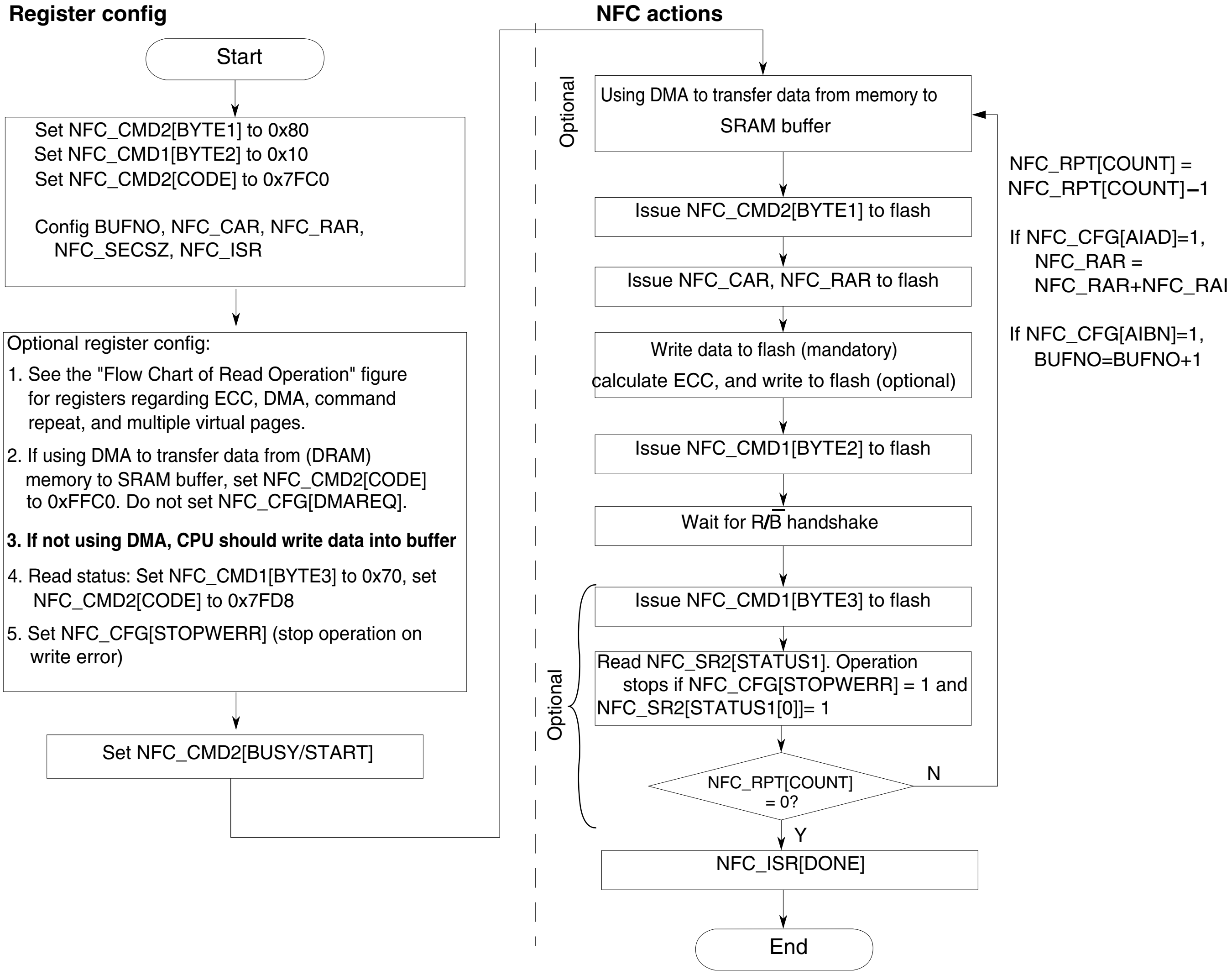

Register config
Start
Set NFC_CMD2[BYTE1] to 0x80
Set NFC_CMD1[BYTE2] to 0x10
Set NFC_CMD2[CODE] to 0x7FC0
Config BUFNO, NFC_CAR, NFC_RAR, NFC_SECSZ, NFC_ISR
Optional register config:
1. See the "Flow Chart of Read Operation" figure for registers regarding ECC, DMA, command repeat, and multiple virtual pages.
2. If using DMA to transfer data from (DRAM) memory to SRAM buffer, set NFC_CMD2[CODE] to 0xFFC0. Do not set NFC_CFG[DMAREQ].
3. If not using DMA, CPU should write data into buffer
4. Read status: Set NFC_CMD1[BYTE3] to 0x70, set NFC_CMD2[CODE] to 0x7FD8
5. Set NFC_CFG[STOPWERR] (stop operation on write error)
Set NFC_CMD2[BUSY/START]
NFC actions
Optional
Using DMA to transfer data from memory to SRAM buffer
Issue NFC_CMD2[BYTE1] to flash
Issue NFC_CAR, NFC_RAR to flash
Write data to flash (mandatory)
calculate ECC, and write to flash (optional)
Issue NFC_CMD1[BYTE2] to flash
Wait for R/B̄ handshake
Optional
Issue NFC_CMD1[BYTE3] to flash
Read NFC_SR2[STATUS1]. Operation stops if NFC_CFG[STOPWERR] = 1 and NFC_SR2[STATUS1[0]]= 1
NFC_RPT[COUNT] = 0?
N
Y
NFC_ISR[DONE]
End
NFC_RPT[COUNT] = NFC_RPT[COUNT]−1
If NFC_CFG[AIAD]=1, NFC_RAR = NFC_RAR+NFC_RAI
If NFC_CFG[AIBN]=1, BUFNO=BUFNO+1

Figure 32-20. Flow Chart of Page Program Operation

Note

COL_ADDR2 and NFC_RAR[BYTE3] (bold) are not necessary for some flash devices. See their data sheets for detail. See the footnote of Figure 32-18.

The next figure shows the particular case which is similar to Figure 32-19. The CPU writes at most four virtual pages of data into the buffer before the first start command. Set NFC_CFG[PAGECNT] to 4 and set NFC_CMD2[CODE] twice:

- First, set it to 0x7F00 (0111_1111_0000_0000). The NFC issues NFC_CMD2[BYTE1], address cycles, four virtual pages data to flash. After NFC_ISR[DONE] is set, the CPU can write the next four virtual pages data into the SRAM buffer.
- Second, set CODE to 0x01C0 (0000_0001_1100_0000). The NFC sends the next four virtual pages of data to flash, issues NFC_CMD1[BYTE2], waits for R/$\overline{B}$ handshake, and waits for NFC_ISR[DONE] to set.

Like the read operation, if DMA transfers data from memory to NFC SRAM buffer (instead of the CPU), the flow in the preceding figure is used and set NFC_CFG[PAGECNT] to 0x8.

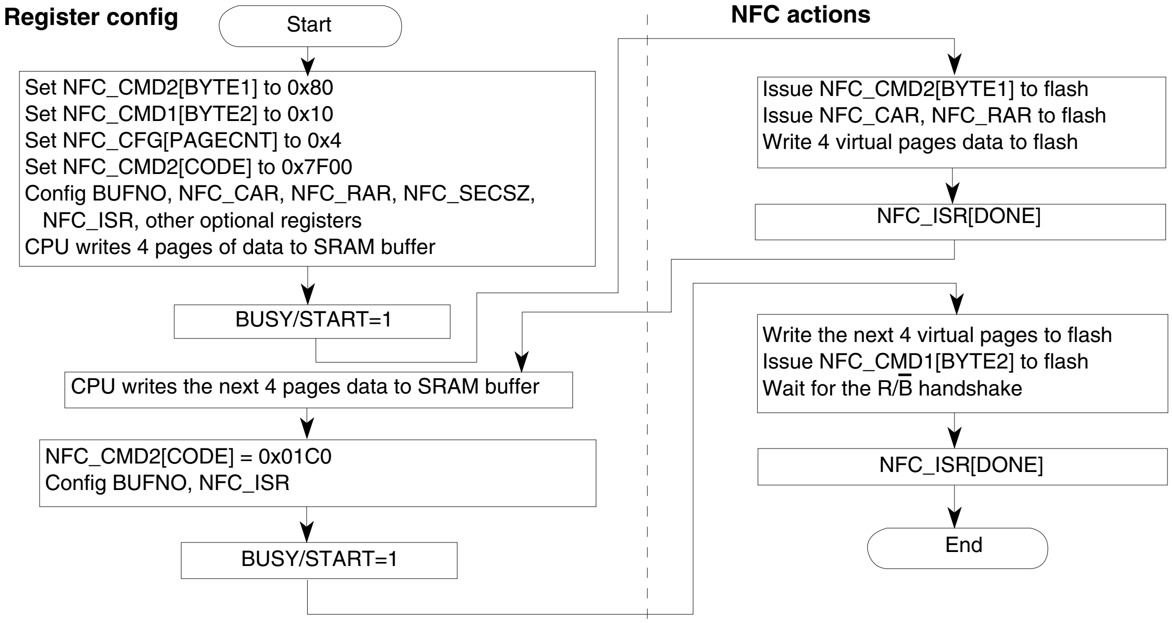

Register config
Start
Set NFC_CMD2[BYTE1] to 0x80
Set NFC_CMD1[BYTE2] to 0x10
Set NFC_CFG[PAGECNT] to 0x4
Set NFC_CMD2[CODE] to 0x7F00
Config BUFNO, NFC_CAR, NFC_RAR, NFC_SECSZ,
NFC_ISR, other optional registers
CPU writes 4 pages of data to SRAM buffer
BUSY/START=1
CPU writes the next 4 pages data to SRAM buffer
NFC_CMD2[CODE] = 0x01C0
Config BUFNO, NFC_ISR
BUSY/START=1
NFC actions
Issue NFC_CMD2[BYTE1] to flash
Issue NFC_CAR, NFC_RAR to flash
Write 4 virtual pages data to flash
NFC_ISR[DONE]
Write the next 4 virtual pages to flash
Issue NFC_CMD1[BYTE2] to flash
Wait for the R/B handshake
NFC_ISR[DONE]
End

Figure 32-21. Flow Chart of Page Program Operation, NFC_CFG[PAGECNT] = 8, No DMA

Note

If you want to read the status after the second 0x10 command, set NFC_CMD1[BYTE3] to 0x70 and NFC_CMD2[CODE] to 0x01D8 (0000_0001_1101_1000). Then, after "Wait for the R/$\overline{B}$ handshake", the NFC issues NFC_CMD1[BYTE3] to flash, and reads the status. If NFC_CFG[STOPWERR] is set and NFC_SR2[STATUS1[0]]=1, operation stops. Otherwise, NFC_ISR[DONE] comes out. The COL_ADDR2 and NFC_RAR[BYTE3] of the first NFC_CMD2[CODE] may not be necessary. See note 1 of Figure 32-18.

32.4.3.3 Block Erase

This command is used to erase blocks.

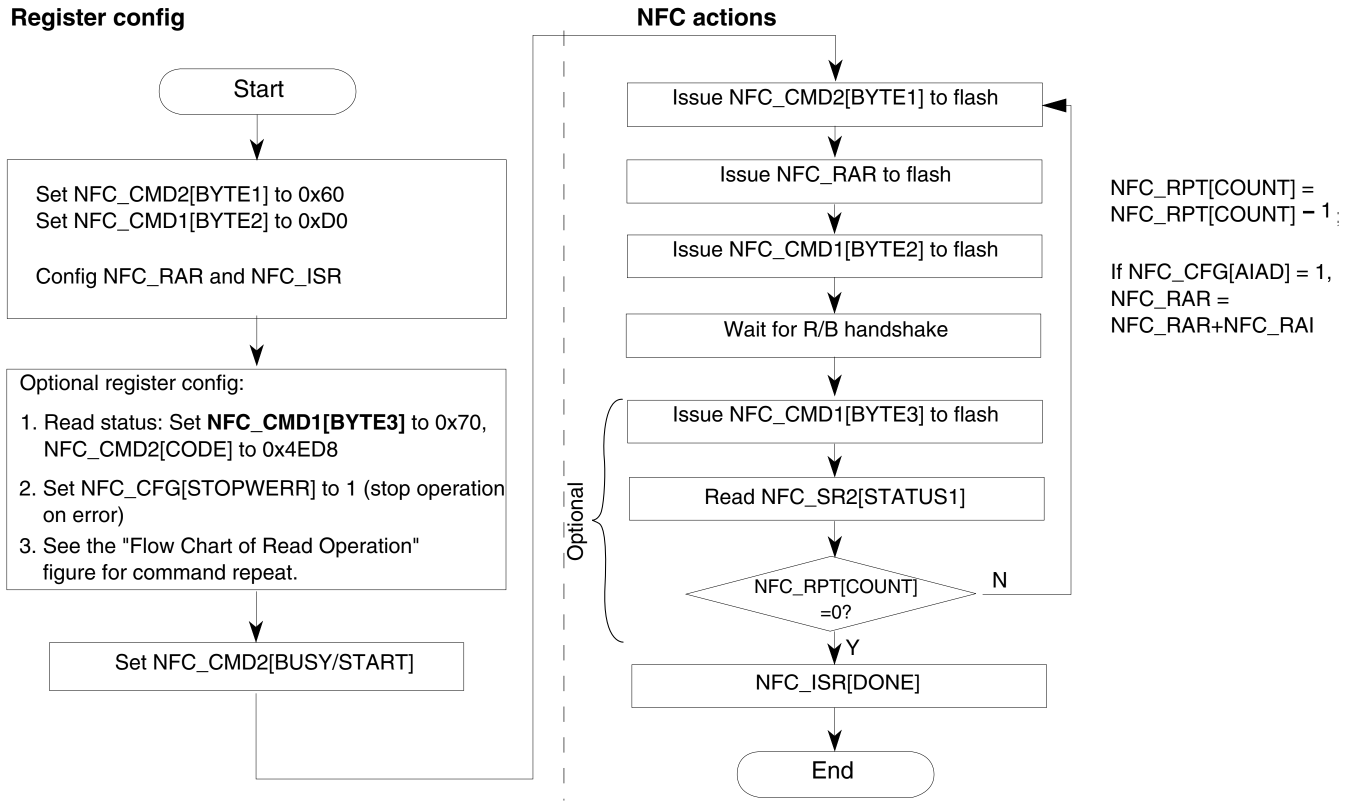

Register config
NFC actions
Start
Set NFC_CMD2[BYTE1] to 0x60
Set NFC_CMD1[BYTE2] to 0xD0
Config NFC_RAR and NFC_ISR
Optional register config:
1. Read status: Set **NFC_CMD1[BYTE3]** to 0x70, NFC_CMD2[CODE] to 0x4ED8
2. Set NFC_CFG[STOPWERR] to 1 (stop operation on error)
3. See the "Flow Chart of Read Operation" figure for command repeat.
Set NFC_CMD2[BUSY/START]
Issue NFC_CMD2[BYTE1] to flash
Issue NFC_RAR to flash
Issue NFC_CMD1[BYTE2] to flash
Wait for R/B handshake
Issue NFC_CMD1[BYTE3] to flash
Read NFC_SR2[STATUS1]
NFC_RPT[COUNT] =0?
N
Y
NFC_ISR[DONE]
End
Optional
NFC_RPT[COUNT] = NFC_RPT[COUNT] – 1
If NFC_CFG[AIAD] = 1, NFC_RAR = NFC_RAR+NFC_RAI

Figure 32-22. Flow Chart of Block Erase Operation

Note

NFC_RAR[BYTE3] (bold) is not necessary for some flash devices. See their data sheets for detail.

Note

If NFC_CFG[STOPWERR] is set and NFC_SR2[STATUS1[0]]=1, operation stops.

32.4.3.4 Read ID

This command reads the flash ID.

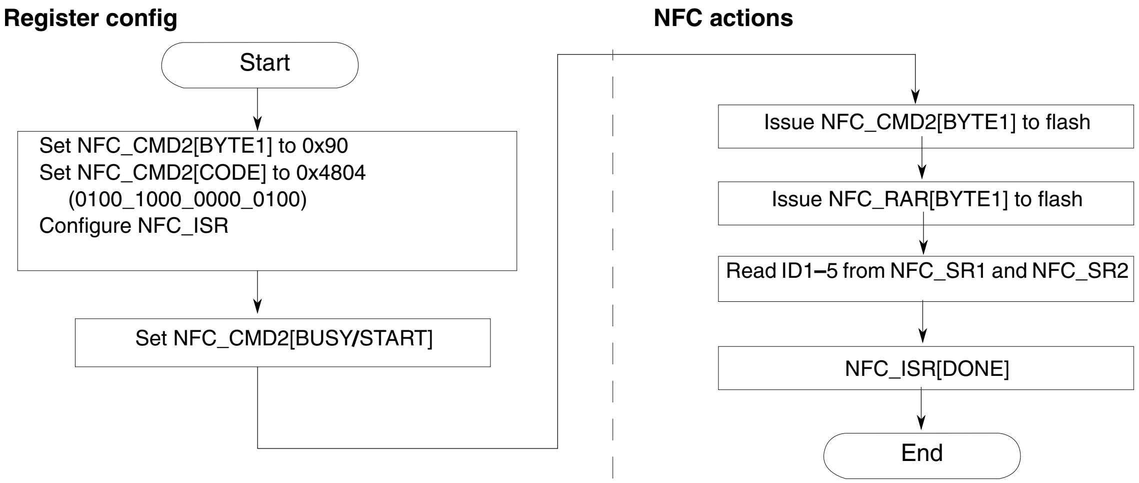

Register config
NFC actions
Start
Set NFC_CMD2[BYTE1] to 0x90
Set NFC_CMD2[CODE] to 0x4804
(0100_1000_0000_0100)
Configure NFC_ISR
Set NFC_CMD2[BUSY/START]
Issue NFC_CMD2[BYTE1] to flash
Issue NFC_RAR[BYTE1] to flash
Read ID1–5 from NFC_SR1 and NFC_SR2
NFC_ISR[DONE]
End

Figure 32-23. Flow Chart of Read ID Operation

32.4.3.5 Reset

This command sends a single reset command to the flash.

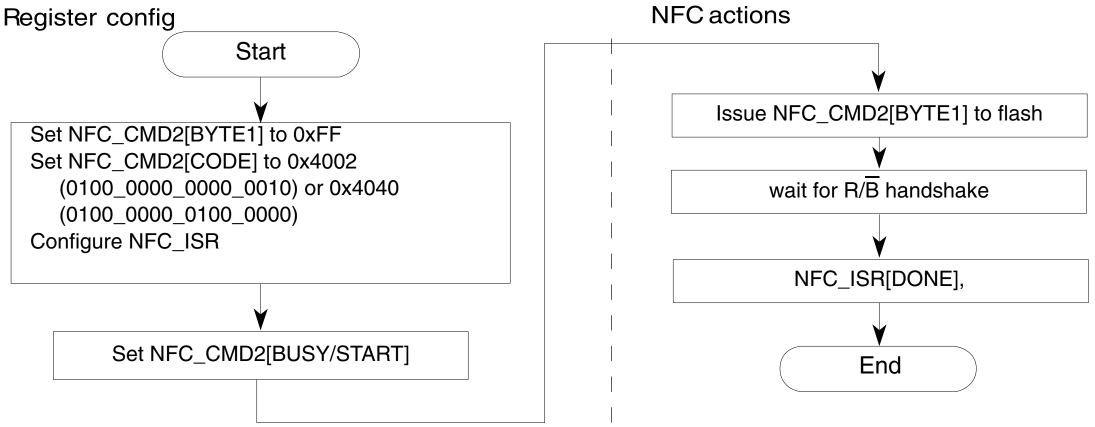

Register config
NFC actions
Start
Set NFC_CMD2[BYTE1] to 0xFF
Set NFC_CMD2[CODE] to 0x4002
(0100_0000_0000_0010) or 0x4040
(0100_0000_0100_0000)
Configure NFC_ISR
Set NFC_CMD2[BUSY/START]
Issue NFC_CMD2[BYTE1] to flash
wait for R/$\overline{B}$ handshake
NFC_ISR[DONE],
End

Figure 32-24. Flow Chart of Reset Operation

32.4.4 Fast Flash Configuration for EDO

Normally, read out data goes valid after the high-to-low transition of $\overline{RE}$, and invalid on the low-to-high transition (as shown in the next figure) t_{RHOH}<t_{REH}. NFC sampled the read data at the negedge of flash_clk, and because the data is invalid at that time, a latch is used here to maintain the valid data during the high period of flash_clk, so that NFC can sample correct data.

Some flash devices contain an EDO (enhanced data out) feature, where the data can be held until the next high-to-low $\overline{RE}$ transition (see the figure after next, labeled "Read Operation, EDO type"), t_{RHOH}>t_{REH}. The read data is valid at the negedge of flash_clk, NFC can sample data directly without latching it. To support the EDO feature, the NFC must work in fast mode (NFC_CFG[FAST] set). The NFC clock must be configured fast enough (usually > 33 MHz) according to the data sheet of flash devices.

* Serial access Cycle after Read(CLE=L, $\overline{WE}$=H, ALE=L)

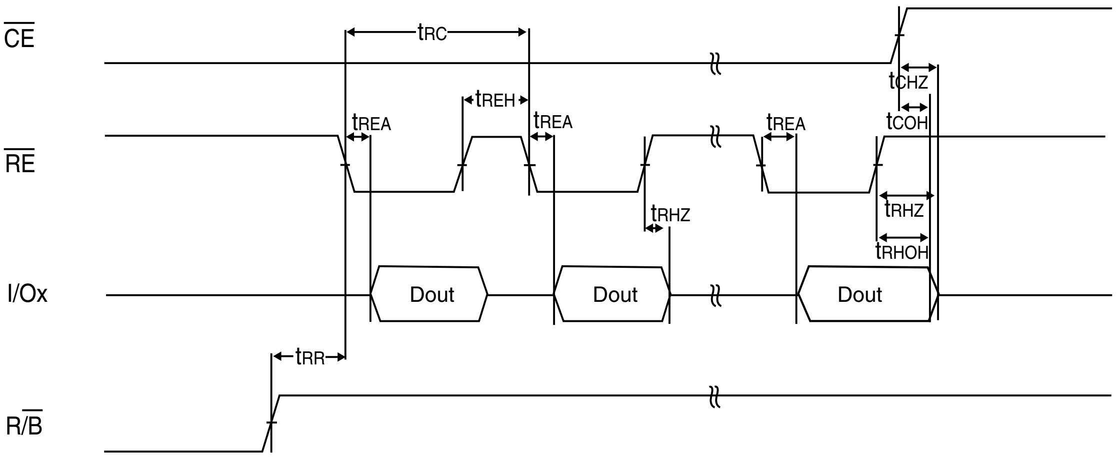


Figure 32-25. Read Operation

Serial Access Cycle after Read(EDO Type, CLE=L, $\overline{WE}$=H, ALE=L)

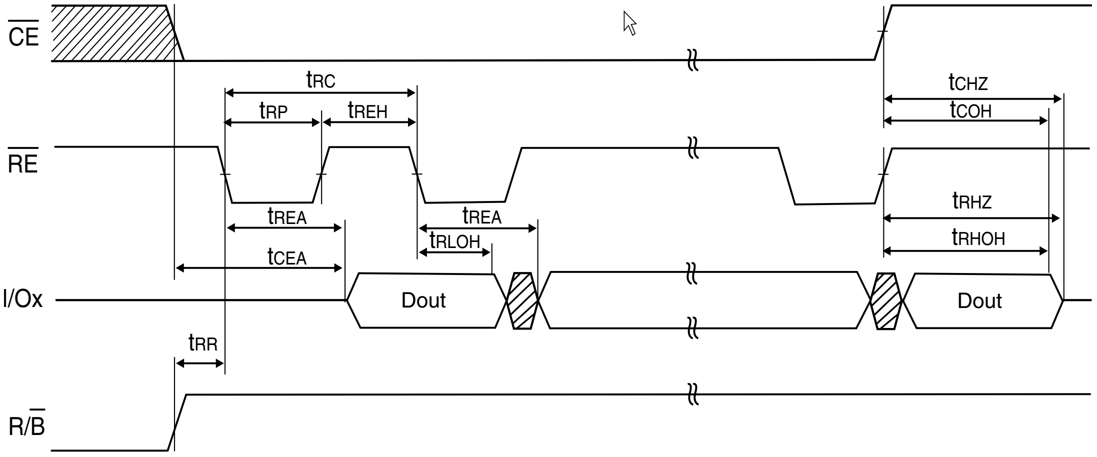


Figure 32-26. Read Operation, EDO type

32.4.5 Organization of the Data in the NAND Flash

Pages on the flash can be split into multiple virtual ECC/DMA pages. The parameter that controls this is NFC_CFG[PAGECNT]. This parameter gives the number of virtual ECC/ DMA pages in one flash page.

The virtual page is split into a user (main) area and ECC (spare) area. Data in the user area can be set or used by the application, while data in the ECC area is set and used by the ECC.

The following tables give virtual-to-physical mappings for various flash devices and their recommended settings.

Table 32-19. Virtual-to-Physical Mappings of Different Flash,

Flash page size (main +spare) bytes	NFC_CFG [ECC MODE]	ECC bits	NFC_CFG [PAGE CNT]	Sector size (bytes)	Virtual page user size (bytes)	Mapping
512 + 16	000	0	1	528	528	VirtualPage_0[527:0] = Physical[527:0]
512 + 16	001	4	1	528	520	VirtualPage_0[519:0] = Physical[519:0]
2048 + 64	000	0	1	2112	2112	VirtualPage_0[2111:0] = Physical[2111:0]
2048 + 64	101	16	1	2112	2082	VirtualPage_0[2081:0] = Physical[2081:0]
2048 + 64	110	24	1	2112	2067	VirtualPage_0[2066:0] = Physical[2066:0]
2048 + 64	111	32	1	2112	2052	VirtualPage_0[2051:0] = Physical[2051:0]
2048 + 64	000	0	4	528	528	VirtualPage_0[527:0] = Physical[527:0] VirtualPage_1[527:0] = Physical[1055:528] VirtualPage_2[527:0] = Physical[1583:1056] VirtualPage_3[527:0] = Physical[2111:1584]
2048 + 64	001	4	4	528	520	VirtualPage_0[519:0] = Physical[519:0] VirtualPage_1[519:0] = Physical[1047:528] VirtualPage_2[519:0] = Physical[1575:1056] VirtualPage_3[519:0] = Physical[2103:1584]
4096 + 128	000	0	2	2112	2112	VirtualPage_0[2111:0] = Physical[2111:0] VirtualPage_1[2111:0] = Physical[4223:2112]
4096 + 128	101	16	2	2112	2082	VirtualPage_0[2081:0] = Physical[2081:0] VirtualPage_1[2081:0] = Physical[4193:2112]
4096 + 128	110	24	2	2112	2067	VirtualPage_0[2066:0] = Physical[2066:0] VirtualPage_1[2066:0] = Physical[4178:2112]
4096 + 128	111	32	2	2112	2052	VirtualPage_0[2051:0] = Physical[2051:0][1] VirtualPage_1[2051:0] = Physical[4163:2112]

Table continues on the next page...

Table 32-19. Virtual-to-Physical Mappings of Different Flash, (continued)

Flash page size (main +spare) bytes	NFC_CFG [ECC MODE]	ECC bits	NFC_CFG [PAGE CNT]	Sector size (bytes)	Virtual page user size (bytes)	Mapping
4096 + 128	000	0	8[2]	528	528	VirtualPage_0[527:0] = Physical[527:0] VirtualPage_1[527:0] = Physical[1055:528] VirtualPage_2[527:0] = Physical[1583:1056] VirtualPage_3[527:0] = Physical[2111:1584] VirtualPage_4[527:0] = Physical[2639:2112] VirtualPage_5[527:0] = Physical[3167:2640] VirtualPage_6[527:0] = Physical[3695:3168] VirtualPage_7[527:0] = Physical[4223:3696]
4096 + 128	001	4	8	528	520	VirtualPage_0[519:0] = Physical[519:0] VirtualPage_1[519:0] = Physical[1047:528] VirtualPage_2[519:0] = Physical[1575:1056] VirtualPage_3[519:0] = Physical[2103:1584] VirtualPage_4[519:0] = Physical[2631:2112] VirtualPage_5[519:0] = Physical[3159:2640] VirtualPage_6[519:0] = Physical[3687:3168] VirtualPage_7[519:0] = Physical[4215:3696]
4096 + 208	000	0	2	2152	2152	VirtualPage_0[2151:0] = Physical[2151:0] VirtualPage_1[2151:0] = Physical4303:2152]
4096 + 208	101	16	2	2152	2122	VirtualPage_0[2121:0] = Physical[2121:0] VirtualPage_1[2121:0] = Physical[4273:2152]
4096 + 208	110	24	2	2152	2104	VirtualPage_0[2103:0] = Physical[2103:0] VirtualPage_1[2103:0] = Physical[4255:2152]
4096 + 208	111	32	2	2152	2092	VirtualPage_0[2091:0] = Physical[2091:0] VirtualPage_1[2091:0] = Physical[4243:2152]
4096 + 208	000	0	8	538	538	VirtualPage_0[537:0] = Physical[537:0] VirtualPage_1[537:0] = Physical[1075:538] VirtualPage_2[537:0] = Physical[1613:1076] VirtualPage_3[537:0] = Physical[2151:1614] VirtualPage_4[537:0] = Physical[2689:2152] VirtualPage_5[537:0] = Physical[3227:2690] VirtualPage_6[537:0] = Physical[3765:3228] VirtualPage_7[537:0] = Physical[4304:3766]

Table continues on the next page...

Table 32-19. Virtual-to-Physical Mappings of Different Flash, (continued)

Flash page size (main +spare) bytes	NFC_CFG [ECC MODE]	ECC bits	NFC_CFG [PAGE CNT]	Sector size (bytes)	Virtual page user size (bytes)	Mapping
4096 + 208	001	4	8	538	530	VirtualPage_0[529:0] = Physical[529:0] VirtualPage_1[529:0] = Physical[1067:538] VirtualPage_2[529:0] = Physical[1605:1076] VirtualPage_3[529:0] = Physical[2143:1614] VirtualPage_4[529:0] = Physical[2681:2152] VirtualPage_5[529:0] = Physical[3219:2690] VirtualPage_6[529:0] = Physical[3757:3228] VirtualPage_7[529:0] = Physical[4295:3766]
4096 + 208	010	6	8	538	526	VirtualPage_0[525:0] = Physical[525:0] VirtualPage_1[525:0] = Physical[1063:538] VirtualPage_2[525:0] = Physical[1601:1076] VirtualPage_3[525:0] = Physical[2139:1614] VirtualPage_4[525:0] = Physical[2677:2152] VirtualPage_5[525:0] = Physical[3215:2690] VirtualPage_6[525:0] = Physical[3753:3228] VirtualPage_7[525:0] = Physical[4291:3766]
4096 + 208	011	8	8	538	523	VirtualPage_0[522:0] = Physical[523:0] VirtualPage_1[522:0] = Physical[1060:538] VirtualPage_2[522:0] = Physical[1598:1076] VirtualPage_3[522:0] = Physical[2136:1614] VirtualPage_4[522:0] = Physical[2674:2152] VirtualPage_5[522:0] = Physical[3212:2690] VirtualPage_6[522:0] = Physical[3750:3228] VirtualPage_7[522:0] = Physical[4288:3766]

1. In most applications, this mode is of no use because user size is too small.
2. When 4KB page is split into eight virtual pages, if page program/read using DMA, set NFC_CFG[PAGECNT] to 8. if not using DMA, set NFC_CFG[PAGECNT] to 4. See Page Read and Page Program for details.

If flash devices with a physical page size of 4K or more are used, the bad block marker appears as the first byte of the spare area. But, because of the physical-to-virtual mapping, it does not appear in byte 2048 of the virtual page, where its logical place would be. The DMA engine contains the option to swap some bytes, and to make the bad block marker appear in the requested place.

Table 32-20. Using the Swap Field to Move the Bad Block Marker

Flash sector size (main + spare) bytes	Bad block marker (physical)	Bad block marker (virtual) Before swap	Bad block marker (expected) After swap	Swap

4096 + 128	4096	Page 1/byte 1984	Page 1/byte 2048	NFC_SWAP[ADDR1] = (1984/8)[1]
				NFC_SWAP[ADDR2] = (2048/8)

4096 + 208	4096	Page 1/byte 1944	Page 1/byte 2048	NFC_SWAP[ADDR1] = (1944/8)
				NFC_SWAP[ADDR2] = (2048/8)

[1] Only works with a user page size of at least 2055 bytes. Does not work with ECC mode 111.

32.4.6 Flash Command Code Description

The 16-bit command code in NFC_CMD2[CODE] is defined in the next table. If a bit is set, the action is executed. The command is repeated for the number of the NFC_RPT[COUNT] value. If NFC_RPT[COUNT] is zero or one, the command is executed once.

Table 32-21. NFC_CMD2[CODE] Detail

NFC_CMD2 [CODE] bit	Action when Bit is Set
15	Start DMA transfer to read data from memory , and write to SRAM.
14	Send command byte 1 (NFC_CMD2[BYTE1]) to flash
13	Send column address 1 (NFC_CAR[BYTE1]) to flash
12	Send column address 2 (NFC_CAR[BYTE2]) to flash
11	Send row address 1 (NFC_RAR[BYTE1]) to flash
10	Send row address 2 (NFC_RAR[BYTE2]) to flash
9	Send row address 3 (NFC_RAR[BYTE3]) to flash
8	Write data to flash. Total of NFC_CFG[PAGECNT] pages is written to the flash, and equal number of starts is sent to the residue engine. Also, additional starts to the DMA engine are sent, until DMA has transferred the NFC_CFG[PAGECNT] data from memory to NFC.
7	Send command byte 2 (NFC_CMD1[BYTE2]) to flash
6	Wait for flash R/$\overline{B}$ handshake
5	Read data from flash. Read is only started if the new NFC_CMD2[BUFNO] is idle. One or more starts are sent to the residue engine, total NFC_CFG[PAGECNT] starts. **Note:** For reads, DMA is not started. Instead, to start DMA for reads, NFC_CFG[DMAREQ] must be set.
4	Send command byte 3 (NFC_CMD1[BYTE3]) to flash
3	Read flash status
2	Read ID

Table continues on the next page...

Table 32-21. NFC_CMD2[CODE] Detail (continued)

NFC_CMD2 [CODE] bit	Action when Bit is Set
1	Always set. End-of-command marker used to signal done.
0	Reserved, must be cleared.

32.4.7 Interrupts

There are two interrupts to flag the end of a command execution:

1. The done interrupt, NFC_ISR[DONE]. Use this interrupt if commands are sent back-to-back to the flash. It indicates when a new command can be dispatched. The done interrupt is given before the flash data is corrected and resident in memory, because operation of the ECC engine and DMA engine is pipelined.

   When the done interrupt is tracks command completion, the software may also monitor the NFC_ISR[ECCBUSY, DMABUSY, ECCBN, DMABN] fields.

   a. NFC_ISR[ECCBUSY] indicates that the ECC block is still busy, and reports the buffer number the ECC block is working on in NFC_ISR[ECCBN].

   b. NFC_ISR[DMABUSY] indicates that the DMA block is still busy, and reports the buffer number the DMA block is working on in NFC_ISR[DMABN].

2. The command idle interrupt, NFC_ISR[IDLE]. Use this interrupt if you want to use the data produced in the next process. The idle interrupt indicates all command processing has terminated, and the relevant data is now available in memory or the NFC SRAM buffer. When using back-to-back reads to the flash, use of the idle interrupt means the NFC does not operate at its maximum transfer speed, as ECC and DMA are now done in foreground.

When using the done interrupt, transfer completion for write pages can be assumed when the done interrupt is received. When done is received for read pages, the data may still be in flight in the DMA or the ECC. To check this, the CPU should remember the buffer number (NFC_CMD2[BUFNO]) associated with the command, and wait until the DMA and ECC are either idle, or are both busy on a different buffer number. (The ECC buffer number and DMA buffer number fields do not match the BUFNO specified with command.) You can check on any done interrupt or by polling the register.

Chapter 33
External Bus Interface (FlexBus)

33.1 Introduction

NOTE

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

This chapter describes external bus data transfer operations and error conditions. It describes transfers initiated by the core processor (or any other bus master) and includes detailed timing diagrams showing the interaction of signals in supported bus operations.

33.1.1 Overview

A multi-function external bus interface called the FlexBus interface controller is provided on the device with basic functionality of interfacing to slave-only devices. It can be directly connected to the following asynchronous or synchronous devices with little or no additional circuitry:

- External ROMs
- Flash memories
- Programmable logic devices
- Other simple target (slave) devices

For asynchronous devices, a simple chip-select based interface can be used.

The FlexBus interface has up to six general purpose chip-selects, $\overline{\text{FB\_CS}}$[5:0]. The actual number of chip selects available depends upon the device and its pin configuration.

33.1.2 Features

Key FlexBus features include:

- Six independent, user-programmable chip-select signals ($\overline{\text{FB\_CS}}$[5:0]) that can interface with external SRAM, PROM, EPROM, EEPROM, flash, and other peripherals
- 8-, 16-, and 32-bit port sizes with configuration for multiplexed or non-multiplexed address and data buses
- 8-bit, 16-bit, 32-bit, and 16-byte transfers
- Programmable burst- and burst-inhibited transfers selectable for each chip select and transfer direction
- Programmable address-setup time with respect to the assertion of chip select
- Programmable address-hold time with respect to the negation of chip select and transfer direction
- Extended address latch enable option helps with glueless connections to synchronous and asynchronous memory devices

33.1.3 Modes of Operation

The external interface is a configurable multiplexed bus set to one of the following modes:

- 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)
- Non-multiplexed 32-bit address and 32-bit data busses

33.2 Signal Descriptions

This section 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.

Table 33-1. FlexBus Signal Summary

Signal	Description	I/O
FB_A[31:0]	In a non-multiplexed configuration, this is the address bus.	O
FB_D[31:0]/ FB_AD[31:0]	In a non-multiplexed configuration, this is the data bus. In a multiplexed configuration this bus is the address/data bus, FB_AD[31:0]. In non-multiplexed and multiplexed configurations, during the first cycle, this bus drives the upper address byte, addr[31:24].	I/O
FB_CS[5:0]	General purpose chip-selects. The actual number of chip selects available depends upon the device and its pin configuration.	O
FB_BE_31_24 FB_BE_23_16 FB_BE_15_8 FB_BE_7_0	Byte enables	O
FB_OE	Output enable	O
FB_R/W	Read/write. 1 = Read, 0 = Write	O
FB_TS	Transfer start	O
FB_ALE	Address latch enable (an inverse of FB_TS)	O
FB_TSIZ[1:0]	Transfer size	O
FB_TBST	Burst transfer indicator	O
FB_TA	Transfer acknowledge	I
FB_CLK	FlexBus clock output	O

33.2.1 Address and Data Buses (FB_A*n*, FB_D*n*, FB_AD*n*)

In non-multiplexed mode, the FB_A[31:0] and FB_D[31:0] buses carry the address and data, respectively. The number of byte lanes carrying the data is determined by the port size associated with the matching chip select.

In multiplexed mode, the FB_AD[31:0] bus carries the address and data. The full 32-bit address is driven on the first clock of a bus cycle (address phase). Following the first clock, the data is driven on the bus (data phase). During the data phase, the address continues driving on the pins not used for data. For example, in 16-bit mode the lower address continues driving on FB_AD[15:0] and in 8-bit mode the lower address continues driving on FB_AD[23:0].

33.2.2 Chip Selects (FB_CS[5 :0])

The chip-select signal indicates which device is selected. A particular chip-select asserts when the transfer address is within the device's address space, as defined in the base- and mask-address registers. The actual number of chip selects available depends upon the pin configuration.

33.2.3 Byte Enables (FB_BE_31_24, FB_BE_23_16, FB_BE_15_8, FB_BE_7_0)

When driven low, the byte enable outputs indicate data is to be latched or driven onto a specific byte lane of the data bus. A configuration option is provided to assert these signals on reads and writes or writes only.

For external SRAM or flash devices, the FB_BE*n* outputs must be connected to individual byte strobe signals.

33.2.4 Output Enable (FB_OE)

The output enable signal (FB_OE) is sent to the interfacing memory and/or peripheral to enable a read transfer. FB_OE is only asserted during read accesses when a chip select matches the current address decode.

33.2.5 Read/Write (FB_R/W)

The processor drives the FB_R/W signal to indicate the current bus operation direction. It is driven high during read bus cycles and low during write bus cycles.

33.2.6 Transfer Start/Address Latch Enable (FB_TS/FB_ALE)

The assertion of FB_TS indicates that the device has begun a bus transaction and the address and attributes are valid.

In multiplexed mode, an inverted FB_TS (FB_ALE) is available as an address latch enable, which indicates when the address is being driven on the FB_AD bus.

FB_TS/FB_ALE is asserted for one bus clock cycle.

This device can extend this signal until the first positive clock edge after FB_CS*n* asserts. See CSCR*n*[EXTS] and Extended Transfer Start/Address Latch Enable.

33.2.7 Transfer Size (FB_TSIZ[1:0])

For memory accesses, these signals, along with $\overline{\text{FB\_TBST}}$, indicate 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.

For misaligned transfers, FB_TSIZ[1:0] indicates the size of each transfer. For example, if a 32-bit access through a 32-bit port device occurs at a misaligned offset of 0x1, 8 bits is transferred first (FB_TSIZ[1:0] = 01), 16 bits is transferred next at offset 0x2 (FB_TSIZ[1:0] = 10), and the final 8 bits is transferred at offset 0x4 (FB_TSIZ[1:0] = 01).

For aligned transfers larger than the port size, FB_TSIZ[1:0] behaves as follows:

- If bursting is used, FB_TSIZ[1:0] is driven to the transfer size.
- If bursting is inhibited, FB_TSIZ[1:0] first shows the entire transfer size and then shows the port size.

Table 33-2. Data Transfer Size

FB_TSIZ[1:0]	Transfer Size
00	4 bytes
01	1 byte
10	2 bytes
11	16 bytes (line)

For burst-inhibited transfers, FB_TSIZ[1:0] changes with each $\overline{\text{FB\_TS}}$ assertion to reflect the next transfer size. For transfers to port sizes smaller than the transfer size, FB_TSIZ[1:0] indicates 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_TSIZ[1:0] equals 00 for the first transaction and 01 for the next three transactions. If bursting is used for a 32-bit write to an 8-bit port, FB_TSIZ[1:0] is driven to 00 for the entire transfer.

33.2.8 Transfer Burst ($\overline{\text{FB\_TBST}}$)

Transfer burst indicates that a burst transfer is in progress as driven by the device. A burst transfer can be two to 16 beats depending on FB_TSIZ[1:0] and the port size.

Note

> When burst (FB_TBST = 0), transfer size is 16 bytes (FB_TSIZ[1:0] = 11) and the address is misaligned within the 16-byte boundary, the external device must be able to wrap around the address.

33.2.9 Transfer Acknowledge (FB_TA)

This input signal indicates the external data transfer is complete. When the processor recognizes FB_TA during a read cycle, it latches the data and then terminates the bus cycle. When the processor recognizes FB_TA during a write cycle, the bus cycle is terminated.

If auto-acknowledge is disabled (CSCR*n*[AA] = 0), the external device drives FB_TA to terminate the bus transfer; if auto-acknowledge is enabled (CSCR*n*[AA] = 1), FB_TA is generated internally after a specified number of wait states, or the external device may assert external FB_TA before the wait-state countdown, terminating the cycle early. The device negates FB_CS*n* one cycle after the last FB_TA asserts. During read cycles, the peripheral must continue to drive data until FB_TA is recognized. For write cycles, the processor continues driving data one clock after FB_CS*n* is negated.

The number of wait states is determined by CSCR*n* or the external FB_TA input. If the external FB_TA is used, the peripheral has total control on the number of wait states.

Note

> External devices should only assert FB_TA while the FB_CS*n* signal to the external device is asserted.
>
> The CSPMCR register controls muxing of FB_TA with other signals. If auto-acknowledge is not used and CSPMCR does not allow FB_TA control, the FlexBus may hang.

33.3 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.

FB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_C000	Chip select address register (FB_CSAR0)	32	R/W	0000_0000h	33.3.1/ 856
4000_C004	Chip select mask register (FB_CSMR0)	32	R/W	0000_0000h	33.3.2/ 857
4000_C008	Chip select control register (FB_CSCR0)	32	R/W	0000_0000h	33.3.3/ 858
4000_C00C	Chip select address register (FB_CSAR1)	32	R/W	0000_0000h	33.3.1/ 856
4000_C010	Chip select mask register (FB_CSMR1)	32	R/W	0000_0000h	33.3.2/ 857
4000_C014	Chip select control register (FB_CSCR1)	32	R/W	0000_0000h	33.3.3/ 858
4000_C018	Chip select address register (FB_CSAR2)	32	R/W	0000_0000h	33.3.1/ 856
4000_C01C	Chip select mask register (FB_CSMR2)	32	R/W	0000_0000h	33.3.2/ 857
4000_C020	Chip select control register (FB_CSCR2)	32	R/W	0000_0000h	33.3.3/ 858
4000_C024	Chip select address register (FB_CSAR3)	32	R/W	0000_0000h	33.3.1/ 856
4000_C028	Chip select mask register (FB_CSMR3)	32	R/W	0000_0000h	33.3.2/ 857
4000_C02C	Chip select control register (FB_CSCR3)	32	R/W	0000_0000h	33.3.3/ 858
4000_C030	Chip select address register (FB_CSAR4)	32	R/W	0000_0000h	33.3.1/ 856
4000_C034	Chip select mask register (FB_CSMR4)	32	R/W	0000_0000h	33.3.2/ 857
4000_C038	Chip select control register (FB_CSCR4)	32	R/W	0000_0000h	33.3.3/ 858
4000_C03C	Chip select address register (FB_CSAR5)	32	R/W	0000_0000h	33.3.1/ 856

Table continues on the next page...

FB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4000_C040	Chip select mask register (FB_CSMR5)	32	R/W	0000_0000h	33.3.2/ 857
4000_C044	Chip select control register (FB_CSCR5)	32	R/W	0000_0000h	33.3.3/ 858
4000_C060	Chip select port multiplexing control register (FB_CSPMCR)	32	R/W	0000_0000h	33.3.4/ 861

33.3.1 Chip select address register (FB_CSAR*n*)

The CSARn registers specify the chip-select base addresses.

NOTE

> Because the FlexBus module is one of the slaves connected to the crossbar switch, it is only accessible within a certain memory range. Refer to the device memory map for the applicable FlexBus "expansion" address range for which the chip-selects can be active. Set the CSARn registers appropriately.

Addresses: FB_CSAR0 is 4000_C000h base + 0h offset = 4000_C000h
FB_CSAR1 is 4000_C000h base + Ch offset = 4000_C00Ch
FB_CSAR2 is 4000_C000h base + 18h offset = 4000_C018h
FB_CSAR3 is 4000_C000h base + 24h offset = 4000_C024h
FB_CSAR4 is 4000_C000h base + 30h offset = 4000_C030h
FB_CSAR5 is 4000_C000h base + 3Ch offset = 4000_C03Ch

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	BA																0															
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

FB_CSAR*n* field descriptions

Field	Description
31–16 BA	Base address Defines the base address for memory dedicated to chip-select $\overline{\text{FB\_CSn}}$. BA is compared to bits 31–16 on the internal address bus to determine if chip-select memory is being accessed.
15–0 Reserved	This read-only field is reserved and always has the value zero.

33.3.2 Chip select mask register (FB_CSMR*n*)

CSMRn registers specify the address mask and allowable access types for the respective chip-selects.

Addresses: FB_CSMR0 is 4000_C000h base + 4h offset = 4000_C004h
FB_CSMR1 is 4000_C000h base + 10h offset = 4000_C010h
FB_CSMR2 is 4000_C000h base + 1Ch offset = 4000_C01Ch
FB_CSMR3 is 4000_C000h base + 28h offset = 4000_C028h
FB_CSMR4 is 4000_C000h base + 34h offset = 4000_C034h
FB_CSMR5 is 4000_C000h base + 40h offset = 4000_C040h

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	BAM																0							WP	0							V
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

FB_CSMR*n* field descriptions

Field	Description
31–16 BAM	Base address mask Defines the chip-select block size by masking address bits. Setting a BAM bit causes the corresponding CSAR bit to be a don’t care in the decode. The block size for $\overline{\text{FB\_CSn}}$ is 2^n; n = (number of bits set in respective CSMR[BAM]) + 16. For example, if CSAR0[BA] equals 0x0040 and CSMR0[BAM] equals 0x0008, $\overline{\text{FB\_CS0}}$ addresses two discontinuous 64 KB memory blocks: one from 0x40_0000 – 0x40_FFFF and one from 0x48_0000 – 0x48_FFFF. Likewise, for $\overline{\text{FB\_CS0}}$ to access 32 MB of address space starting at location 0x00_0000, $\overline{\text{FB\_CS1}}$ must begin at the next byte after $\overline{\text{FB\_CS0}}$ for a 16 MB address space. Therefore, CSAR0[BA] equals 0x0000, CSMR0[BAM] equals 0x01FF, CSAR1[BA] equals 0x0200, and CSMR1[BAM] equals 0x00FF. 0 Corresponding address bit is used in chip-select decode 1 Corresponding address bit is a don’t care in chip-select decode.
15–9 Reserved	This read-only field is reserved and always has the value zero.
8 WP	Write protect Controls write accesses to the address range in the corresponding CSAR. Attempting to write to the range of addresses for which CSARn[WP] is set results in a bus error termination of the internal cycle and no external cycle. 0 Read and write accesses are allowed 1 Only read accesses are allowed
7–1 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

FB_CSMR*n* field descriptions (continued)

Field	Description
0 V	Valid Indicates whether the corresponding CSAR, CSMR, and CSCR contents are valid. Programmed chip-selects do not assert until V bit is set (except for FB_CS0, which acts as the global chip-select). Reset clears each CSMRn[V]. **NOTE:** At reset, no chip-select other than FB_CS0 can be used until the CSMR0[V] is set. Afterward, FB_CS[5:0] functions as programmed. 0 Chip select invalid 1 Chip select valid

33.3.3 Chip select control register (FB_CSCR*n*)

Each CSCRn controls the auto-acknowledge, address setup and hold times, port size, burst capability, and number of wait states. To support the global chip-select (FB_CS0) the CSCR0 reset values differ from the other CSCRs.

NOTE

The reset value of CSCR0 is as follows:

- Bits 31-24 are 0
- Bit 23-3 are device-dependent
- Bits 3-0 are 0

See the Chip Configuration details for your particular device for information on the exact CSCR0 reset value.

Addresses: FB_CSCR0 is 4000_C000h base + 8h offset = 4000_C008h
FB_CSCR1 is 4000_C000h base + 14h offset = 4000_C014h
FB_CSCR2 is 4000_C000h base + 20h offset = 4000_C020h
FB_CSCR3 is 4000_C000h base + 2Ch offset = 4000_C02Ch
FB_CSCR4 is 4000_C000h base + 38h offset = 4000_C038h
FB_CSCR5 is 4000_C000h base + 44h offset = 4000_C044h

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	SWS						0		SWSEN	EXTS	ASET		RDAH		WRAH		WS						BLS	AA	PS		BEM	BSTR	BSTW	0		
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

FB_CSCR*n* field descriptions

Field	Description
31–26 SWS	Secondary wait states

Table continues on the next page...

FB_CSCR*n* field descriptions (continued)

Field	Description
	If the SWSEN bit is set, 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 the wait state count (CSCR*n*[WS]). If the SWSEN bit is cleared, the WS value is used for all burst transfers and this field is ignored.
25–24 Reserved	This read-only field is reserved and always has the value zero.
23 SWSEN	Secondary wait state enable 0 The WS value inserts wait states before an internal transfer acknowledge is generated for all transfers 1 The SWS value inserts wait states before an internal transfer acknowledge is generated for burst transfer secondary terminations
22 EXTS	Extended address latch enable 0 FB_TS/FB_ALE asserts for one bus clock cycle 1 FB_TS/FB_ALE remains asserted until the first positive clock edge after FB_CSn asserts
21–20 ASET	Address setup Controls the assertion of the chip-select with respect to assertion of a valid address and attributes. The address and attributes are considered valid at the same time FB_TS/FB_ALE asserts. 00 Assert FB_CSn on first rising clock edge after address is asserted. (Default FB_CSn) 01 Assert FB_CSn on second rising clock edge after address is asserted. 10 Assert FB_CSn on third rising clock edge after address is asserted. 11 Assert FB_CSn on fourth rising clock edge after address is asserted. (Default FB_CS0)
19–18 RDAH	Read address hold or deselect This field controls the address and attribute hold time after the termination during a read cycle that hits in the chip-select 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. The number of cycles the address and attributes are held after FB_CSn negation depends on the value of CSCRn[AA]. 00 If AA is cleared, 1 cycle. If AA is set, 0 cycles. 01 If AA is cleared, 2 cycles. If AA is set, 1 cycle. 10 If AA is cleared, 3 cycles. If AA is set, 2 cycles. 11 If AA is cleared, 4 cycles. If AA is set, 3 cycles.
17–16 WRAH	Write address hold or deselect Write address hold or deselect. This field controls the address, data, and attribute hold time after the termination of a write cycle that hits in the chip-select 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. 00 Hold address and attributes one cycle after FB_CSn negates on writes. (Default FB_CSn) 01 Hold address and attributes two cycles after FB_CSn negates on writes. 10 Hold address and attributes three cycles after FB_CSn negates on writes. 11 Hold address and attributes four cycles after FB_CSn negates on writes. (Default FB_CS0)

Table continues on the next page...

FB_CSCR*n* field descriptions (continued)

Field	Description
15–10 WS	Wait states The number of wait states inserted after FB_CSn asserts and before an internal transfer acknowledge is generated (WS = 0 inserts zero wait states, WS = 0x3F inserts 63 wait states). If AA is reserved, FB_TA must be asserted by the external system regardless of the number of generated wait states. In that case, the external transfer acknowledge ends the cycle. An external FB_TA supersedes the generation of an internal FB_TA.
9 BLS	Byte-lane shift Determines if data on FB_AD appears left-justified or right-justified during the data phase of a FlexBus access. 0 Not shifted. Data is left-justfied on FB_AD. 1 Shifted. Data is right justified on FB_AD.
8 AA	Auto-acknowledge enable Determines the assertion of the internal transfer acknowledge for accesses specified by the chip-select address. **NOTE:** If AA is set for a corresponding FB_CSn and the external system asserts an external FB_TA before the wait-state countdown asserts the internal FB_TA, the cycle is terminated. Burst cycles increment the address bus between each internal termination. **NOTE:** This bit must be set if CSPMCR disables FB_TA. 0 No internal FB_TA is asserted. Cycle is terminated externally 1 Internal transfer acknowledge is asserted as specified by WS
7–6 PS	Port size Specifies the data port width associated with each chip-select. It determines where data is driven during write cycles and where data is sampled during read cycles. 00 32-bit port size. Valid data sampled and driven on FB_D[31:0] 01 8-bit port size. Valid data sampled and driven on FB_D[31:24] if BLS = 0 or FB_D[7:0] if BLS = 1 10 16-bit port size. Valid data sampled and driven on FB_D[31:16] if BLS = 0 or FB_D[15:0] if BLS = 1 11 16-bit port size. Valid data sampled and driven on FB_D[31:16] if BLS = 0 or FB_D[15:0] if BLS = 1
5 BEM	Byte-enable mode Specifies the byte enable operation. Certain memories have byte enables that must be asserted during reads and writes. BEM can be set in the relevant CSCR to provide the appropriate mode of byte enable support for these SRAMs. 0 The FB_BE*n* signals are not asserted for reads. The FB_BE*n* signals are asserted for data write only. 1 The FB_BE*n* signals are asserted for read and write accesses
4 BSTR	Burst-read enable Specifies whether burst reads are used for memory associated with each FB_CSn. 0 Data exceeding the specified port size is broken into individual, port-sized, non-burst reads. For example, a longword read from an 8-bit port is broken into four 8-bit reads. 1 Enables data burst reads larger than the specified port size, including longword reads from 8- and 16-bit ports, word reads from 8-bit ports, and line reads from 8, 16-, and 32-bit ports.

Table continues on the next page...

FB_CSCR*n* field descriptions (continued)

Field	Description
3 BSTW	Burst-write enable Specifies whether burst writes are used for memory associated with each FB_CSn. 0 Break data larger than the specified port size into individual, port-sized, non-burst writes. For example, a longword write to an 8-bit port takes four byte writes. 1 Enables burst write of data larger than the specified port size, including longword writes to 8 and 16-bit ports, word writes to 8-bit ports, and line writes to 8-, 16-, and 32-bit ports.
2–0 Reserved	This read-only field is reserved and always has the value zero.

33.3.4 Chip select port multiplexing control register (FB_CSPMCR)

The CSPMCR register controls the multiplexing of the FlexBus signals.

NOTE

A bus error occurs when:

- writing a reserved value,
- writing to a reserved bit location in this register, or
- not accessing this register as 32-bit.

Address: FB_CSPMCR is 4000_C000h base + 60h offset = 4000_C060h

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	GROUP1				GROUP2				GROUP3				GROUP4				GROUP5				0											
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

FB_CSPMCR field descriptions

Field	Description
31–28 GROUP1	FlexBus signal group 1 multiplex control Controls the multiplexing of the FB_ALE, FB_CS1, and FB_TS signals. 0000 FB_ALE 0001 FB_CS1 0010 FB_TS Else Reserved
27–24 GROUP2	FlexBus signal group 2 multiplex control Controls the multiplexing of the FB_CS4, FB_TSIZ0, and FB_BE_31_24 signals. 0000 FB_CS4 0001 FB_TSIZ0

Table continues on the next page...

FB_CSPMCR field descriptions (continued)

Field	Description
	0010 FB_BE_31_24 Else Reserved
23–20 GROUP3	FlexBus signal group 3 multiplex control Controls the multiplexing of the FB_CS5, FB_TSIZ1, and FB_BE_23_16 signals. 0000 FB_CS5 0001 FB_TSIZ1 0010 FB_BE_23_16 Else Reserved
19–16 GROUP4	FlexBus signal group 4 multiplex control Controls the multiplexing of the FB_TBST, FB_CS2, and FB_BE_15_8 signals. 0000 FB_TBST 0001 FB_CS2 0010 FB_BE_15_8 Else Reserved
15–12 GROUP5	FlexBus signal group 5 multiplex control Controls the multiplexing of the FB_TA, FB_CS3, and FB_BE_7_0 signals. **NOTE:** When GROUP5 is not 0000, you must set the CSCR*n*[AA] bit. Else, the bus hangs during a transfer. 0000 FB_TA 0001 FB_CS3. You must also set CSCR*n*[AA]. 0010 FB_BE_7_0. You must also set CSCR*n*[AA]. Else Reserved
11–0 Reserved	This read-only field is reserved and always has the value zero.

33.4 Functional Description

This section provides the functional description of the module.

33.4.1 Chip-Select Operation

Each chip-select has a dedicated set of registers for configuration and control:

- Chip-select address registers (CSAR*n*) control the base address space of the chip-select.

- Chip-select mask registers (CSMR*n*) provide 16-bit address masking and access control.
- Chip-select control registers (CSCR*n*) provide port size and burst capability indication, wait-state generation, address setup and hold times, and automatic acknowledge generation features.

33.4.1.1 General Chip-Select Operation

When a bus cycle is routed to the FlexBus, the device first compares its address with the base address and mask configurations programmed for chip-selects 0 to 5 (configured in CSCR*n*). The results depend on if the address matches or not as shown in the following table.

Table 33-26. Results of Address Comparison

Address Matches CSAR*n*?	Result
Yes, one CSAR	The appropriate chip-select is asserted, generating a FlexBus bus cycle as defined in the chip-select control register. If CSMR[WP] is set and a write access is performed, the internal bus cycle terminates with a bus error, no chip select is asserted, and no external bus cycle is performed.
No	The access is terminated with a bus error response, no chip select is asserted and no FlexBus cycle is performed.
Yes, multiple CSARs	The access is terminated with a bus error response, no chip select is asserted and no FlexBus cycle is performed.

33.4.1.2 8-, 16-, and 32-Bit Port Sizing

Static bus sizing is programmable through the port size bits, CSCR[PS]. The processor always drives a 32-bit address on the FB_AD bus regardless of the external device's address size. The external device must connect its address/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

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 addr[15:0] to FB_AD[16:1] and data[15:0] to FB_AD[31:16]. See Data Byte Alignment and Physical Connections for a graphical connection.

33.4.2 Data Transfer Operation

Data transfers between the chip and other devices involve these signals:

- Address/data bus (FB_AD[31:0])
- Control signals (FB_TS/FB_ALE, FB_TA, FB_CS*n*, FB_OE, FB_BE*n*)
- Attribute signals (FB_R/W, FB_TBST, FB_TSIZ[1:0])

The address, write data, FB_TS/FB_ALE, FB_CS*n*, and all attribute signals change on the rising edge of the FlexBus clock (FB_CLK). Read data is latched into the device on the rising edge of the clock.

The FlexBus supports 8-bit, 16-bit, 32-bit, and 16-byte (line) operand transfers and allows accesses to 8-, 16-, and 32-bit data ports. Transfer parameters (address setup and hold, port size, the number of wait states for the external device being accessed, automatic internal transfer termination enable or disable, and burst enable or disable) are programmed in the chip-select control registers (CSCRs).

33.4.3 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 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_AD[31:24] (FB_BE_31_24). A 32-bit transfer through this 8-bit port takes four transfers, starting with the LSB to the MSB. A 32-bit transfer through a 32-bit port requires one transfer on each four-byte lane of the FlexBus.

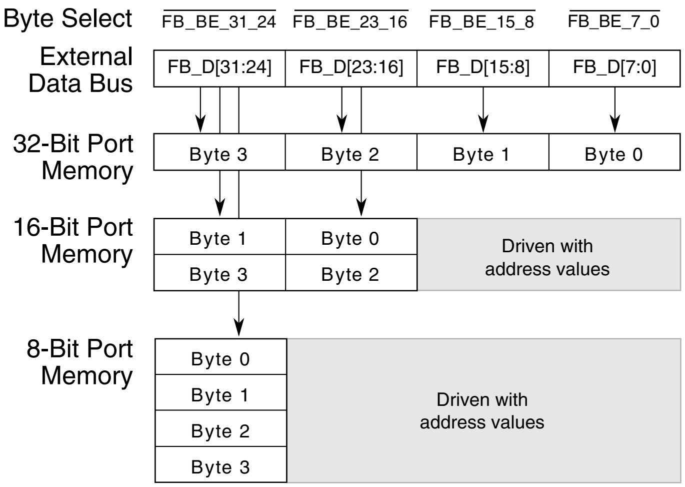


Figure 33-23. Connections for External Memory Port Sizes (CSCR*n*[BLS] = 0)

The following figure shows the byte lanes that external memory connects to and the sequential transfers of a 32-bit transfer for the supported port sizes when byte lane shift is enabled.

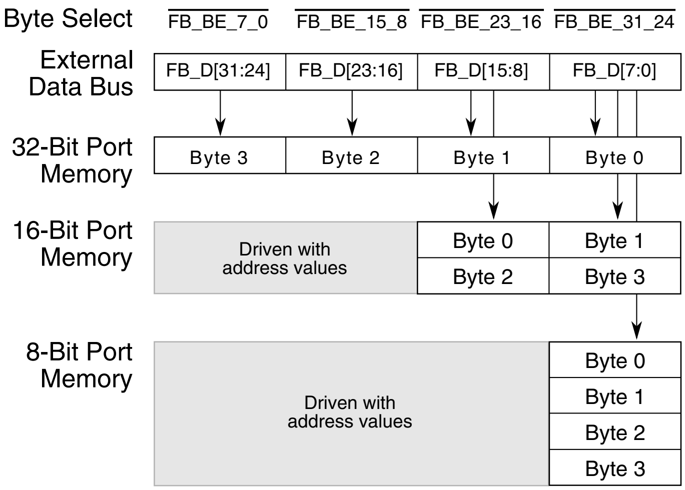


Figure 33-24. Connections for External Memory Port Sizes (CSCR*n*[BLS] = 1)

33.4.4 Address/Data Bus Multiplexing

The interface supports a single 32-bit wide multiplexed address and data bus (FB_AD[31:0]). The full 32-bit address is always driven on the first clock of a bus cycle. During the data phase, the FB_AD[31:0] lines used for data are determined by the programmed port size for the corresponding chip select. The device continues to drive the address on any FB_AD[31:0] lines not used for data.

The tables below lists the supported combinations of address and data bus widths for each CSCR*n*[BLS] setting.

Table 33-27. FlexBus Multiplexed Operating Modes for CSCR*n*[BLS]=0

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	
8-bit	Address phase	Address			
	Data phase	Data	Address		

Table 33-28. FlexBus Multiplexed Operating Modes for CSCR*n*[BLS]=1

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

33.4.5 Bus Cycle Execution

As shown in Figure 33-27 and Figure 33-29, basic bus operations occur in four clocks:

1. S0: At the first clock edge, the address, attributes, and FB_TS/FB_ALE are driven.
2. S1: FB_CS*n* is asserted at the second rising clock edge to indicate the device selected; by that time, the address and attributes are valid and stable. FB_TS/FB_ALE is negated at this edge.

   For a write transfer, data is driven on the bus at this clock edge and continues to be driven until one clock cycle after FB_CS*n* negates. For a read transfer, data is also driven into the device during this cycle.

   External slave asserts FB_TA at this clock edge.

3. S2: Read data and FB_TA are sampled on the third clock edge. FB_TA can be negated after this edge and read data can then be tri-stated.

4. S3: $\overline{\text{FB\_CS}n}$ is negated at the fourth rising clock edge. This last clock of the bus cycle uses what would be an idle clock between cycles to provide hold time for address, attributes, and write data.

33.4.5.1 Data Transfer Cycle States

An on-chip state machine controls the data-transfer operation in the device. The following figure shows the state-transition diagram for basic read and write cycles.

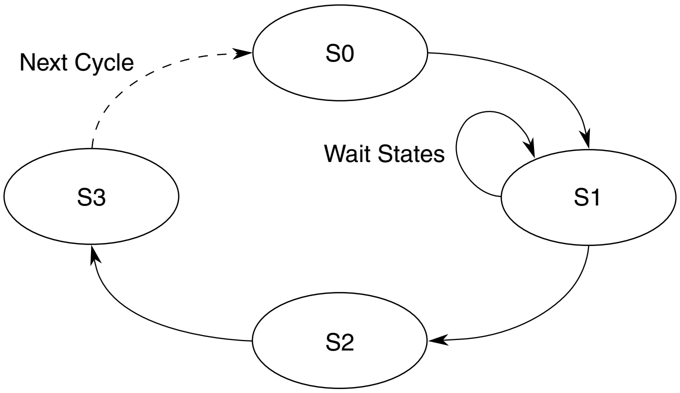


Figure 33-25. Data-Transfer-State-Transition Diagram

The following table describes the states as they appear in subsequent timing diagrams.

Table 33-29. Bus Cycle States

State	Cycle	Description
S0	All	The read or write cycle is initiated. On the rising clock edge, the device places a valid address on FB_AD*n*, asserts $\overline{\text{FB\_TS}}$/FB_ALE, and drives FB_R/$\overline{\text{W}}$ high for a read and low for a write.
S1	All	$\overline{\text{FB\_TS}}$/FB_ALE is negated on the rising edge of FB_CLK, and FB_CSn is asserted. Data is driven on FB_AD[31:*X*] for writes, and FB_AD[31:*X*] is tristated for reads. Address continues to be driven on the FB_AD pins that are unused for data. If $\overline{\text{FB\_TA}}$ is recognized asserted, then the cycle moves on to S2. If $\overline{\text{FB\_TA}}$ is not asserted internally or externally, then the S1 state continues to repeat.
	Read	Data is driven by the external device before the next rising edge of FB_CLK (the rising edge that begins S2) with $\overline{\text{FB\_TA}}$ asserted.
S2	All	For internal termination, $\overline{\text{FB\_CS}n}$ is negated and the internal system bus transfer is completed. For external termination, the external device should negate $\overline{\text{FB\_TA}}$, and the $\overline{\text{FB\_CS}n}$ chip select negates after the rising edge of FB_CLK at the end of S2.
	Read	The processor latches data on the rising clock edge entering S2. The external device can stop driving data after this edge. However, data can be driven until the end of S3 or any additional address hold cycles.
S3	All	Address, data, and FB_R/$\overline{\text{W}}$ go invalid off the rising edge of FB_CLK at the beginning of S3, terminating the read or write cycle.

33.4.6 FlexBus Timing Examples

Note

The timing diagrams throughout this section use signal names that may not be included on your particular device. Ignore these extraneous 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.

33.4.6.1 Basic Read Bus Cycle

During a read cycle, the MCU receives data from memory or a peripheral device. The following figure shows a read cycle flowchart.

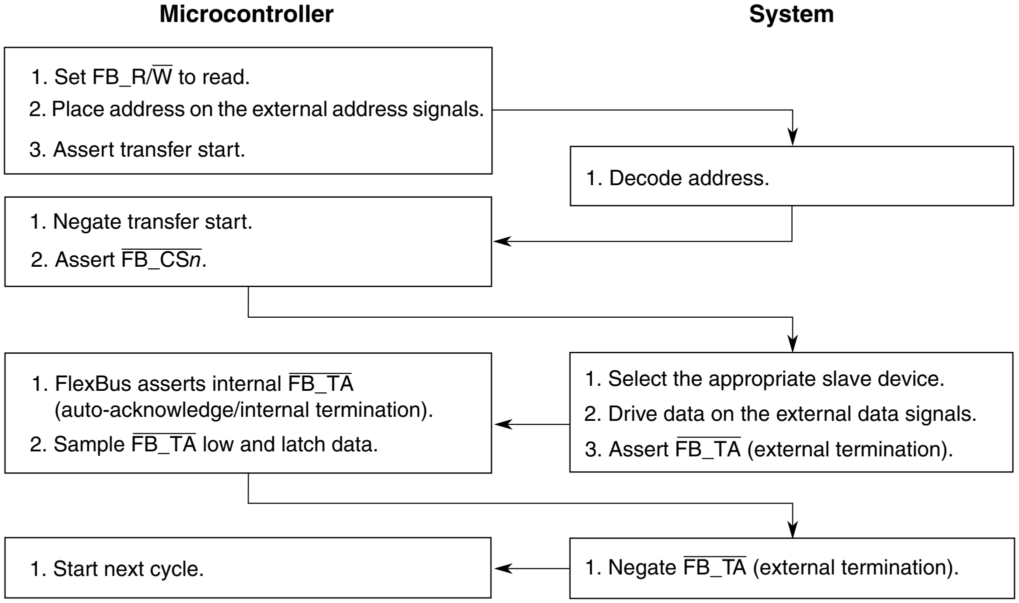

Microcontroller
System
1. Set FB_R/W to read.
2. Place address on the external address signals.
3. Assert transfer start.
1. Decode address.
1. Negate transfer start.
2. Assert FB_CSn.
1. FlexBus asserts internal FB_TA (auto-acknowledge/internal termination).
2. Sample FB_TA low and latch data.
1. Select the appropriate slave device.
2. Drive data on the external data signals.
3. Assert FB_TA (external termination).
1. Start next cycle.
1. Negate FB_TA (external termination).

Figure 33-26. Read Cycle Flowchart

The read cycle timing diagram is shown in the following figure.

Note

FB_TA 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.

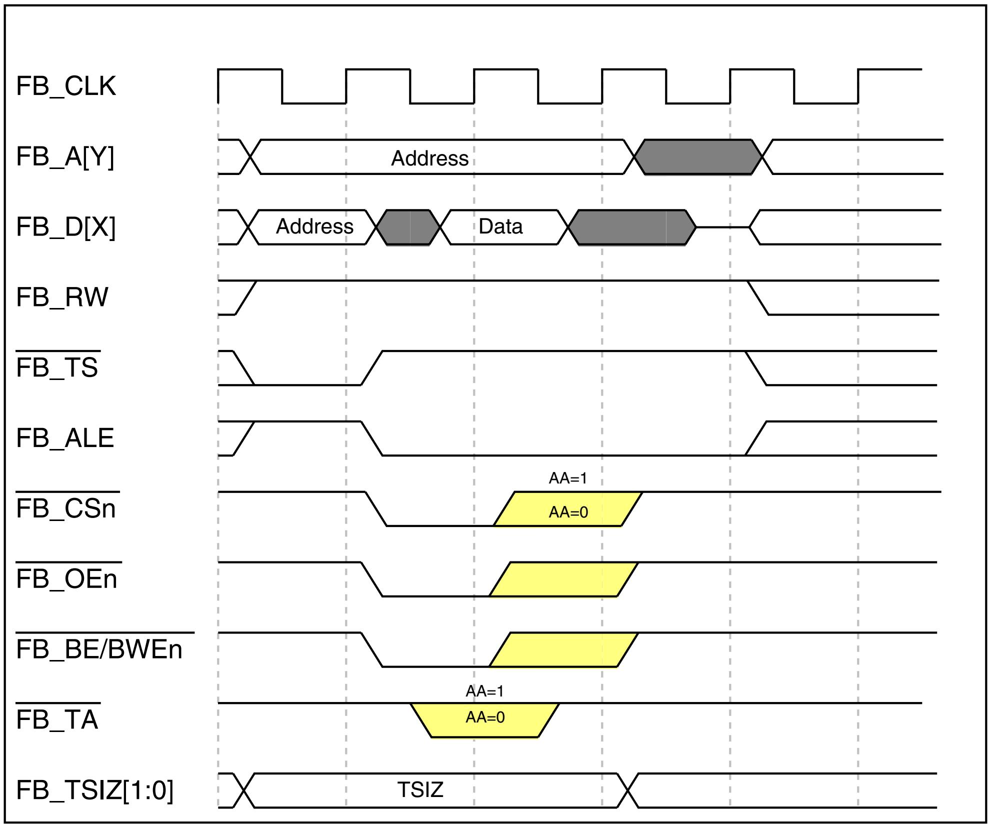


Figure 33-27. Basic Read-Bus Cycle

33.4.6.2 Basic Write Bus Cycle

During a write cycle, the device sends data to memory or to a peripheral device. The following figure shows the write cycle flowchart.

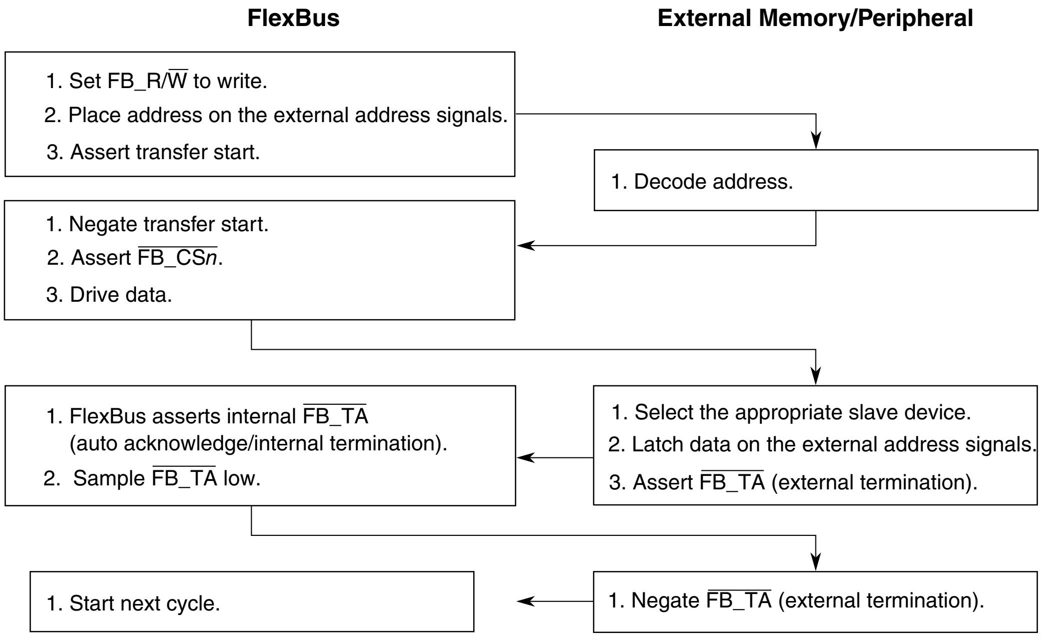

FlexBus
External Memory/Peripheral
1. Set FB_R/W to write.
2. Place address on the external address signals.
3. Assert transfer start.
1. Decode address.
1. Negate transfer start.
2. Assert FB_CSn.
3. Drive data.
1. Select the appropriate slave device.
2. Latch data on the external address signals.
3. Assert FB_TA (external termination).
1. FlexBus asserts internal FB_TA (auto acknowledge/internal termination).
2. Sample FB_TA low.
1. Negate FB_TA (external termination).
1. Start next cycle.

Figure 33-28. Write-Cycle Flowchart

The following figure shows the write cycle timing diagram.

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.

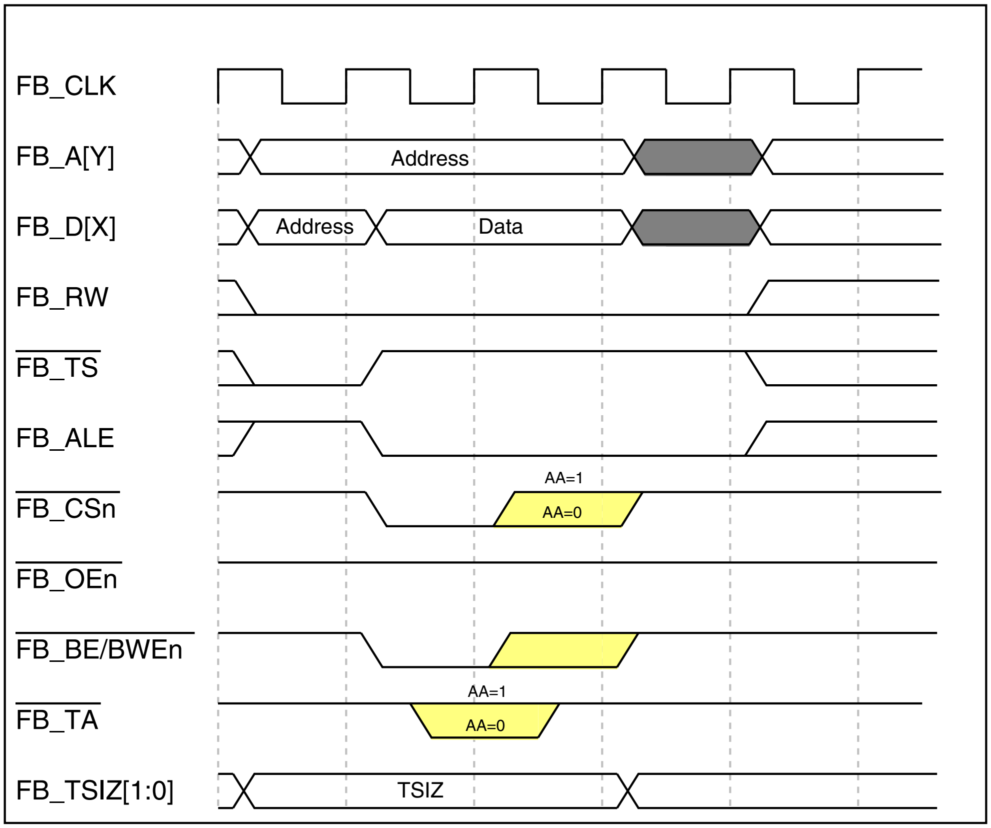


Figure 33-29. Basic Write-Bus Cycle

33.4.6.3 Bus Cycle Sizing

This section shows timing diagrams for various port size scenarios.

33.4.6.3.1 Bus Cycle Sizing—Byte Transfer, 8-bit Device, No Wait States

The following figure illustrates the basic byte read transfer to an 8-bit device with no wait states:

- The address is driven on the full FB_AD[31:8] bus in the first clock.
- The device tristates FB_AD[31:24] on the second 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 one clock after FB_TA is sampled asserted.

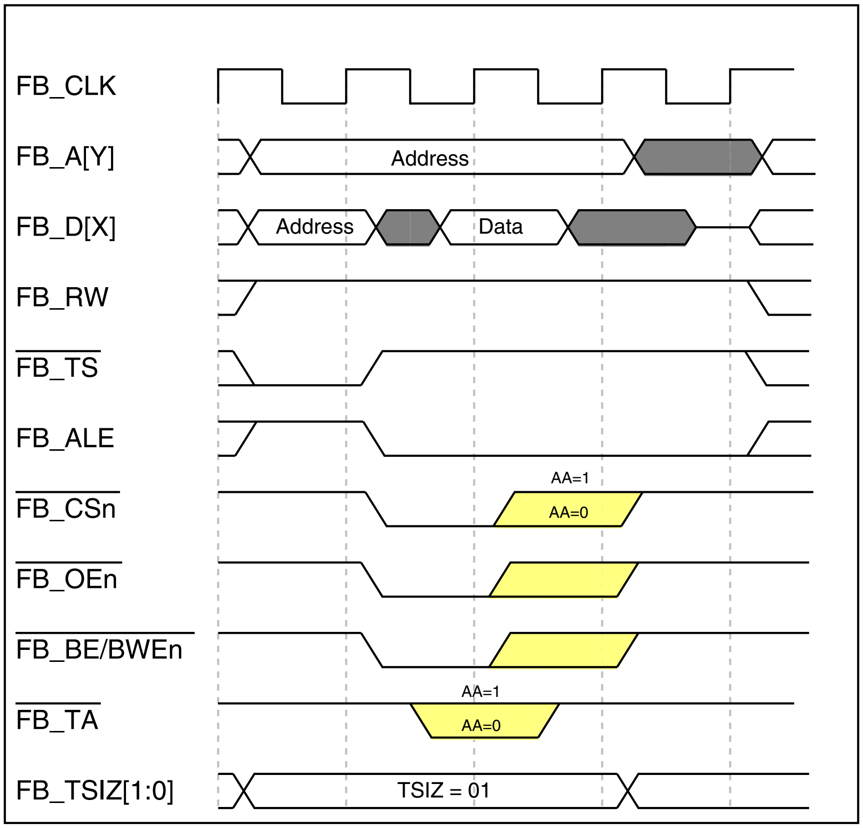


Figure 33-30. Single Byte-Read Transfer

The following figure shows the similar configuration for a write transfer. The data is driven from the second clock on FB_AD[31:24].

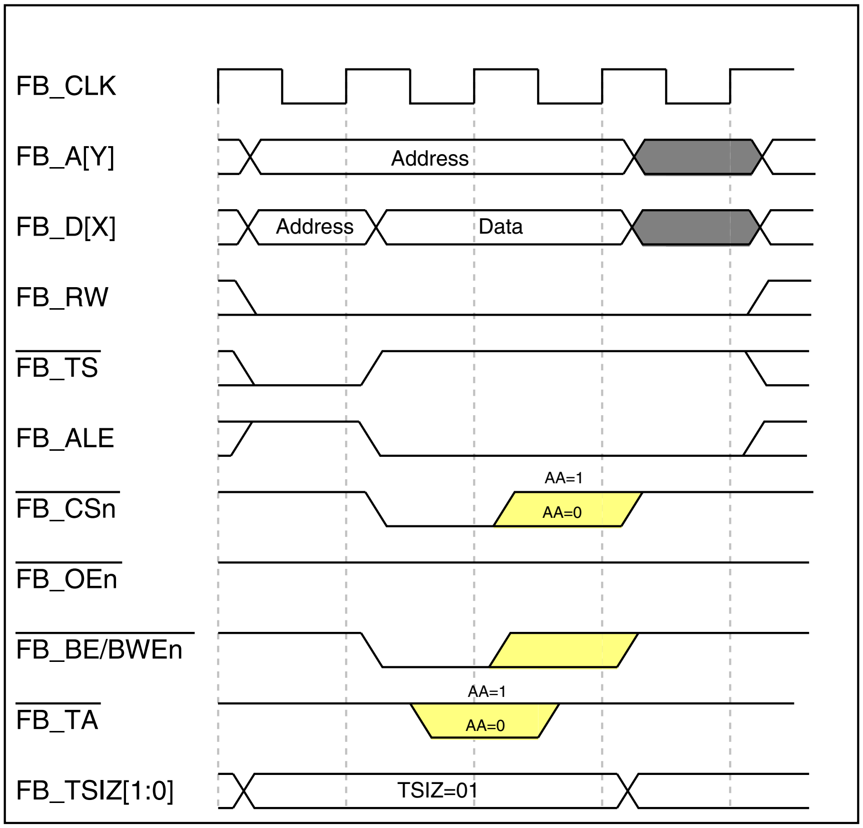


Figure 33-31. Single Byte-Write Transfer

33.4.6.3.2 Bus Cycle Sizing—Word Transfer, 16-bit Device, No Wait States

The following figure illustrates the basic word read transfer to a 16-bit device with no wait states.

- The address is driven on the full FB_AD[31:8] 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 is sampled asserted.

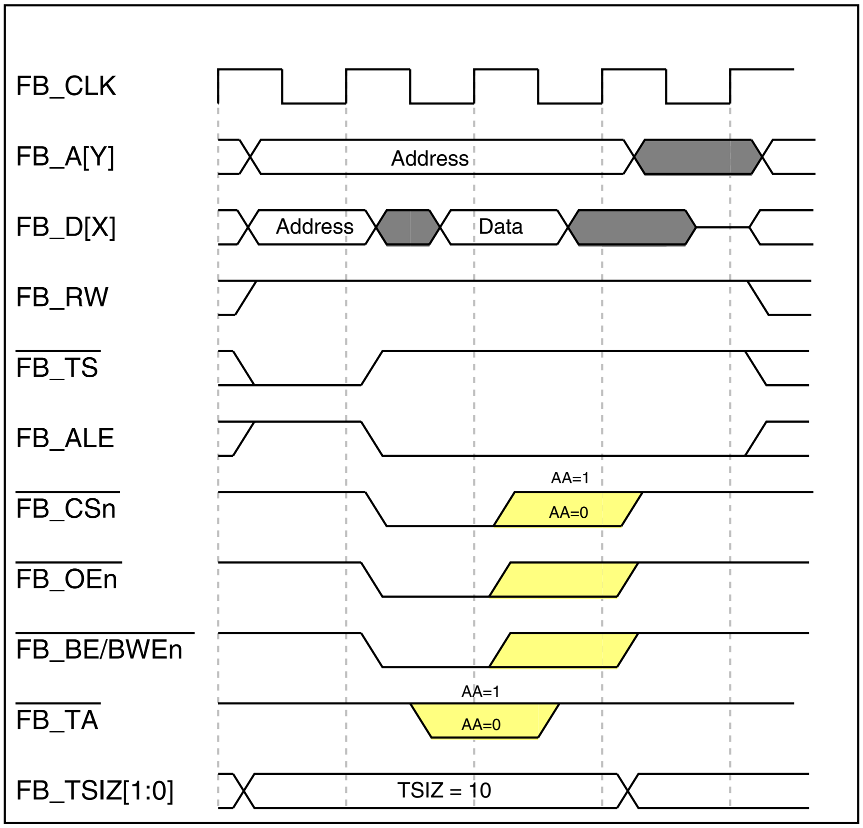


Figure 33-32. Single Word-Read Transfer

The following figure shows the similar configuration for a write transfer. The data is driven from the second clock on FB_AD[31:16].

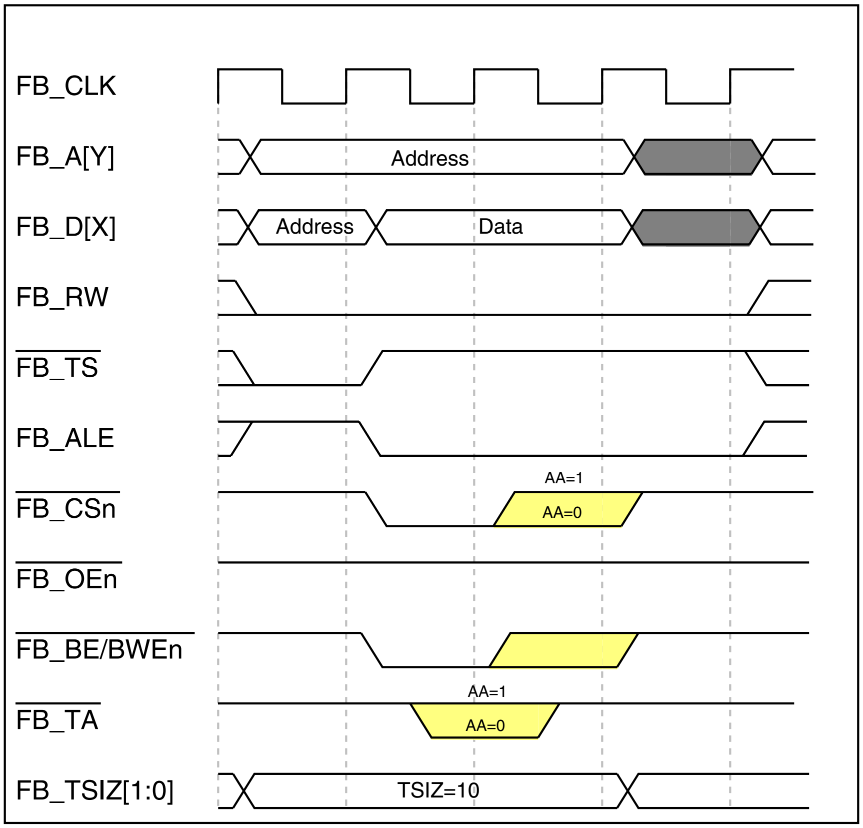


Figure 33-33. Single Word-Write Transfer

33.4.6.3.3 Bus Cycle Sizing—Longword Transfer, 32-bit Device, No Wait States

The following figure depicts a longword read from a 32-bit device.

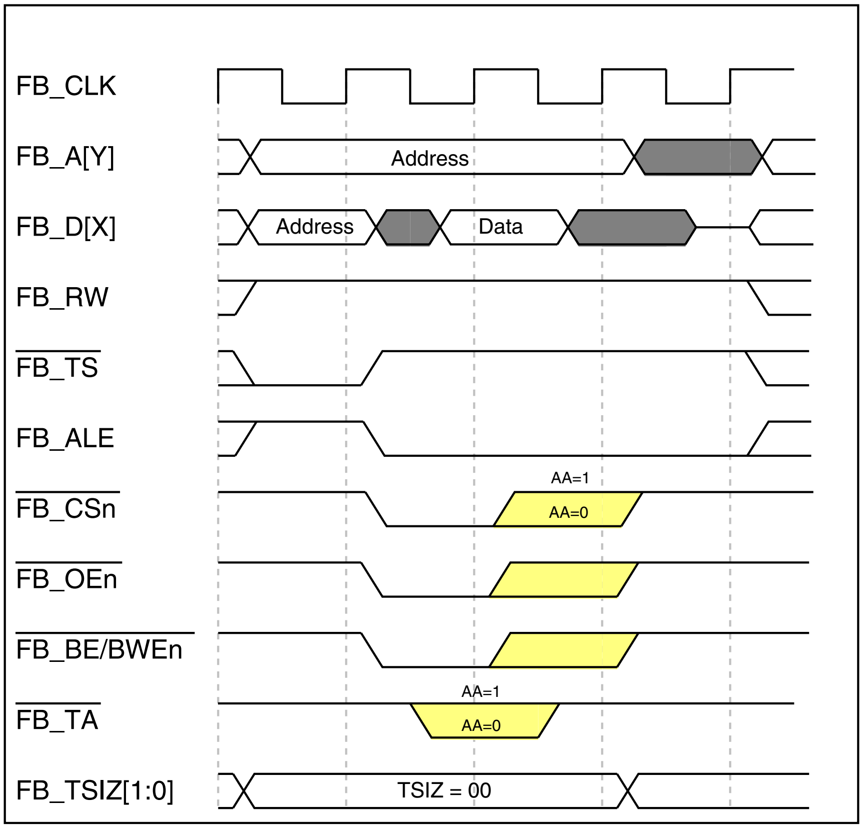


Figure 33-34. Longword-Read Transfer

The following figure illustrates the longword write to a 32-bit device.

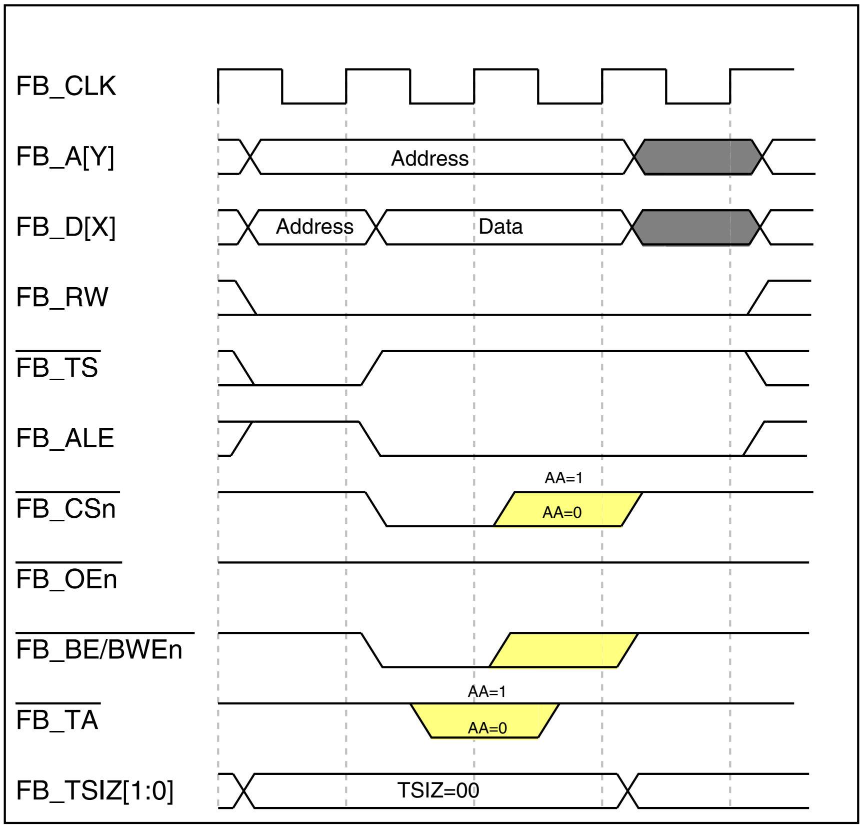


Figure 33-35. Longword-Write Transfer

33.4.6.4 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.

33.4.6.4.1 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 Figure 33-27 and Figure 33-32) with the default of no wait states respectively.

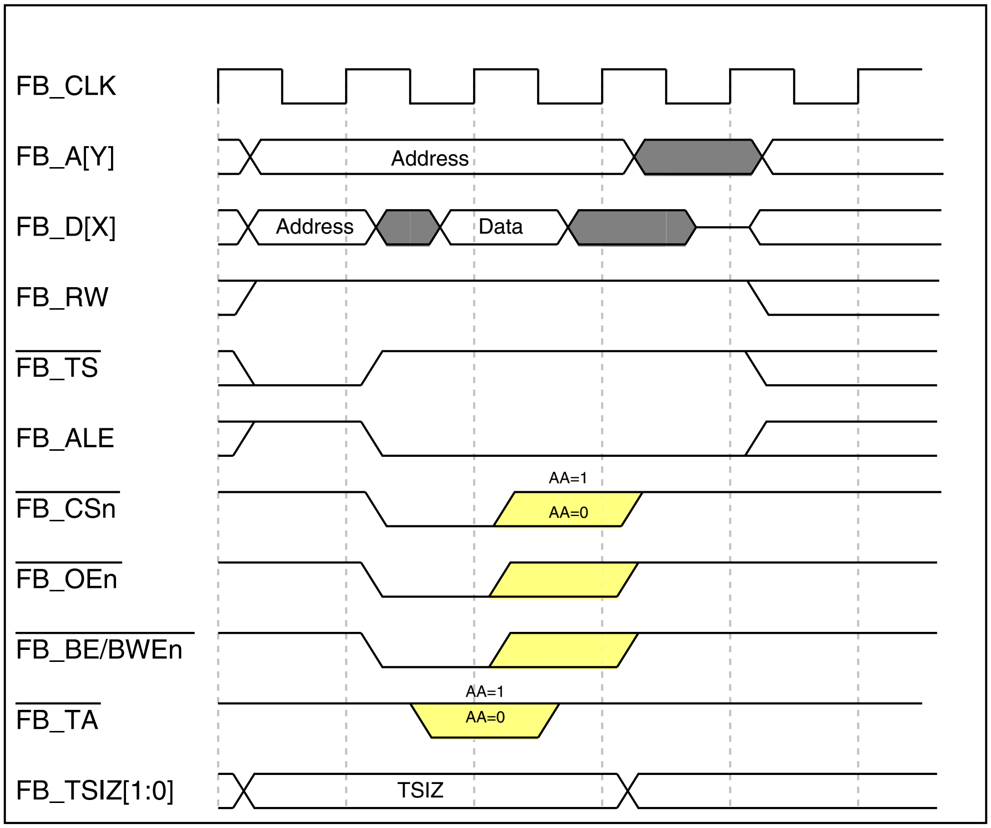


Figure 33-36. Basic Read-Bus Cycle (No Wait States)

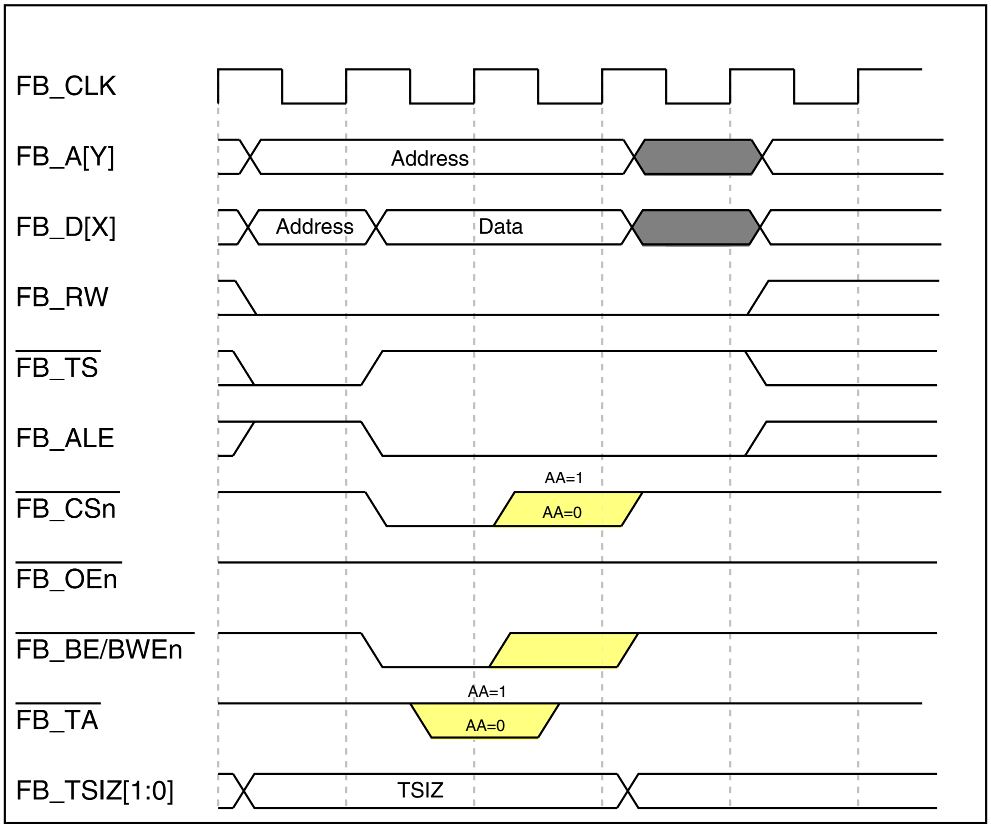


Figure 33-37. Basic Write-Bus Cycle (No Wait States)

If wait states are used, the S1 state repeats continuously until the the chip-select auto-acknowledge unit asserts internal transfer acknowledge or the external FB_TA is recognized as asserted. The following figures show a read and write cycle with one wait state respectively.

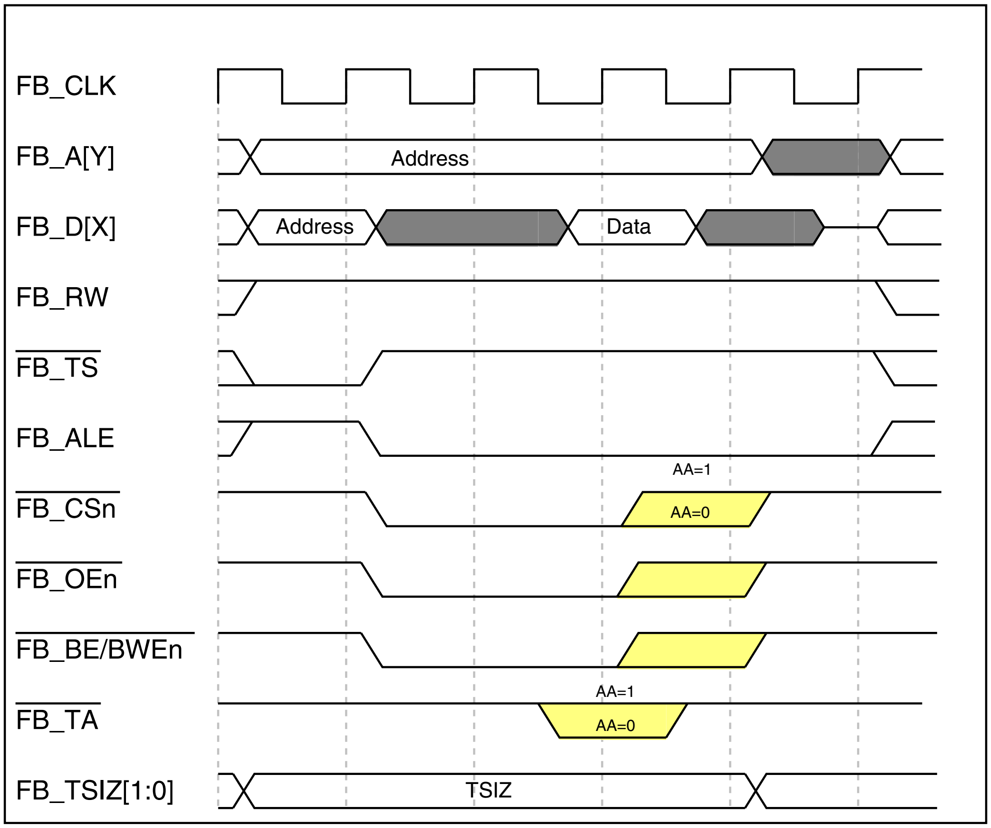


Figure 33-38. Read-Bus Cycle (One Wait State)

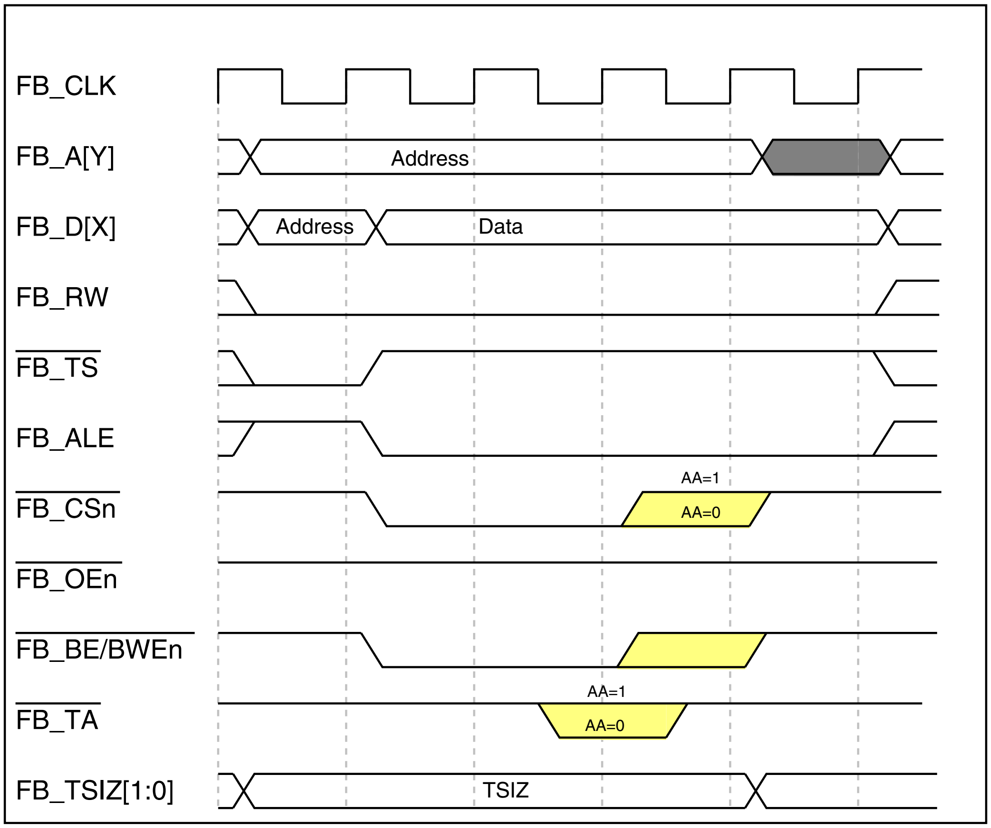


Figure 33-39. Write-Bus Cycle (One Wait State)

33.4.6.4.2 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/FB_ALE) is asserted. The following figures show read- and write-bus cycles with two clocks of address setup respectively.

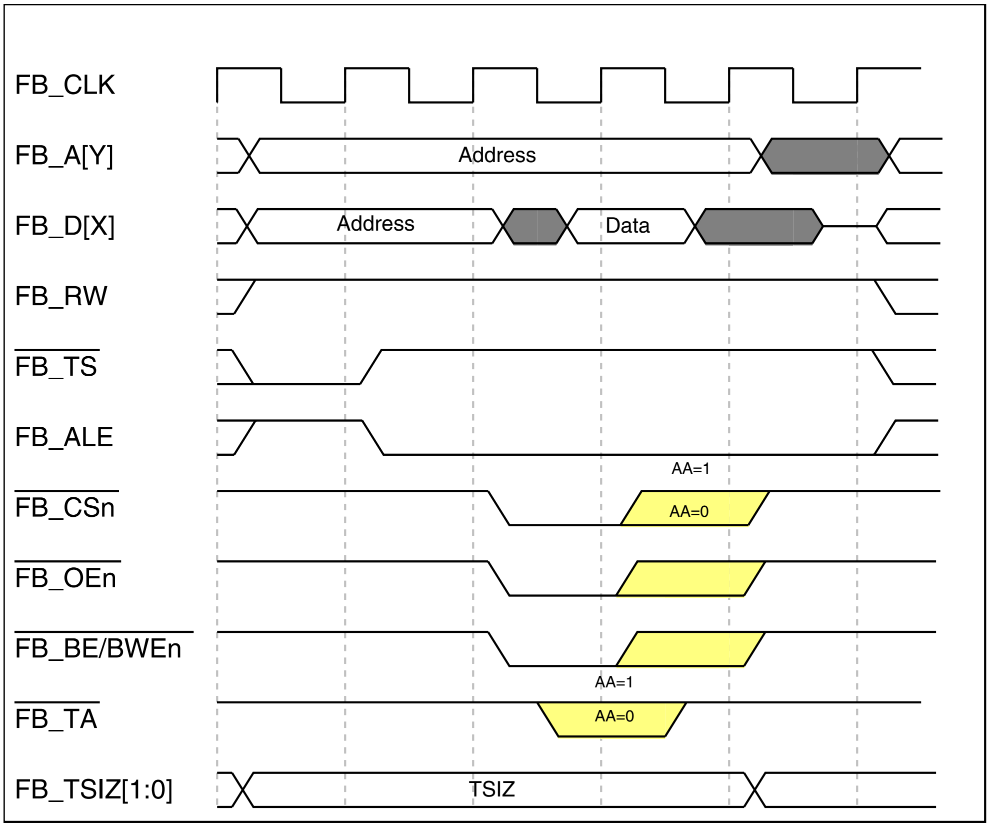


Figure 33-40. Read-Bus Cycle with Two-Clock Address Setup (No Wait States)

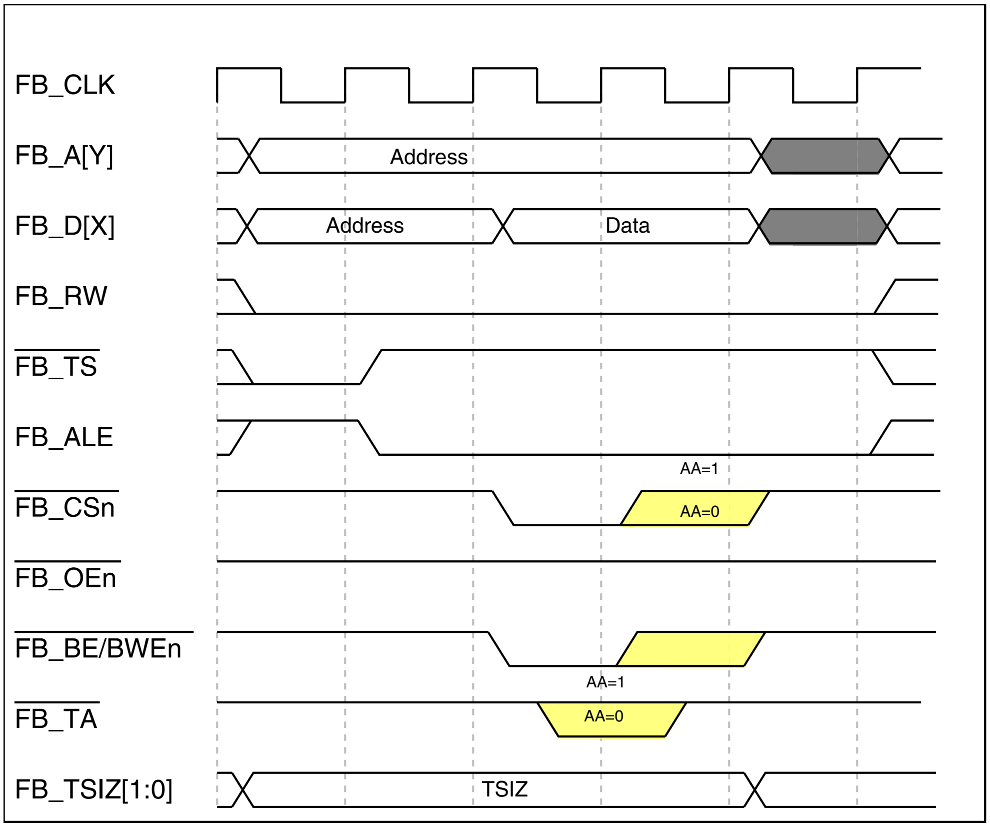


Figure 33-41. Write-Bus Cycle with Two Clock Address Setup (No Wait States)

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.

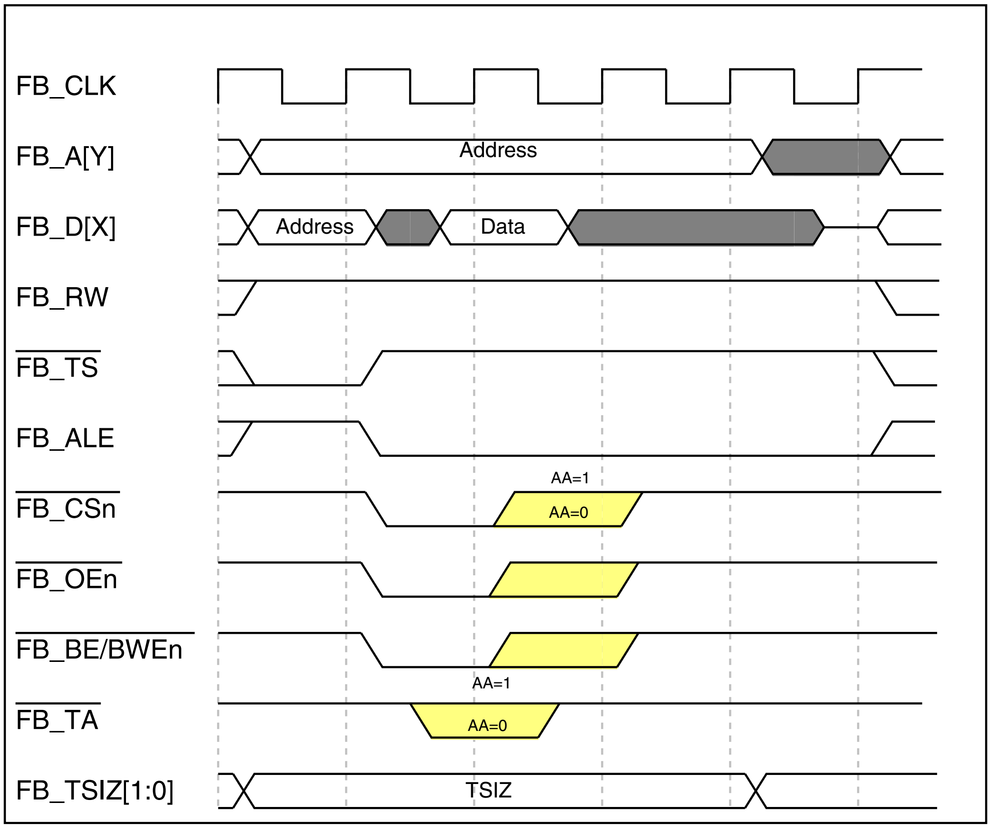


Figure 33-42. Read Cycle with Two-Clock Address Hold (No Wait States)

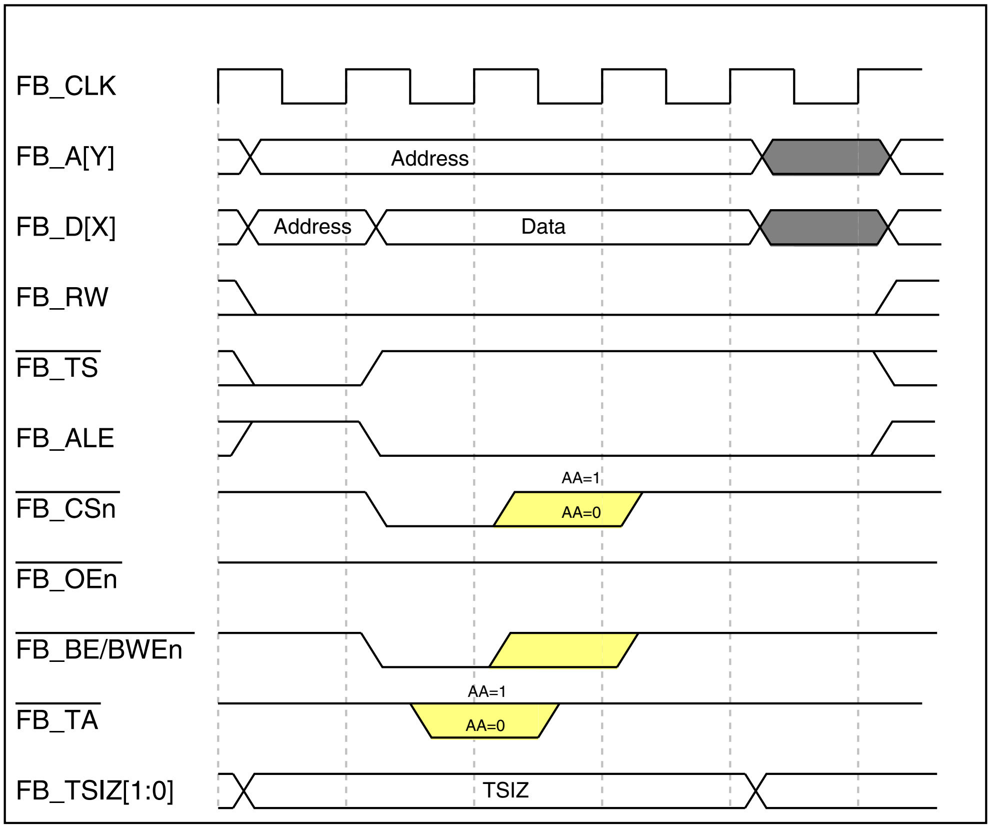


Figure 33-43. Write Cycle with Two-Clock Address Hold (No Wait States)

The following figure shows a bus cycle using address setup, wait states, and address hold.

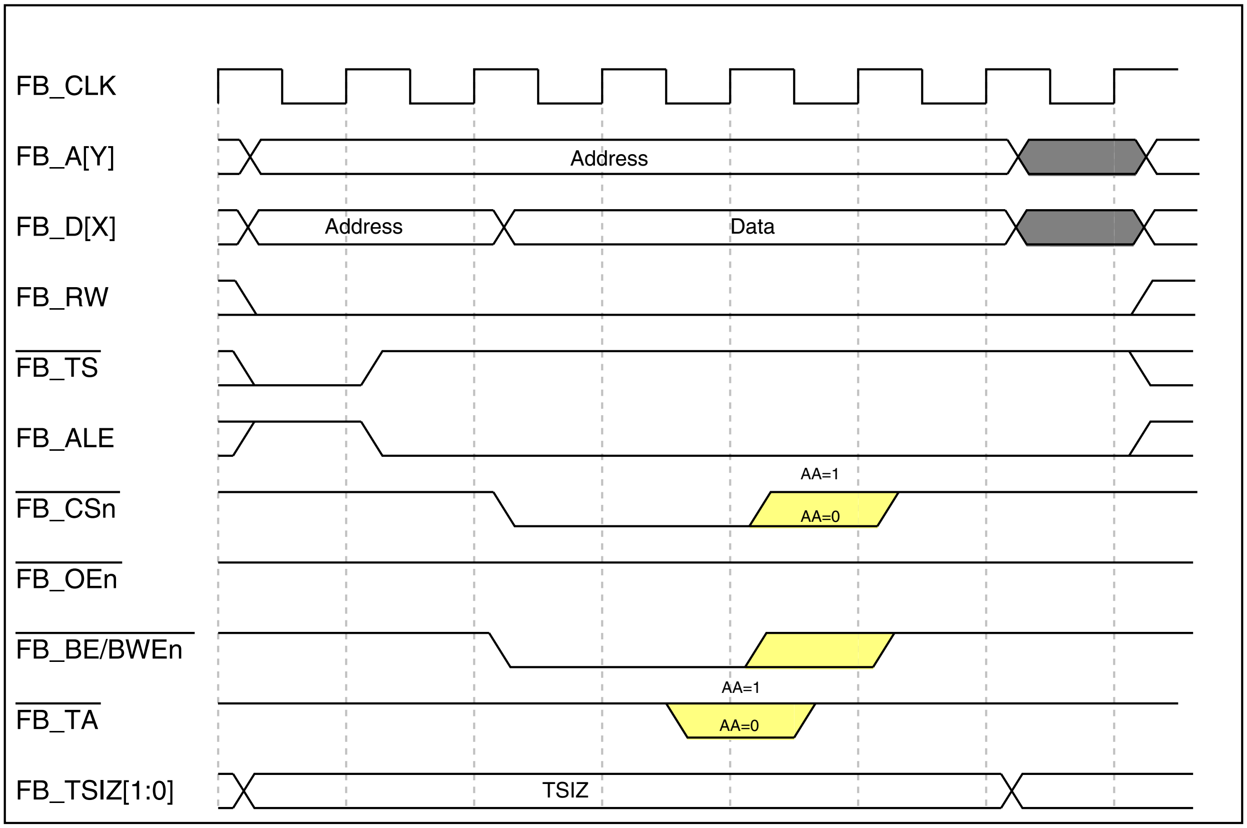


Figure 33-44. Write Cycle with Two-Clock Address Setup and Two-Clock Hold (One Wait State)

33.4.7 Burst Cycles

The device 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 size pins. 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 would take a 4-byte burst cycle, for which FB_TSIZ[1:0] equals 00b throughout.

With bursting disabled, any transfer larger than the port size breaks into multiple individual transfers. With bursting enabled, an access larger than port size results in a burst cycle of multiple beats. The following table shows the result of such transfer translations.

Table 33-30. Transfer Size and Port Size Translation

Port Size PS[1:0]	Transfer Size FB_TSIZ[1:0]	Burst-Inhibited: Number of Transfers Burst Enabled: Number of Beats
01 (8-bit)	10 (16-bits)	2
	00 (32-bits)	4
	11 (16 bytes)	16
1*x* (16-bit)	00 (32 bits)	2
	11 (16 bytes)	8
00 (32-bit)	11 (line)	4

The FlexBus can support 2-1-1-1 burst cycles to maximize system performance. 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.

The CSCR*n* registers enable bursting for reads, writes, or both. Memory spaces can be declared burst-inhibited for reads and writes by clearing the appropriate CSCRn[BSTR,BSTW] bits.

The following figure shows a 32-bit read to an 8-bit device 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 (00) 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.

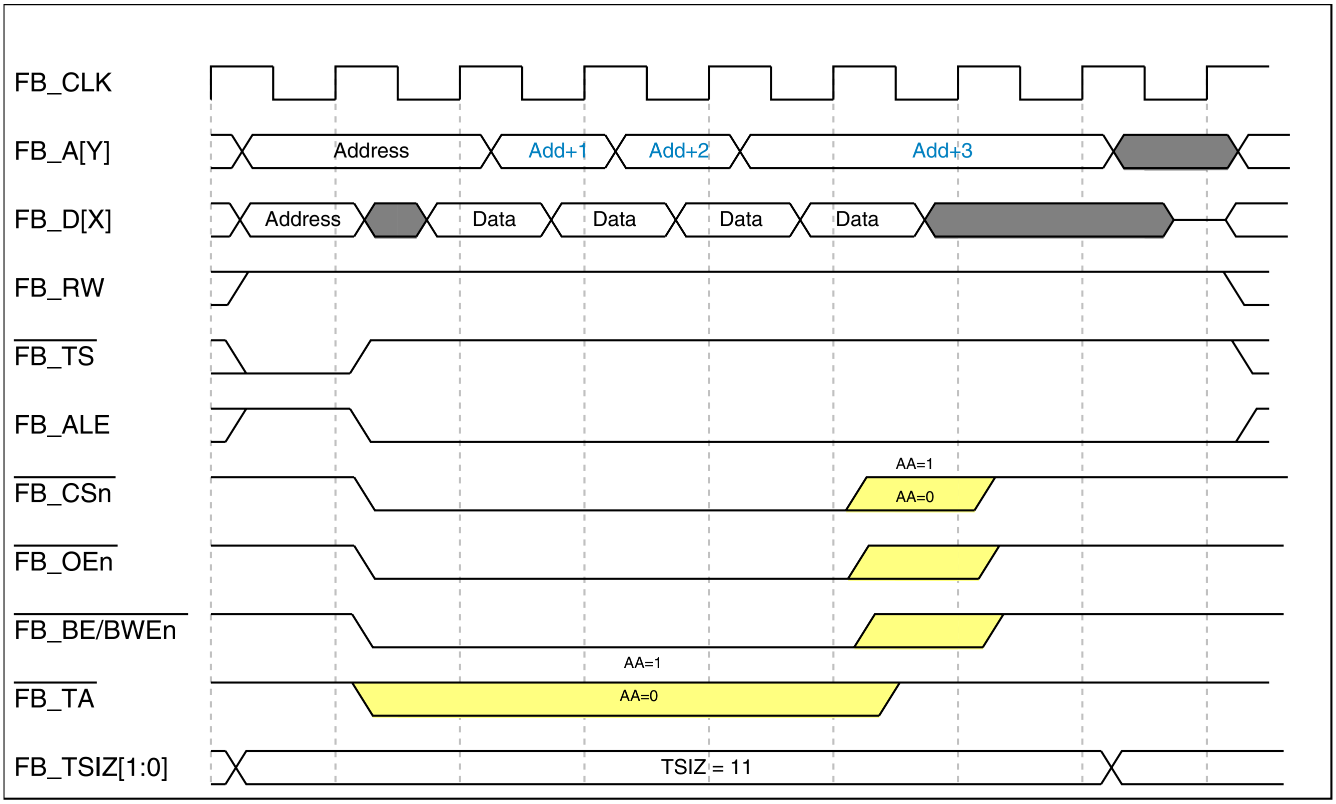


Figure 33-45. 32-bit-Read Burst from 8-Bit Port 2-1-1-1 (No Wait States)

The following figure shows a 32-bit write to an 8-bit device 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 (00) 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 (CSCR*n*[WS] = 0), one wait state is added. Otherwise, the programmed number of wait states are used.

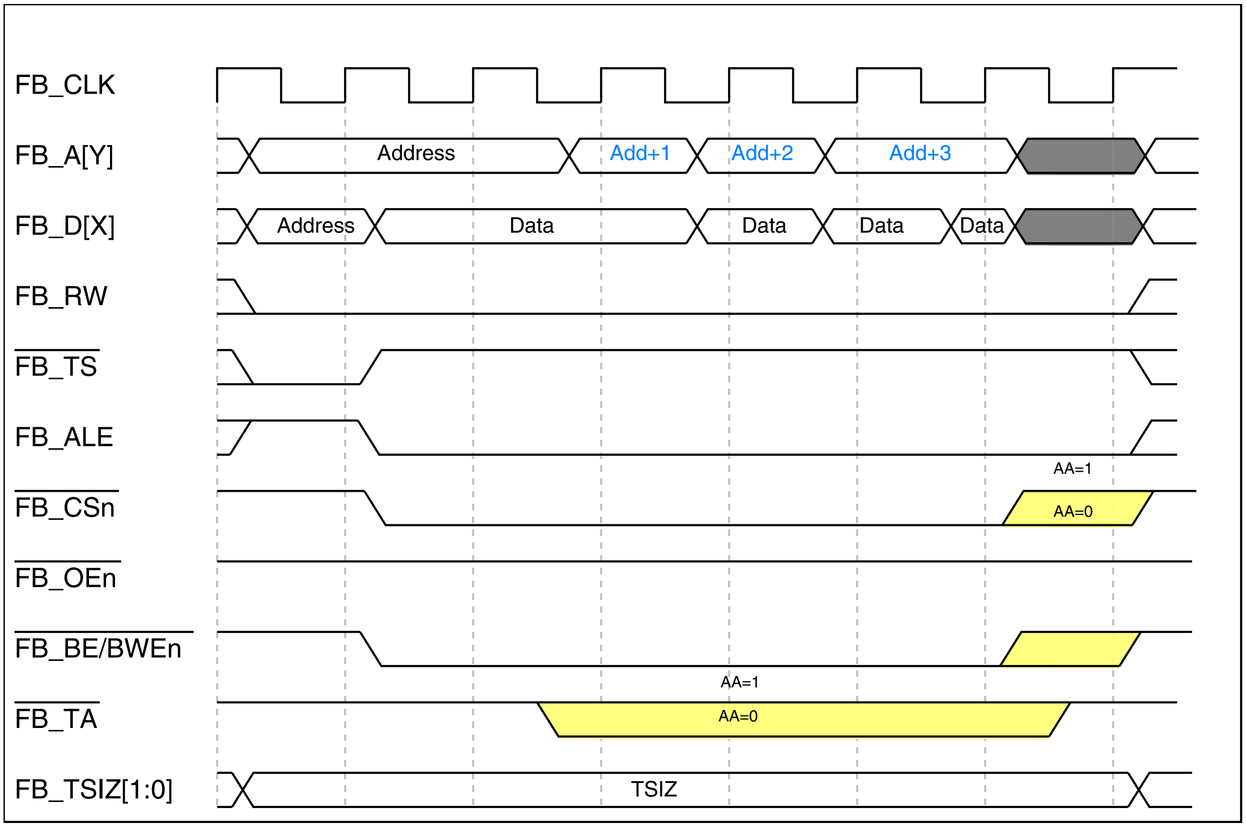


Figure 33-46. 32-bit-Write Burst to 8-Bit Port 3-1-1-1 (No Wait States)

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 (00) during the first transfer and at byte (01) 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.

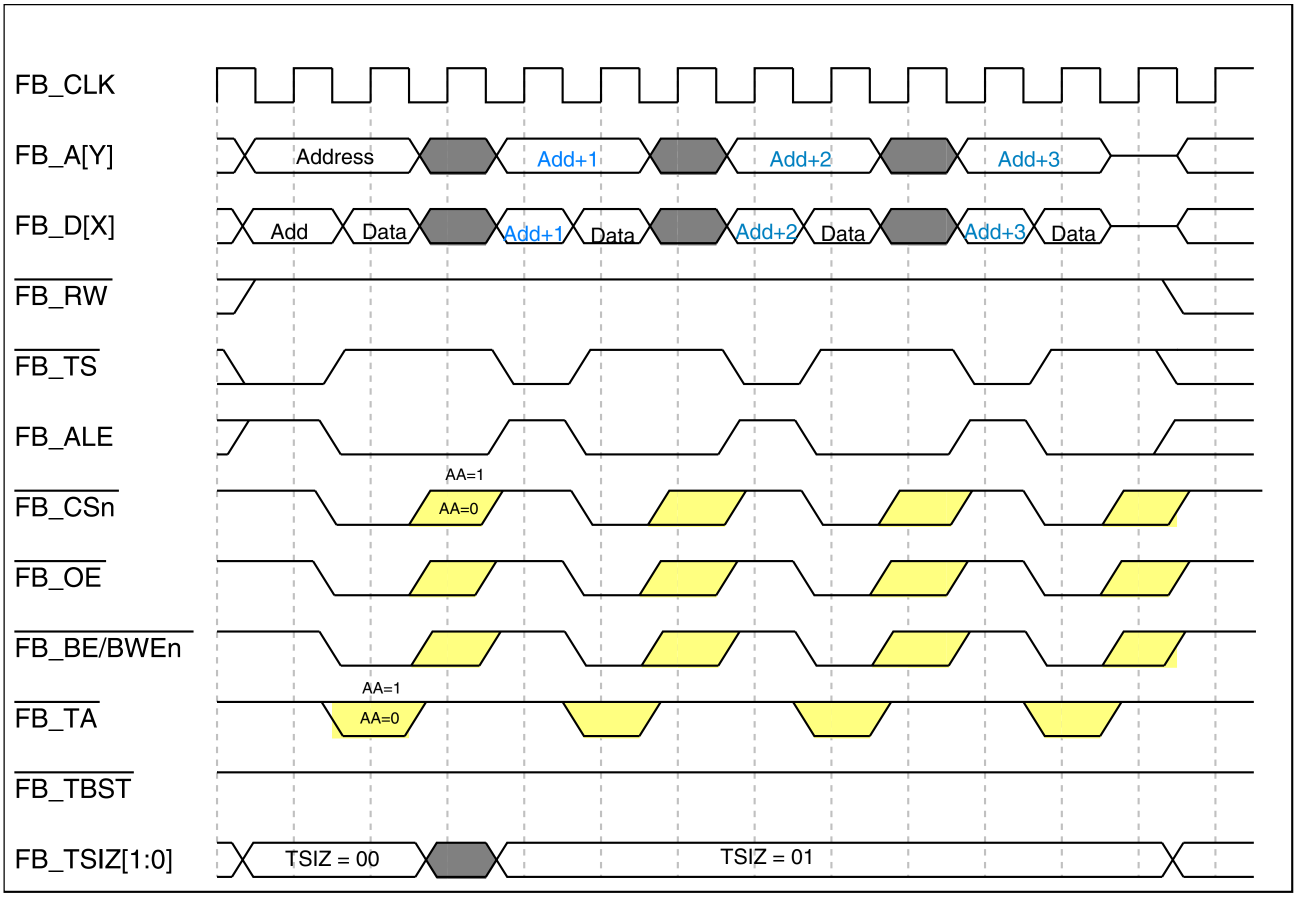


Figure 33-47. 32-bit-Read Burst-Inhibited from 8-Bit Port (No Wait States)

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 (00) during the first transfer and at byte (01) during the next three transfers.

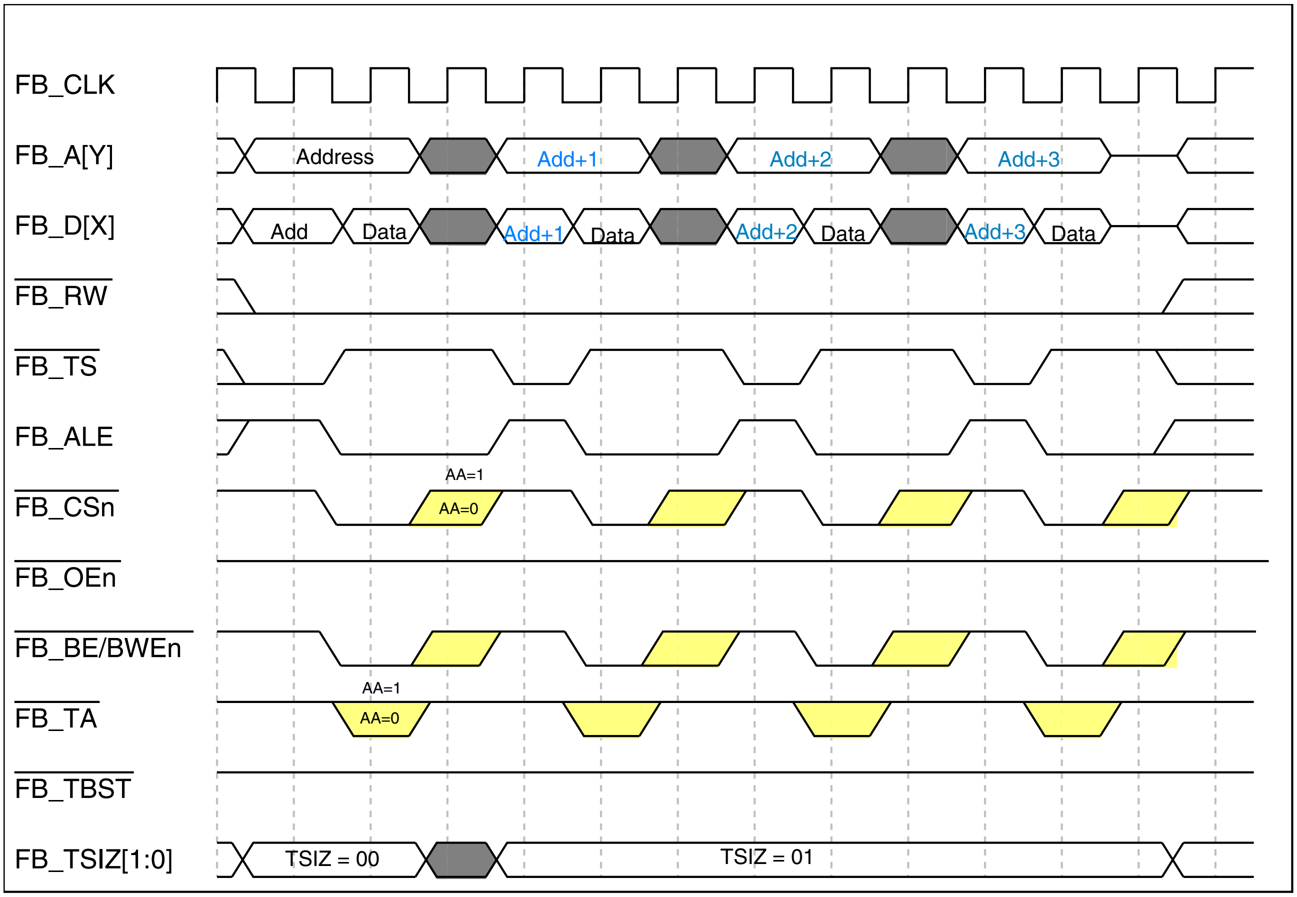


Figure 33-48. 32-bit-Write Burst-Inhibited to 8-Bit Port (No Wait States)

The following figure illustrates another read burst transfer, but in this case a wait state is added between individual beats.

Note

> CSCR*n*[WS] determines the number of wait states in the first beat. However, for subsequent beats, the CSCR*n*[WS] (or CSCR*n*[SWS] if CSCR*n*[SWSEN] is set) determines the number of wait states.

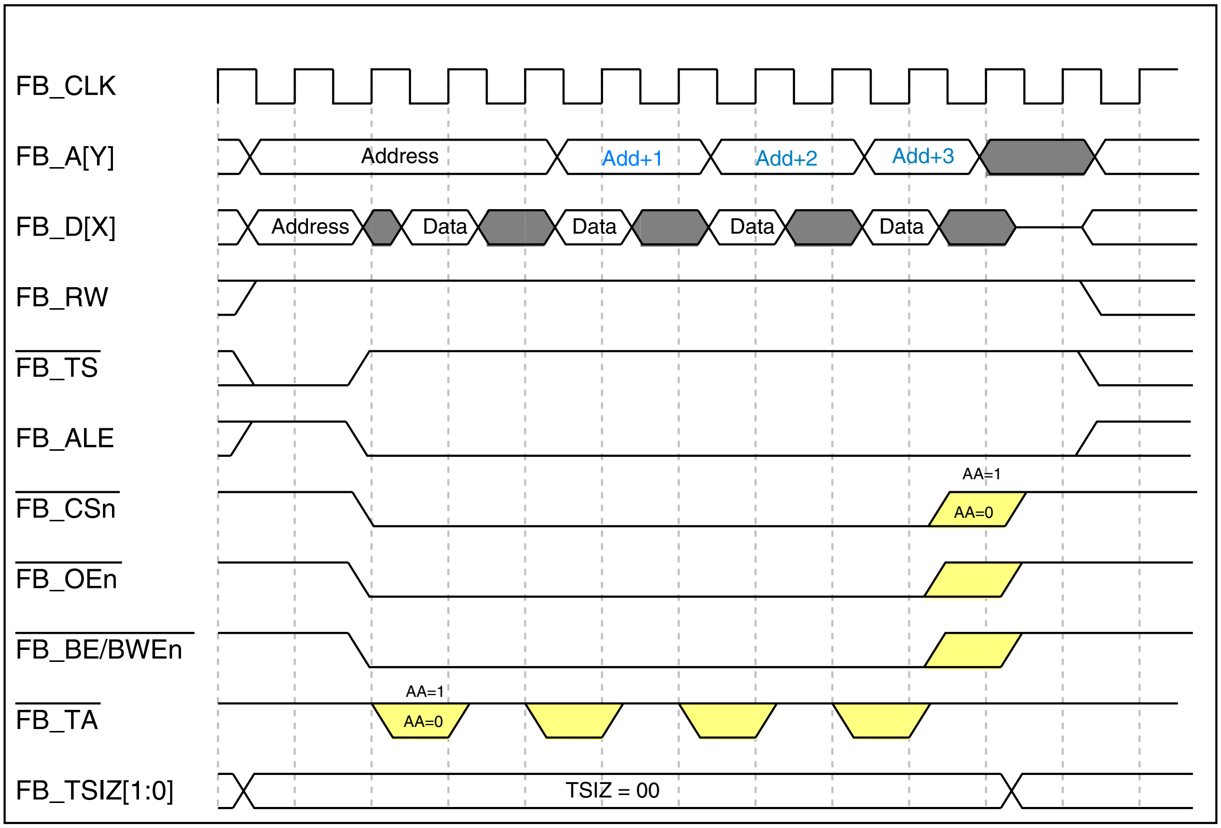


Figure 33-49. 32-bit-Read Burst from 8-Bit Port 3-2-2-2 (One Wait State)

The following figure illustrates a write burst transfer with one wait state.

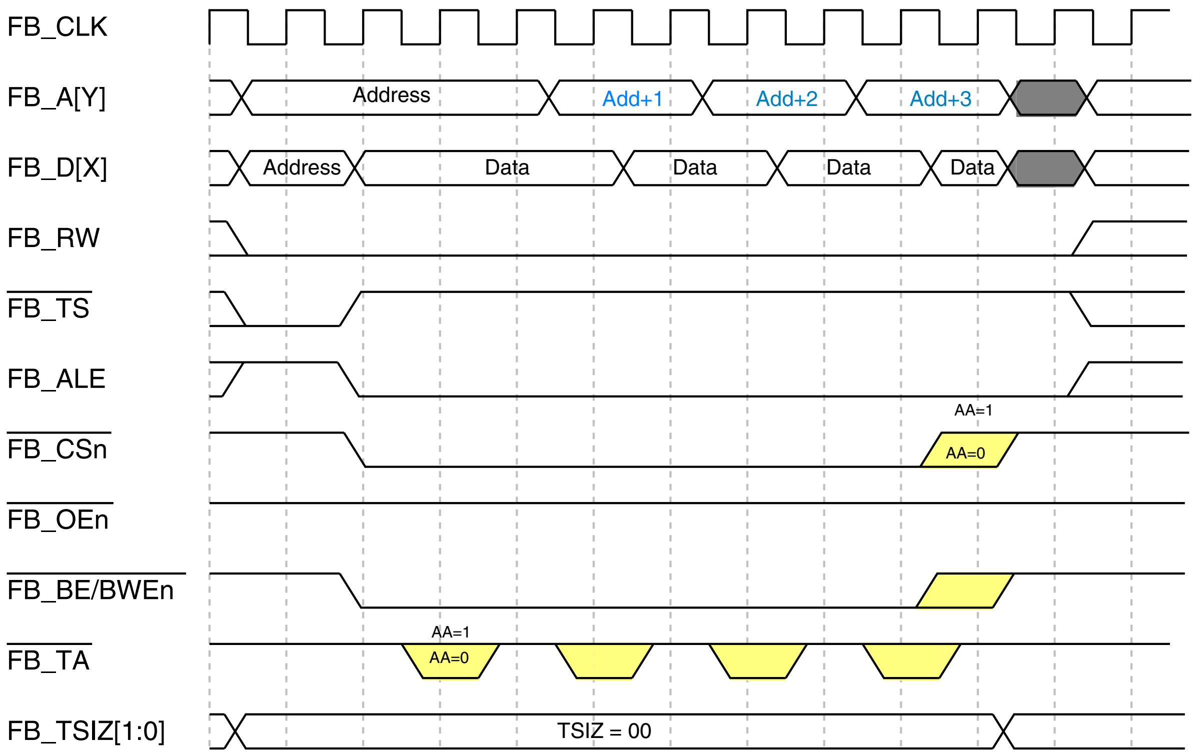


Figure 33-50. 32-bit-Write Burst to 8-Bit Port 3-2-2-2 (One Wait State)

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 non-multiplexed address/data mode, the address on FB_A increments only during internally-terminated burst cycles (CSCR*n*[AA] = 1). The attached device must be able to account for this, or a wait state must be added. 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 internally- and externally-terminated cycles.

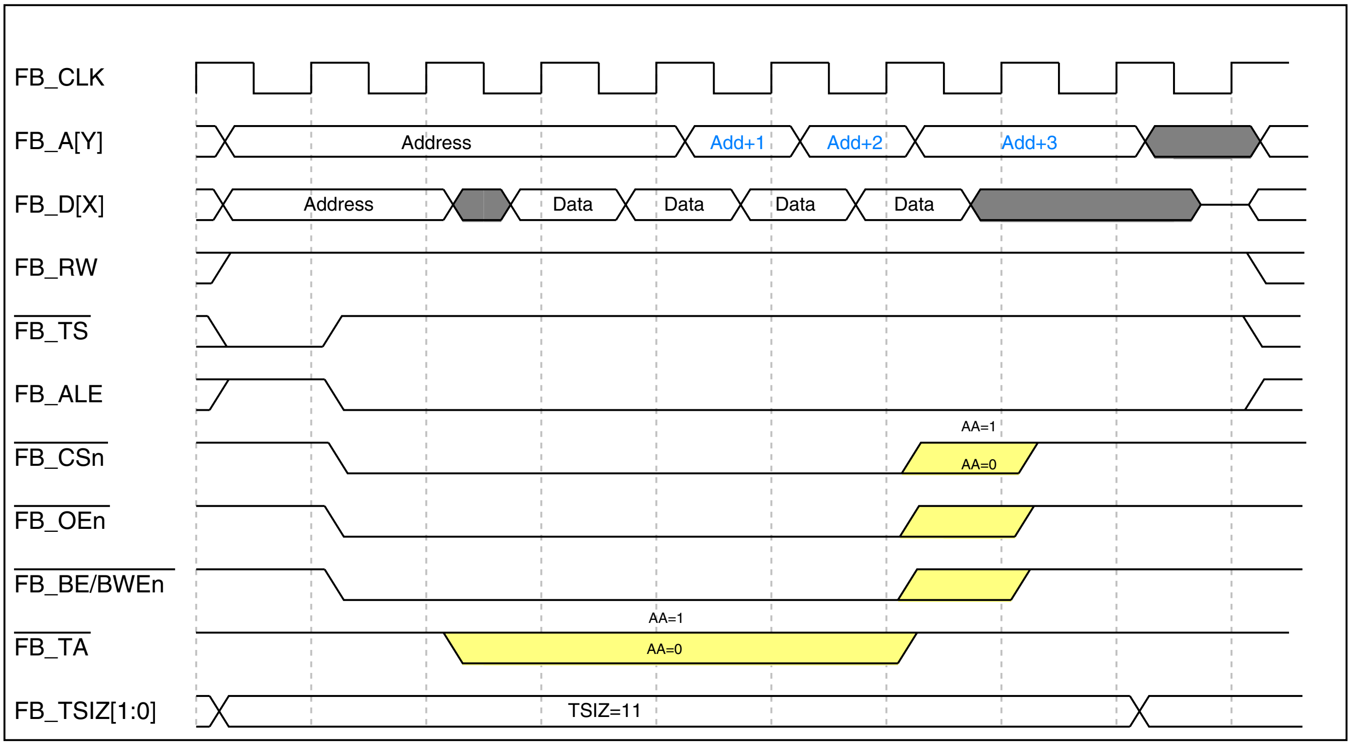


Figure 33-51. 32-bit-Read Burst from 8-Bit Port 3-1-1-1 (Address Setup and Hold)

The following figure shows a write cycle with one clock of address setup and address hold.

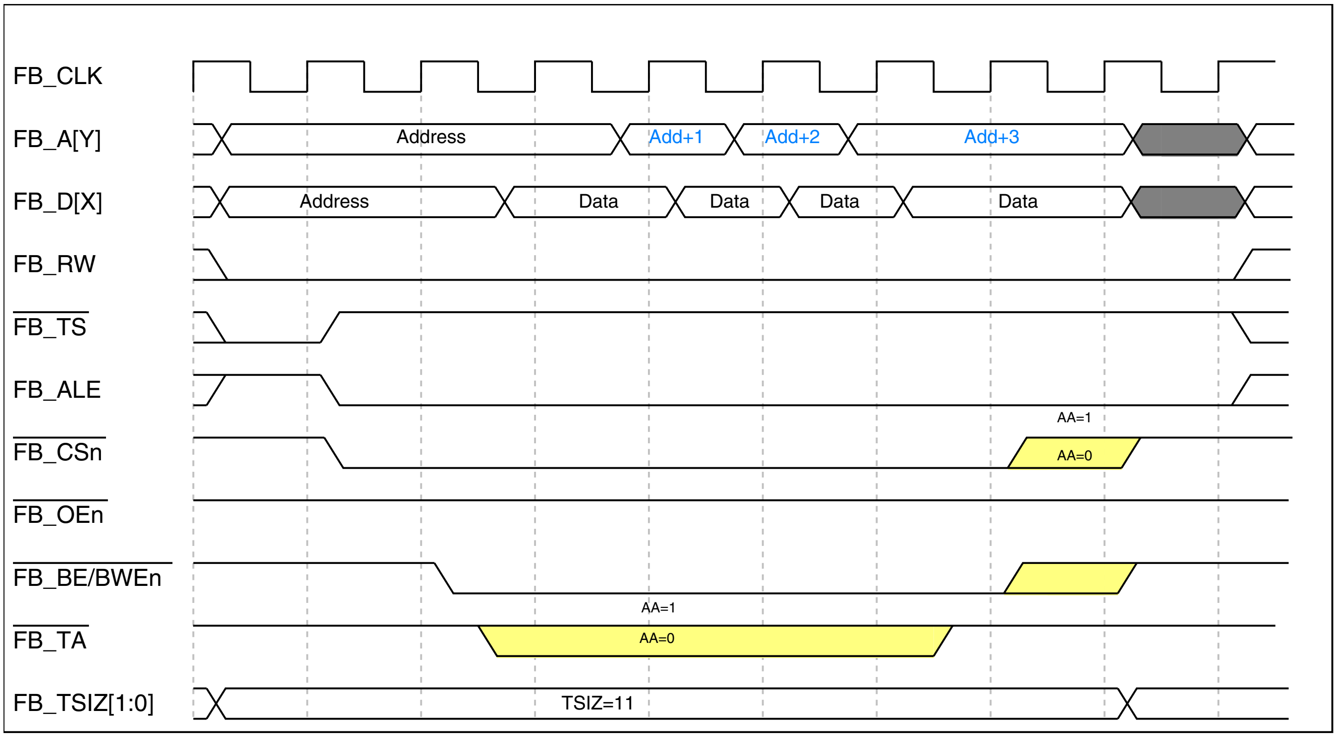


Figure 33-52. 32-bit-Write Burst to 8-Bit Port 3-1-1-1 (Address Setup and Hold)

33.4.8 Extended Transfer Start/Address Latch Enable

The FB_TS/FB_ALE signal indicates that a bus transaction has begun and the address and attributes are valid. By default, the FB_TS/FB_ALE signal asserts for a single bus clock cycle. When CSCR*n*[EXTS] is set, the FB_TS/FB_ALE signal asserts and remain asserted until the first positive clock edge after FB_CS*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.

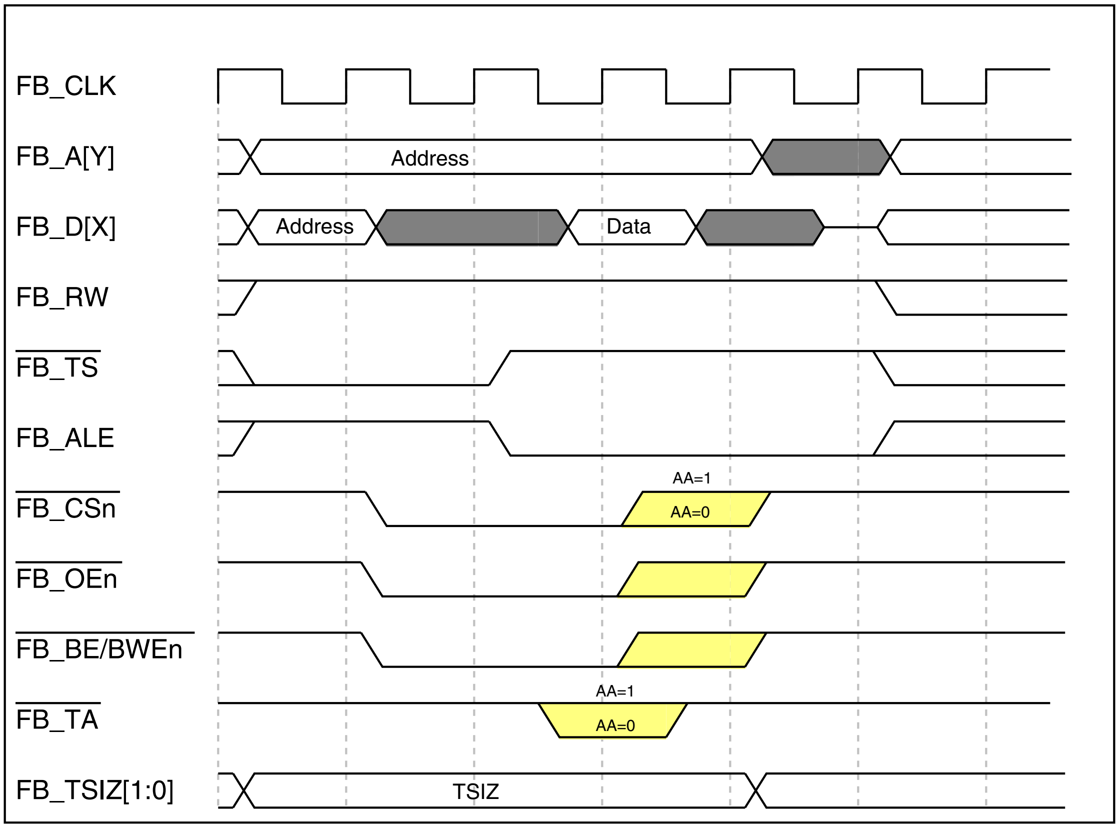


Figure 33-53. Read-Bus Cycle with CSCR*n*[EXTS] = 1 (One Wait State)

33.4.9 Bus Errors

If the auto-acknowledge feature is not enabled for the address that generates the error, the bus cycle can be terminated by asserting FB_TA. If the processor must manage a bus error differently, asserting an interrupt to the core along with FB_TA when the bus error occurs can invoke an interrupt handler.

The types of accesses that cause the access to terminate with a bus error are:

- Writes to write-protected region
- Address with no hit to any chip select
- Address with hits to multiple chip selects
- Writes to reserved addresses in the memory map
- Writes to reserved bits in the CSPMCR register
- FlexBus accesses when the FlexBus is secure

Also, the device can hang if the FlexBus is configured for external termination and the CSPMCR is not configured for $\overline{\text{FB\_TA}}$.

33.5 Initialization/Application Information

33.5.1 Initializing a Chip Select

To initially use a chip select:

1. Configure the CSAR register.
2. Configure the CSCR register.
3. Configure the CSMR register, setting the valid bit.

The CSPMCR register is not required to be part of this procedure. However, it should only be configured when the FlexBus is idle. The corresponding chip select can be valid.

33.5.2 Reconfiguring a Chip Select

To reconfigure a previously-used chip select, the chip select must be specified as invalid as shown below:

1. Clear the CSMR register's valid bit.
2. Change settings in the CSAR register as necessary.
3. Change settings in the CSCR register as necessary.
4. Change settings in the CSMR register as necessary, and set the valid bit.

The CSPMCR register is not required to be part of this procedure. However, it should only be altered when the FlexBus is idle. The corresponding chip select can be valid.

Chapter 34
Cyclic redundancy check (CRC)

34.1 Introduction

NOTE

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

The cyclic redundancy check (CRC) module generates 16/32-bit CRC code for error detection.

The CRC module provides a programmable polynomial, WAS, 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.

34.1.1 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.

34.1.2 Block diagram

This is a block diagram of the CRC.

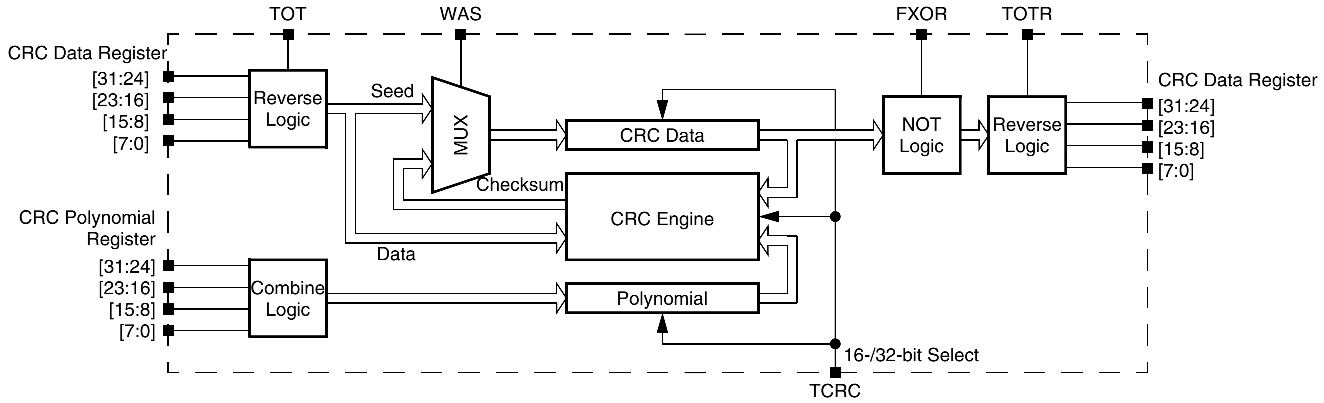


Figure 34-1. Programmable cyclic redundancy check (CRC) block diagram

34.1.3 Modes of operation

Various MCU modes affect the CRC module's functionality.

34.1.3.1 Run mode

This is the basic mode of operation.

34.1.3.2 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 MCU dependent.

34.2 Memory map and register descriptions

CRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_2000	CRC Data Register (CRC_CRC)	32	R/W	FFFF_ FFFFh	34.2.1/ 899
4003_2004	CRC Polynomial Register (CRC_GPOLY)	32	R/W	0000_1021h	34.2.2/ 900
4003_2008	CRC Control Register (CRC_CTRL)	32	R/W	0000_0000h	34.2.3/ 901

34.2.1 CRC Data Register (CRC_CRC)

The CRC data register contains the value of the seed, data, and checksum. When the CTRL[WAS] bit is set, any write to the data register is regarded as the seed value. When the CTRL[WAS] bit 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.

Address: CRC_CRC is 4003_2000h base + 0h offset = 4003_2000h

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	HU								HL								LU								LL							
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

CRC_CRC field descriptions

Field	Description
31–24 HU	CRC High Upper Byte In 16-bit CRC mode (the CTRL[TCRC] bit is 0), this field is not used for programming a seed value. In 32-bit CRC mode (the CTRL[TCRC] bit is 1), values written to this field are part of the seed value when the CTRL[WAS] bit is 1. When the CTRL[WAS] bit is 0, data written to this field is used for CRC checksum generation in both 16-bit and 32-bit CRC modes.

Table continues on the next page...

CRC_CRC field descriptions (continued)

Field	Description
23–16 HL	CRC High Lower Byte In 16-bit CRC mode (the CTRL[TCRC] bit is 0), this field is not used for programming a seed value. In 32-bit CRC mode (the CTRL[TCRC] bit is 1), values written to this field are part of the seed value when the CTRL[WAS] bit is 1. When the CTRL[WAS] bit 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 the CTRL[WAS] bit is 1, values written to this field are part of the seed value. When the CTRL[WAS] bit is 0, data written to this field is used for CRC checksum generation.
7–0 LL	CRC Low Lower Byte When the CTRL[WAS] bit is 1, values written to this field are part of the seed value. When the CTRL[WAS] bit is 0, data written to this field is used for CRC checksum generation.

34.2.2 CRC Polynomial Register (CRC_GPOLY)

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.

Address: CRC_GPOLY is 4003_2000h base + 4h offset = 4003_2004h

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	HIGH																LOW															
Reset	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	1	0	0	0	0	1

CRC_GPOLY field descriptions

Field	Description
31–16 HIGH	High polynominal half-word This field is writable and readable in 32-bit CRC mode (the CTRL[TCRC] bit is 1). This field is not writable in 16-bit CRC mode (the CTRL[TCRC] bit is 0).
15–0 LOW	Low polynominal half-word This field is writable and readable in both 32-bit and 16-bit CRC modes.

34.2.3 CRC Control Register (CRC_CTRL)

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 the CTRL[WAS] bit and then writing the seed into the CRC data register.

Address: CRC_CTRL is 4003_2000h base + 8h offset = 4003_2008h

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	TOT		TOTR		0	FXOR	WAS	TCRC	0																							
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

CRC_CTRL field descriptions

Field	Description
31–30 TOT	Type of Transpose for Writes These bits define the transpose configuration of the data written to the CRC data register. Refer to the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
29–28 TOTR	Type of Transpose for Read These bits identify the transpose configuration of the value read from the CRC data register. Refer to the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
27 Reserved	This read-only field is reserved and always has the value zero.
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. 0 No XOR on reading. 1 Invert or complement the read value of the CRC data register.
25 WAS	Write CRC data register as seed When this bit is asserted, a value written to the CRC data register is considered a seed value. When this bit is de-asserted, a value written to the CRC data register is taken as data for CRC computation.

Table continues on the next page...

CRC_CTRL field descriptions (continued)

Field	Description
	0 Writes to the CRC data register are data values. 1 Writes to the CRC data register are seed values.
24 TCRC	Width of CRC protocol. 0 16-bit CRC protocol. 1 32-bit CRC protocol.
23–0 Reserved	This read-only field is reserved and always has the value zero.

34.3 Functional description

34.3.1 CRC initialization/re-initialization

To enable the CRC calculation, the user must program the WAS, POLYNOMIAL, and necessary parameters for transpose and CRC result inversion in the applicable registers. Asserting the CTRL[WAS] bit enables the programming of the seed value into the CRC data register.

After a completed CRC calculation, re-asserting the CTRL[WAS] bit and programming a seed (whether the value is new or a previously used seed value) re-initialize the CRC module for a new CRC computation. All other parameters must be set before programming the seed value and subsequent data values.

34.3.2 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. Non-contiguous bytes can lead to an incorrect CRC computation.

34.3.2.1 16-bit CRC

Compute a 16-bit CRC with the following steps:

1. Clear the CTRL[TCRC] bit 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 GPOLY[LOW] field. The GPOLY[HIGH] field is not usable in 16-bit CRC mode.
4. Set the CTRL[WAS] bit to program the seed value.
5. Write a 16-bit seed to CRC[LU:LL]. CRC[HU:HL] are not used.
6. Clear the CTRL[WAS] bit to start writing data values.
7. Write data values into CRC[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC[LU:LL].
8. When all values have been written, read the final CRC result from CRC[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.

34.3.2.2 32-bit CRC

Compute a 32-bit CRC with the following steps:

1. Set the CTRL[TCRC] bit 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 GPOLY[HIGH:LOW].
4. Set the CTRL[WAS] bit to program the seed value.
5. Write a 32-bit seed to CRC[HU:HL:LU:LL].
6. Clear the CTRL[WAS] bit to start writing data values.
7. Write data values into CRC[HU:HL:LU:LL]. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC[HU:HL:LU:LL].
8. When all values have been written, read the final CRC result from CRC[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.

34.3.3 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.

34.3.3.1 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]}

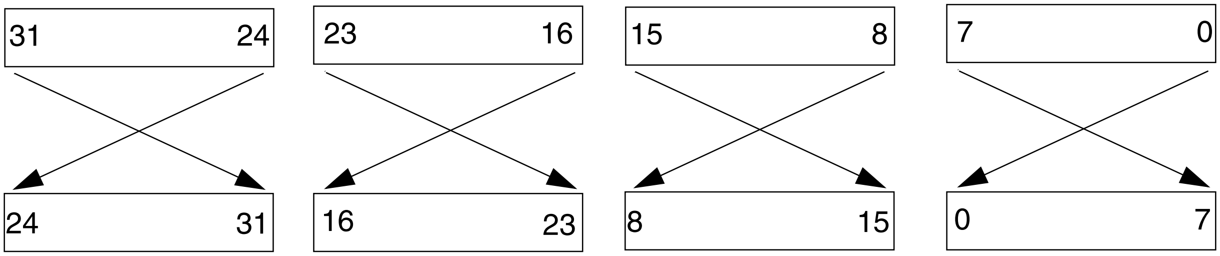


Figure 34-5. Transpose type 01

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]}

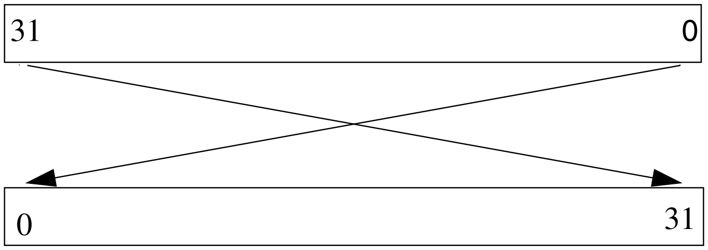


Figure 34-6. Transpose type 10

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]}

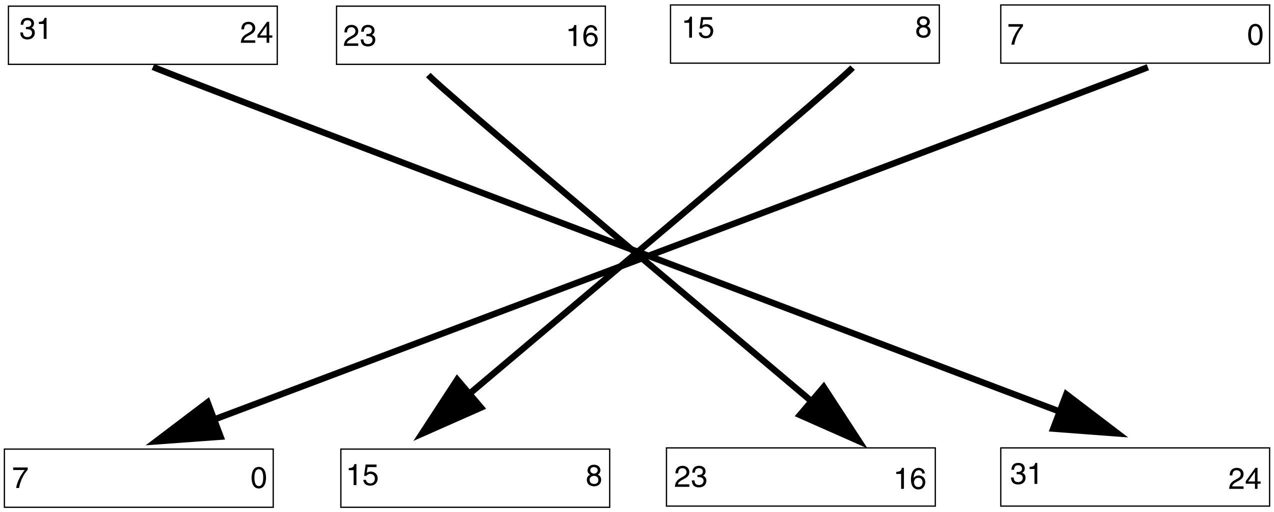


Figure 34-7. Transpose type 11

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.

34.3.4 CRC result complement

When the CTRL[FXOR] bit 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 the CTRL[FXOR] bit is cleared, reading the CRC data register accesses the raw checksum value.

Chapter 35
Analog-to-Digital Converter (ADC)

35.1 Introduction

NOTE

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

The 16-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, refer to the Power Management information for the device.

35.1.1 Features

Features of the ADC module include:

- Linear successive approximation algorithm with up to 16-bit resolution
- Up to 4 pairs of differential and 24 single-ended external analog inputs
- Output modes: differential 16-bit, 13-bit, 11-bit and 9-bit modes, or single-ended 16-bit, 12-bit, 10-bit and 8-bit modes
- Output formatted in 2's complement 16-bit sign extended for differential modes
- Output in right-justified unsigned format for single-ended
- Single or continuous conversion (automatic return to idle after single conversion)
- Configurable sample time and conversion speed/power
- Conversion complete / hardware average complete flag and interrupt

- Input clock selectable from up to four sources
- Operation in low power modes for lower noise operation
- Asynchronous clock source for lower noise operation with option to output the clock
- Selectable hardware conversion trigger with hardware channel select
- Automatic compare with interrupt for less-than, greater-than or equal-to, within range, or out-of-range, programmable value
- Temperature sensor
- Hardware average function
- Selectable voltage reference: external or alternate
- Self-calibration mode
- Programmable Gain Amplifier (PGA) with up to x64 gain

35.1.2 Block diagram

The following figure is the ADC module block diagram.

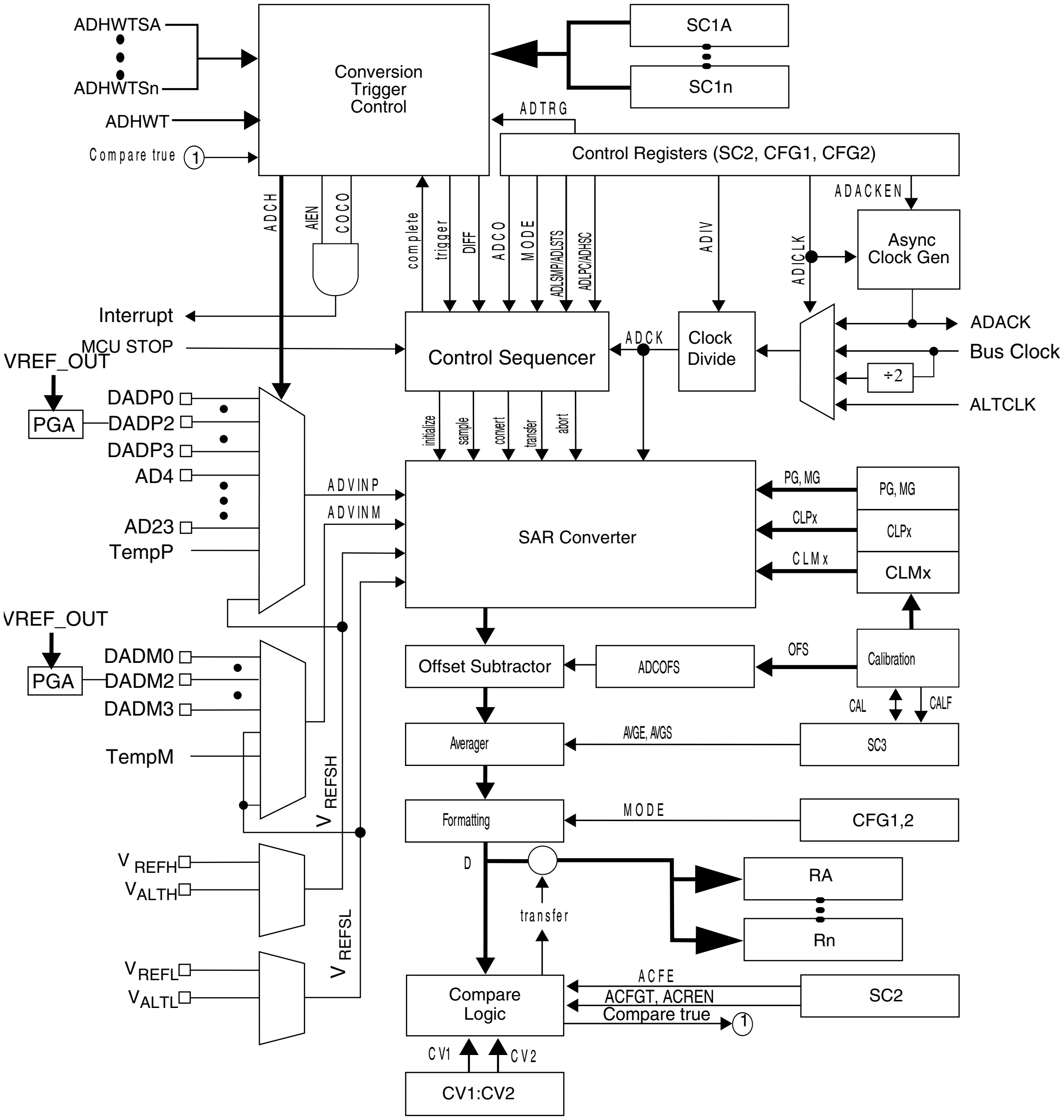


Figure 35-1. ADC block diagram

35.2 ADC Signal Descriptions

The ADC module supports up to 4 pairs of differential inputs and up to 24 single-ended inputs. Each differential pair requires two inputs, DADPx and DADMx. The ADC also requires four supply/reference/ground connections.

Table 35-1. ADC Signal Descriptions

Signal	Description	I/O
DADP[3:0]	Differential analog channel inputs	I

Table continues on the next page...

Table 35-1. ADC Signal Descriptions (continued)

Signal	Description	I/O
DADM[3:0]	Differential analog channel inputs	I
AD[23:4]	Single-ended analog channel inputs	I
V_{REFSH}	Voltage reference select high	I
V_{REFSL}	Voltage reference select low	I
V_{DDA}	Analog power supply	I
V_{SSA}	Analog ground	I

35.2.1 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.

35.2.2 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}.

35.2.3 Voltage reference select

V_{REFSH} and V_{REFSL} are the high and low reference voltages for the converter.

The ADC can be configured to accept one of two voltage reference pairs for V_{REFSH} and V_{REFSL}. Each pair contains a positive reference that must be between the minimum Ref Voltage High and V_{DDA}, and a ground reference that must be at the same potential as V_{SSA}. The two pairs are external (V_{REFH} and V_{REFL}) and alternate (V_{ALTH} and V_{ALTL}). These voltage references are selected using the REFSEL bits in the SC2 register. The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration. Refer to the Chip Configuration information on the Voltage References specific to this MCU.

In some packages, V_{REFH} is connected in the package to V_{DDA} and V_{REFL} to V_{SSA}. If externally available, the positive reference(s) may be connected to the same potential as V_{DDA} or may be driven by an external source to a level between the minimum Ref Voltage High and the V_{DDA} potential (V_{REFH} must never exceed V_{DDA}). Connect the ground references to the same voltage potential as V_{SSA}.

35.2.4 Analog channel inputs (ADx)

The ADC module supports up to 24 single-ended analog inputs. A single-ended input is selected for conversion through the ADCH channel select bits when the DIFF bit in the SC1n register is low.

35.2.5 Differential analog channel inputs (DADx)

The ADC module supports up to 4 differential analog channel inputs. Each differential analog input is a pair of external pins (DADPx and DADMx) referenced to each other to provide the most accurate analog to digital readings. A differential input is selected for conversion through the ADCH channel select bits when the DIFF bit in the SC1n register bit is high. All DADPx inputs may be used as single-ended inputs if the DIFF bit is low. In certain MCU configurations, some DADMx inputs may also be used as single-ended inputs if the DIFF bit is low. Refer to the Chip Configuration chapter for ADC connections specific to this MCU.

35.3 Register Definition

This section describes the ADC registers.

ADC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_B000	ADC status and control registers 1 (ADC0_SC1A)	32	R/W	0000_001Fh	35.3.1/ 917
4003_B004	ADC status and control registers 1 (ADC0_SC1B)	32	R/W	0000_001Fh	35.3.1/ 917
4003_B008	ADC configuration register 1 (ADC0_CFG1)	32	R/W	0000_0000h	35.3.2/ 920

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_B00C	Configuration register 2 (ADC0_CFG2)	32	R/W	0000_0000h	35.3.3/ 922
4003_B010	ADC data result register (ADC0_RA)	32	R	0000_0000h	35.3.4/ 923
4003_B014	ADC data result register (ADC0_RB)	32	R	0000_0000h	35.3.4/ 923
4003_B018	Compare value registers (ADC0_CV1)	32	R/W	0000_0000h	35.3.5/ 925
4003_B01C	Compare value registers (ADC0_CV2)	32	R/W	0000_0000h	35.3.5/ 925
4003_B020	Status and control register 2 (ADC0_SC2)	32	R/W	0000_0000h	35.3.6/ 926
4003_B024	Status and control register 3 (ADC0_SC3)	32	R/W	0000_0000h	35.3.7/ 928
4003_B028	ADC offset correction register (ADC0_OFS)	32	R/W	0000_0004h	35.3.8/ 929
4003_B02C	ADC plus-side gain register (ADC0_PG)	32	R/W	0000_8200h	35.3.9/ 930
4003_B030	ADC minus-side gain register (ADC0_MG)	32	R/W	0000_8200h	35.3.10/ 930
4003_B034	ADC plus-side general calibration value register (ADC0_CLPD)	32	R/W	0000_000Ah	35.3.11/ 931
4003_B038	ADC plus-side general calibration value register (ADC0_CLPS)	32	R/W	0000_0020h	35.3.12/ 932
4003_B03C	ADC plus-side general calibration value register (ADC0_CLP4)	32	R/W	0000_0200h	35.3.13/ 932
4003_B040	ADC plus-side general calibration value register (ADC0_CLP3)	32	R/W	0000_0100h	35.3.14/ 933
4003_B044	ADC plus-side general calibration value register (ADC0_CLP2)	32	R/W	0000_0080h	35.3.15/ 934
4003_B048	ADC plus-side general calibration value register (ADC0_CLP1)	32	R/W	0000_0040h	35.3.16/ 934
4003_B04C	ADC plus-side general calibration value register (ADC0_CLP0)	32	R/W	0000_0020h	35.3.17/ 935
4003_B050	ADC PGA register (ADC0_PGA)	32	R/W	0000_0000h	35.3.18/ 936
4003_B054	ADC minus-side general calibration value register (ADC0_CLMD)	32	R/W	0000_000Ah	35.3.19/ 937
4003_B058	ADC minus-side general calibration value register (ADC0_CLMS)	32	R/W	0000_0020h	35.3.20/ 938

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_B05C	ADC minus-side general calibration value register (ADC0_CLM4)	32	R/W	0000_0200h	35.3.21/ 939
4003_B060	ADC minus-side general calibration value register (ADC0_CLM3)	32	R/W	0000_0100h	35.3.22/ 939
4003_B064	ADC minus-side general calibration value register (ADC0_CLM2)	32	R/W	0000_0080h	35.3.23/ 940
4003_B068	ADC minus-side general calibration value register (ADC0_CLM1)	32	R/W	0000_0040h	35.3.24/ 940
4003_B06C	ADC minus-side general calibration value register (ADC0_CLM0)	32	R/W	0000_0020h	35.3.25/ 941
400B_B000	ADC status and control registers 1 (ADC1_SC1A)	32	R/W	0000_001Fh	35.3.1/ 917
400B_B004	ADC status and control registers 1 (ADC1_SC1B)	32	R/W	0000_001Fh	35.3.1/ 917
400B_B008	ADC configuration register 1 (ADC1_CFG1)	32	R/W	0000_0000h	35.3.2/ 920
400B_B00C	Configuration register 2 (ADC1_CFG2)	32	R/W	0000_0000h	35.3.3/ 922
400B_B010	ADC data result register (ADC1_RA)	32	R	0000_0000h	35.3.4/ 923
400B_B014	ADC data result register (ADC1_RB)	32	R	0000_0000h	35.3.4/ 923
400B_B018	Compare value registers (ADC1_CV1)	32	R/W	0000_0000h	35.3.5/ 925
400B_B01C	Compare value registers (ADC1_CV2)	32	R/W	0000_0000h	35.3.5/ 925
400B_B020	Status and control register 2 (ADC1_SC2)	32	R/W	0000_0000h	35.3.6/ 926
400B_B024	Status and control register 3 (ADC1_SC3)	32	R/W	0000_0000h	35.3.7/ 928
400B_B028	ADC offset correction register (ADC1_OFS)	32	R/W	0000_0004h	35.3.8/ 929
400B_B02C	ADC plus-side gain register (ADC1_PG)	32	R/W	0000_8200h	35.3.9/ 930
400B_B030	ADC minus-side gain register (ADC1_MG)	32	R/W	0000_8200h	35.3.10/ 930
400B_B034	ADC plus-side general calibration value register (ADC1_CLPD)	32	R/W	0000_000Ah	35.3.11/ 931
400B_B038	ADC plus-side general calibration value register (ADC1_CLPS)	32	R/W	0000_0020h	35.3.12/ 932

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400B_B03C	ADC plus-side general calibration value register (ADC1_CLP4)	32	R/W	0000_0200h	35.3.13/ 932
400B_B040	ADC plus-side general calibration value register (ADC1_CLP3)	32	R/W	0000_0100h	35.3.14/ 933
400B_B044	ADC plus-side general calibration value register (ADC1_CLP2)	32	R/W	0000_0080h	35.3.15/ 934
400B_B048	ADC plus-side general calibration value register (ADC1_CLP1)	32	R/W	0000_0040h	35.3.16/ 934
400B_B04C	ADC plus-side general calibration value register (ADC1_CLP0)	32	R/W	0000_0020h	35.3.17/ 935
400B_B050	ADC PGA register (ADC1_PGA)	32	R/W	0000_0000h	35.3.18/ 936
400B_B054	ADC minus-side general calibration value register (ADC1_CLMD)	32	R/W	0000_000Ah	35.3.19/ 937
400B_B058	ADC minus-side general calibration value register (ADC1_CLMS)	32	R/W	0000_0020h	35.3.20/ 938
400B_B05C	ADC minus-side general calibration value register (ADC1_CLM4)	32	R/W	0000_0200h	35.3.21/ 939
400B_B060	ADC minus-side general calibration value register (ADC1_CLM3)	32	R/W	0000_0100h	35.3.22/ 939
400B_B064	ADC minus-side general calibration value register (ADC1_CLM2)	32	R/W	0000_0080h	35.3.23/ 940
400B_B068	ADC minus-side general calibration value register (ADC1_CLM1)	32	R/W	0000_0040h	35.3.24/ 940
400B_B06C	ADC minus-side general calibration value register (ADC1_CLM0)	32	R/W	0000_0020h	35.3.25/ 941
4003_C000	ADC status and control registers 1 (ADC2_SC1A)	32	R/W	0000_001Fh	35.3.1/ 917
4003_C004	ADC status and control registers 1 (ADC2_SC1B)	32	R/W	0000_001Fh	35.3.1/ 917
4003_C008	ADC configuration register 1 (ADC2_CFG1)	32	R/W	0000_0000h	35.3.2/ 920
4003_C00C	Configuration register 2 (ADC2_CFG2)	32	R/W	0000_0000h	35.3.3/ 922
4003_C010	ADC data result register (ADC2_RA)	32	R	0000_0000h	35.3.4/ 923
4003_C014	ADC data result register (ADC2_RB)	32	R	0000_0000h	35.3.4/ 923
4003_C018	Compare value registers (ADC2_CV1)	32	R/W	0000_0000h	35.3.5/ 925

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_C01C	Compare value registers (ADC2_CV2)	32	R/W	0000_0000h	35.3.5/ 925
4003_C020	Status and control register 2 (ADC2_SC2)	32	R/W	0000_0000h	35.3.6/ 926
4003_C024	Status and control register 3 (ADC2_SC3)	32	R/W	0000_0000h	35.3.7/ 928
4003_C028	ADC offset correction register (ADC2_OFS)	32	R/W	0000_0004h	35.3.8/ 929
4003_C02C	ADC plus-side gain register (ADC2_PG)	32	R/W	0000_8200h	35.3.9/ 930
4003_C030	ADC minus-side gain register (ADC2_MG)	32	R/W	0000_8200h	35.3.10/ 930
4003_C034	ADC plus-side general calibration value register (ADC2_CLPD)	32	R/W	0000_000Ah	35.3.11/ 931
4003_C038	ADC plus-side general calibration value register (ADC2_CLPS)	32	R/W	0000_0020h	35.3.12/ 932
4003_C03C	ADC plus-side general calibration value register (ADC2_CLP4)	32	R/W	0000_0200h	35.3.13/ 932
4003_C040	ADC plus-side general calibration value register (ADC2_CLP3)	32	R/W	0000_0100h	35.3.14/ 933
4003_C044	ADC plus-side general calibration value register (ADC2_CLP2)	32	R/W	0000_0080h	35.3.15/ 934
4003_C048	ADC plus-side general calibration value register (ADC2_CLP1)	32	R/W	0000_0040h	35.3.16/ 934
4003_C04C	ADC plus-side general calibration value register (ADC2_CLP0)	32	R/W	0000_0020h	35.3.17/ 935
4003_C050	ADC PGA register (ADC2_PGA)	32	R/W	0000_0000h	35.3.18/ 936
4003_C054	ADC minus-side general calibration value register (ADC2_CLMD)	32	R/W	0000_000Ah	35.3.19/ 937
4003_C058	ADC minus-side general calibration value register (ADC2_CLMS)	32	R/W	0000_0020h	35.3.20/ 938
4003_C05C	ADC minus-side general calibration value register (ADC2_CLM4)	32	R/W	0000_0200h	35.3.21/ 939
4003_C060	ADC minus-side general calibration value register (ADC2_CLM3)	32	R/W	0000_0100h	35.3.22/ 939
4003_C064	ADC minus-side general calibration value register (ADC2_CLM2)	32	R/W	0000_0080h	35.3.23/ 940
4003_C068	ADC minus-side general calibration value register (ADC2_CLM1)	32	R/W	0000_0040h	35.3.24/ 940

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_C06C	ADC minus-side general calibration value register (ADC2_CLM0)	32	R/W	0000_0020h	35.3.25/ 941
400B_C000	ADC status and control registers 1 (ADC3_SC1A)	32	R/W	0000_001Fh	35.3.1/ 917
400B_C004	ADC status and control registers 1 (ADC3_SC1B)	32	R/W	0000_001Fh	35.3.1/ 917
400B_C008	ADC configuration register 1 (ADC3_CFG1)	32	R/W	0000_0000h	35.3.2/ 920
400B_C00C	Configuration register 2 (ADC3_CFG2)	32	R/W	0000_0000h	35.3.3/ 922
400B_C010	ADC data result register (ADC3_RA)	32	R	0000_0000h	35.3.4/ 923
400B_C014	ADC data result register (ADC3_RB)	32	R	0000_0000h	35.3.4/ 923
400B_C018	Compare value registers (ADC3_CV1)	32	R/W	0000_0000h	35.3.5/ 925
400B_C01C	Compare value registers (ADC3_CV2)	32	R/W	0000_0000h	35.3.5/ 925
400B_C020	Status and control register 2 (ADC3_SC2)	32	R/W	0000_0000h	35.3.6/ 926
400B_C024	Status and control register 3 (ADC3_SC3)	32	R/W	0000_0000h	35.3.7/ 928
400B_C028	ADC offset correction register (ADC3_OFS)	32	R/W	0000_0004h	35.3.8/ 929
400B_C02C	ADC plus-side gain register (ADC3_PG)	32	R/W	0000_8200h	35.3.9/ 930
400B_C030	ADC minus-side gain register (ADC3_MG)	32	R/W	0000_8200h	35.3.10/ 930
400B_C034	ADC plus-side general calibration value register (ADC3_CLPD)	32	R/W	0000_000Ah	35.3.11/ 931
400B_C038	ADC plus-side general calibration value register (ADC3_CLPS)	32	R/W	0000_0020h	35.3.12/ 932
400B_C03C	ADC plus-side general calibration value register (ADC3_CLP4)	32	R/W	0000_0200h	35.3.13/ 932
400B_C040	ADC plus-side general calibration value register (ADC3_CLP3)	32	R/W	0000_0100h	35.3.14/ 933
400B_C044	ADC plus-side general calibration value register (ADC3_CLP2)	32	R/W	0000_0080h	35.3.15/ 934
400B_C048	ADC plus-side general calibration value register (ADC3_CLP1)	32	R/W	0000_0040h	35.3.16/ 934

Table continues on the next page...

ADC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400B_C04C	ADC plus-side general calibration value register (ADC3_CLP0)	32	R/W	0000_0020h	35.3.17/ 935
400B_C050	ADC PGA register (ADC3_PGA)	32	R/W	0000_0000h	35.3.18/ 936
400B_C054	ADC minus-side general calibration value register (ADC3_CLMD)	32	R/W	0000_000Ah	35.3.19/ 937
400B_C058	ADC minus-side general calibration value register (ADC3_CLMS)	32	R/W	0000_0020h	35.3.20/ 938
400B_C05C	ADC minus-side general calibration value register (ADC3_CLM4)	32	R/W	0000_0200h	35.3.21/ 939
400B_C060	ADC minus-side general calibration value register (ADC3_CLM3)	32	R/W	0000_0100h	35.3.22/ 939
400B_C064	ADC minus-side general calibration value register (ADC3_CLM2)	32	R/W	0000_0080h	35.3.23/ 940
400B_C068	ADC minus-side general calibration value register (ADC3_CLM1)	32	R/W	0000_0040h	35.3.24/ 940
400B_C06C	ADC minus-side general calibration value register (ADC3_CLM0)	32	R/W	0000_0020h	35.3.25/ 941

35.3.1 ADC status and control registers 1 (ADCx_SC1*n*)

The SC1A register is used for both software and hardware trigger modes of operation.

To allow sequential conversions of the ADC to be triggered by internal peripherals, the ADC can have more then one status and control register: one for each conversion. The SC1B-SC1n registers indicate potentially multiple SC1 registers for use only in hardware trigger mode. Refer to the Chip Configuration information about the number of SC1n registers specific to this device. The SC1n registers have identical fields, and are used in a "ping-pong" approach to control ADC operation.

At any one point in time, only one of the SC1n registers is actively controlling ADC conversions. Updating SC1A while SC1n is actively controlling a conversion is allowed (and vice-versa for any of the SC1n registers specific to this MCU).

Writing SC1A while SC1A is actively controlling a conversion aborts the current conversion. In software trigger mode (ADTRG=0), writes to the SC1A register subsequently initiate a new conversion (if the ADCH bits are equal to a value other than all 1s).

Similarly, writing any of the SC1n registers while that specific SC1n register is actively controlling a conversion aborts the current conversion. None of the SC1B-SC1n registers are used for software trigger operation and therefore writes to the SC1B - SC1n registers do not initiate a new conversion.

Addresses: ADC0_SC1A is 4003_B000h base + 0h offset = 4003_B000h
ADC0_SC1B is 4003_B000h base + 4h offset = 4003_B004h
ADC2_SC1A is 4003_C000h base + 0h offset = 4003_C000h
ADC2_SC1B is 4003_C000h base + 4h offset = 4003_C004h
ADC1_SC1A is 400B_B000h base + 0h offset = 400B_B000h
ADC1_SC1B is 400B_B000h base + 4h offset = 400B_B004h
ADC3_SC1A is 400B_C000h base + 0h offset = 400B_C000h
ADC3_SC1B is 400B_C000h base + 4h offset = 400B_C004h

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								COCO	AIEN	DIFF	ADCH				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

ADCx_SC1*n* field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 COCO	Conversion complete flag The COCO flag is a read-only bit that is set each time a conversion is completed when the compare function is disabled (ACFE=0) and the hardware average function is disabled (AVGE=0). When the compare function is enabled (ACFE=1), the COCO flag is set upon completion of a conversion only if the compare result is true. When the hardware average function is enabled (AVGE=1), the COCO flag is set upon completion of the selected number of conversions (determined by the AVGS bits). The COCO flag in SC1A is also set at the completion of a Calibration sequence. The COCO bit is cleared when the respective SC1n register is written or when the respective Rn register is read. 0 Conversion not completed. 1 Conversion completed.
6 AIEN	Interrupt enable

Table continues on the next page...

ADCx_SC1*n* field descriptions (continued)

Field	Description
	AIEN enables conversion complete interrupts. When COCO becomes set while the respective AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt disabled. 1 Conversion complete interrupt enabled.
5 DIFF	Differential mode enable DIFF configures the ADC to operate in differential mode. When enabled, this mode automatically selects from the differential channels, and changes the conversion algorithm and the number of cycles to complete a conversion. 0 Single-ended conversions and input channels are selected. 1 Differential conversions and input channels are selected.
4–0 ADCH	Input channel select The ADCH bits form a 5-bit field that selects one of the input channels. The input channel decode depends on the value of the DIFF bit. DAD0-DAD3 are associated with the input pin pairs DADPx and DADMx. The successive approximation converter subsystem is turned off when the channel select bits are all set (ADCH = 11111). This feature allows for explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way prevents an additional single conversion from being performed. It is not necessary to set the channel select bits to all ones to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes. 00000 When DIFF=0, DADP0 is selected as input; when DIFF=1, DAD0 is selected as input. 00001 When DIFF=0, DADP1 is selected as input; when DIFF=1, DAD1 is selected as input. 00010 When DIFF=0, DADP2 is selected as input; when DIFF=1, DAD2 is selected as input. 00011 When DIFF=0, DADP3 is selected as input; when DIFF=1, DAD3 is selected as input. 00100 When DIFF=0, AD4 is selected as input; when DIFF=1, it is reserved. 00101 When DIFF=0, AD5 is selected as input; when DIFF=1, it is reserved. 00110 When DIFF=0, AD6 is selected as input; when DIFF=1, it is reserved. 00111 When DIFF=0, AD7 is selected as input; when DIFF=1, it is reserved. 01000 When DIFF=0, AD8 is selected as input; when DIFF=1, it is reserved. 01001 When DIFF=0, AD9 is selected as input; when DIFF=1, it is reserved. 01010 When DIFF=0, AD10 is selected as input; when DIFF=1, it is reserved. 01011 When DIFF=0, AD11 is selected as input; when DIFF=1, it is reserved. 01100 When DIFF=0, AD12 is selected as input; when DIFF=1, it is reserved. 01101 When DIFF=0, AD13 is selected as input; when DIFF=1, it is reserved. 01110 When DIFF=0, AD14 is selected as input; when DIFF=1, it is reserved. 01111 When DIFF=0, AD15 is selected as input; when DIFF=1, it is reserved. 10000 When DIFF=0, AD16 is selected as input; when DIFF=1, it is reserved. 10001 When DIFF=0, AD17 is selected as input; when DIFF=1, it is reserved. 10010 When DIFF=0, AD18 is selected as input; when DIFF=1, it is reserved. 10011 When DIFF=0, AD19 is selected as input; when DIFF=1, it is reserved. 10100 When DIFF=0, AD20 is selected as input; when DIFF=1, it is reserved. 10101 When DIFF=0, AD21 is selected as input; when DIFF=1, it is reserved. 10110 When DIFF=0, AD22 is selected as input; when DIFF=1, it is reserved. 10111 When DIFF=0, AD23 is selected as input; when DIFF=1, it is reserved.

Table continues on the next page...

ADCx_SC1*n* field descriptions (continued)

Field	Description
	11000 Reserved. 11001 Reserved. 11010 When DIFF=0, Temp sensor (single-ended) is selected as input; when DIFF=1, Temp sensor (differential) is selected as input. 11011 When DIFF=0, Bandgap (single-ended) is selected as input; when DIFF=1, Bandgap (differential) is selected as input. 11100 Reserved. 11101 When DIFF=0, V_{REFSH} is selected as input; when DIFF=1, -V_{REFSH} (differential) is selected as input. Voltage reference selected is determined by the REFSEL bits in the SC2 register. 11110 When DIFF=0, V_{REFSL} is selected as input; when DIFF=1, it is reserved. Voltage reference selected is determined by the REFSEL bits in the SC2 register. 11111 Module disabled.

35.3.2 ADC configuration register 1 (ADCx_CFG1)

CFG1 register selects the mode of operation, clock source, clock divide, and configure for low power or long sample time.

Addresses: ADC0_CFG1 is 4003_B000h base + 8h offset = 4003_B008h
ADC2_CFG1 is 4003_C000h base + 8h offset = 4003_C008h
ADC1_CFG1 is 400B_B000h base + 8h offset = 400B_B008h
ADC3_CFG1 is 400B_C000h base + 8h offset = 400B_C008h

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								ADLPC	ADIV		ADLSMP	MODE		ADICLK	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_CFG1 field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 ADLPC	Low-power configuration

Table continues on the next page...

ADCx_CFG1 field descriptions (continued)

Field	Description
	ADLPC controls the power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required. 0 Normal power configuration. 1 Low power configuration. The power is reduced at the expense of maximum clock speed.
6–5 ADIV	Clock divide select ADIV selects the divide ratio used by the ADC to generate the internal clock ADCK. 00 The divide ratio is 1 and the clock rate is input clock. 01 The divide ratio is 2 and the clock rate is (input clock)/2. 10 The divide ratio is 4 and the clock rate is (input clock)/4. 11 The divide ratio is 8 and the clock rate is (input clock)/8.
4 ADLSMP	Sample time configuration ADLSMP selects between different sample times based on the conversion mode selected. This bit adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption if continuous conversions are enabled and high conversion rates are not required. When ADLSMP=1, the long sample time select bits, (ADLSTS[1:0]), can select the extent of the long sample time. 0 Short sample time. 1 Long sample time.
3–2 MODE	Conversion mode selection MODE bits are used to select the ADC resolution mode. 00 When DIFF=0: It is single-ended 8-bit conversion; when DIFF=1, it is differential 9-bit conversion with 2's complement output. 01 When DIFF=0: It is single-ended 12-bit conversion; when DIFF=1, it is differential 13-bit conversion with 2's complement output. 10 When DIFF=0: It is single-ended 10-bit conversion; when DIFF=1, it is differential 11-bit conversion with 2's complement output. 11 When DIFF=0: It is single-ended 16-bit conversion; when DIFF=1, it is differential 16-bit conversion with 2's complement output.
1–0 ADICLK	Input clock select ADICLK bits select the input clock source to generate the internal clock, ADCK. Note that when the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start (ADACKEN=0), the asynchronous clock is activated at the start of a conversion and shuts off when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated. 00 Bus clock. 01 Bus clock divided by 2. 10 Alternate clock (ALTCLK). 11 Asynchronous clock (ADACK).

35.3.3 Configuration register 2 (ADCx_CFG2)

CFG2 register selects the special high speed configuration for very high speed conversions and selects the long sample time duration during long sample mode.

Addresses: ADC0_CFG2 is 4003_B000h base + Ch offset = 4003_B00Ch
ADC2_CFG2 is 4003_C000h base + Ch offset = 4003_C00Ch
ADC1_CFG2 is 400B_B000h base + Ch offset = 400B_B00Ch
ADC3_CFG2 is 400B_C000h base + Ch offset = 400B_C00Ch

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								0			MUXSEL	ADACKEN	ADHSC	ADLSTS	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_CFG2 field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 MUXSEL	ADC Mux select ADC Mux select bit is used to change the ADC mux setting to select between alternate sets of ADC channels. 0 ADxxa channels are selected. 1 ADxxb channels are selected.
3 ADACKEN	Asynchronous clock output enable ADACKEN enables the ADC's asynchronous clock source and the clock source output regardless of the conversion and input clock select (ADICLK bits) status of the ADC. Based on MCU configuration, the asynchronous clock may be used by other modules (see Chip Configuration information). Setting this bit allows the clock to be used even while the ADC is idle or operating from a different clock source. Also, latency of initiating a single or first-continuous conversion with the asynchronous clock selected is reduced since the ADACK clock is already operational.

Table continues on the next page...

ADCx_CFG2 field descriptions (continued)

Field	Description
	0 Asynchronous clock output disabled; Asynchronous clock only enabled if selected by ADICLK and a conversion is active. 1 Asynchronous clock and clock output enabled regardless of the state of the ADC.
2 ADHSC	High speed configuration ADHSC configures the ADC for very high speed operation. The conversion sequence is altered (2 ADCK cycles added to the conversion time) to allow higher speed conversion clocks. 0 Normal conversion sequence selected. 1 High speed conversion sequence selected (2 additional ADCK cycles to total conversion time).
1–0 ADLSTS	Long sample time select ADLSTS selects between the extended sample times when long sample time is selected (ADLSMP=1). This allows higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required. 00 Default longest sample time (20 extra ADCK cycles; 24 ADCK cycles total). 01 12 extra ADCK cycles; 16 ADCK cycles total sample time. 10 6 extra ADCK cycles; 10 ADCK cycles total sample time. 11 2 extra ADCK cycles; 6 ADCK cycles total sample time.

35.3.4 ADC data result register (ADCx_R*n*)

The data result registers (Rn) contain the result of an ADC conversion of the channel selected by the corresponding status and channel control register (SC1A:SC1n). For every status and channel control register, there is a corresponding data result register.

Unused bits in the Rn register are cleared in unsigned right justified modes and carry the sign bit (MSB) in sign extended 2's complement modes. For example, when configured for 10-bit single-ended mode, D[15:10] are cleared. When configured for 11-bit differential mode, D[15:10] carry the sign bit (bit 10 extended through bit 15).

The following table describes the behavior of the data result registers in the different modes of operation.

Table 35-44. Data result register description

Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
16-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Signed 2's complement
16-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right justified

Table continues on the next page...

Table 35-44. Data result register description (continued)

Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
13-bit differential	S	S	S	S	D	D	D	D	D	D	D	D	D	D	D	D	Sign extended 2's complement
12-bit single-ended	0	0	0	0	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right justified
11-bit differential	S	S	S	S	S	S	D	D	D	D	D	D	D	D	D	D	Sign extended 2's complement
10-bit single-ended	0	0	0	0	0	0	D	D	D	D	D	D	D	D	D	D	Unsigned right justified
9-bit differential	S	S	S	S	S	S	S	S	D	D	D	D	D	D	D	D	Sign extended 2's complement
8-bit single-ended	0	0	0	0	0	0	0	0	D	D	D	D	D	D	D	D	Unsigned right justified

NOTE

S: Sign bit or sign bit extension;

D: Data (2's complement data if indicated)

Addresses: ADC0_RA is 4003_B000h base + 10h offset = 4003_B010h
ADC0_RB is 4003_B000h base + 14h offset = 4003_B014h
ADC2_RA is 4003_C000h base + 10h offset = 4003_C010h
ADC2_RB is 4003_C000h base + 14h offset = 4003_C014h
ADC1_RA is 400B_B000h base + 10h offset = 400B_B010h
ADC1_RB is 400B_B000h base + 14h offset = 400B_B014h
ADC3_RA is 400B_C000h base + 10h offset = 400B_C010h
ADC3_RB is 400B_C000h base + 14h offset = 400B_C014h

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="16">0</td><td colspan="16">D</td></tr>
<tr><td>W</td><td colspan="16"></td><td colspan="16"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

ADCx_R*n* field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 D	Data result

35.3.5 Compare value registers (ADCx_CV*n*)

The compare value registers (CV1 and CV2) contain a compare value used to compare with the conversion result when the compare function is enabled (ACFE=1). This register is formatted the same for both bit position definition and value format (unsigned or sign-extended 2's complement) as the data result registers (Rn) in the different modes of operation. Therefore, the compare function only uses the compare value register bits that are related to the ADC mode of operation.

The compare value 2 register (CV2) is utilized only when the compare range function is enabled (ACREN=1).

Addresses: ADC0_CV1 is 4003_B000h base + 18h offset = 4003_B018h
ADC0_CV2 is 4003_B000h base + 1Ch offset = 4003_B01Ch
ADC2_CV1 is 4003_C000h base + 18h offset = 4003_C018h
ADC2_CV2 is 4003_C000h base + 1Ch offset = 4003_C01Ch
ADC1_CV1 is 400B_B000h base + 18h offset = 400B_B018h
ADC1_CV2 is 400B_B000h base + 1Ch offset = 400B_B01Ch
ADC3_CV1 is 400B_C000h base + 18h offset = 400B_C018h
ADC3_CV2 is 400B_C000h base + 1Ch offset = 400B_C01Ch

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																CV															
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

ADCx_CVn field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 CV	Compare value

35.3.6 Status and control register 2 (ADCx_SC2)

The SC2 register contains the conversion active, hardware/software trigger select, compare function and voltage reference select of the ADC module.

Addresses: ADC0_SC2 is 4003_B000h base + 20h offset = 4003_B020h
ADC2_SC2 is 4003_C000h base + 20h offset = 4003_C020h
ADC1_SC2 is 400B_B000h base + 20h offset = 400B_B020h
ADC3_SC2 is 400B_C000h base + 20h offset = 400B_C020h

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								ADACT	ADTRG	ACFE	ACFGT	ACREN	DMAEN	REFSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_SC2 field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 ADACT	Conversion active ADACT indicates that a conversion or hardware averaging is in progress. ADACT is set when a conversion is initiated and cleared when a conversion is completed or aborted. 0 Conversion not in progress. 1 Conversion in progress.
6 ADTRG	Conversion trigger select ADTRG selects the type of trigger used for initiating a conversion. Two types of trigger are selectable: software trigger and hardware trigger. When software trigger is selected, a conversion is initiated following a write to SC1A. When hardware trigger is selected, a conversion is initiated following the assertion of the ADHWT input after a pulse of the ADHWTSn input. 0 Software trigger selected. 1 Hardware trigger selected.

Table continues on the next page...

ADCx_SC2 field descriptions (continued)

Field	Description
5 ACFE	Compare function enable ACFE enables the compare function. 0 Compare function disabled. 1 Compare function enabled.
4 ACFGT	Compare function greater than enable ACFGT configures the compare function to check the conversion result relative to the compare value register(s) (CV1 and CV2) based upon the value of ACREN. The ACFE bit must be set for ACFGT to have any effect. 0 Configures less than threshold, outside range not inclusive and inside range not inclusive functionality based on the values placed in the CV1 and CV2 registers. 1 Configures greater than or equal to threshold, outside range inclusive and inside range inclusive functionality based on the values placed in the CV1 and CV2 registers.
3 ACREN	Compare function range enable ACREN configures the compare function to check if the conversion result of the input being monitored is either between or outside the range formed by the compare value registers (CV1 and CV2) determined by the value of ACFGT. The ACFE bit must be set for ACFGT to have any effect. 0 Range function disabled. Only the compare value 1 register (CV1) is compared. 1 Range function enabled. Both compare value registers (CV1 and CV2) are compared.
2 DMAEN	DMA enable 0 DMA is disabled. 1 DMA is enabled and will assert the ADC DMA request during a ADC conversion complete event noted by the assertion of any of the ADC COCO flags.
1–0 REFSEL	Voltage reference selection REFSEL bits select the voltage reference source used for conversions. 00 Default voltage reference pin pair (external pins V_{REFH} and V_{REFL}) 01 Alternate reference pair (V_{ALTH} and V_{ALTL}). This pair may be additional external pins or internal sources depending on MCU configuration. Consult the Chip Configuration information for details specific to this MCU. 10 Reserved 11 Reserved

35.3.7 Status and control register 3 (ADCx_SC3)

The SC3 register controls the calibration, continuous convert, and hardware averaging functions of the ADC module.

Addresses: ADC0_SC3 is 4003_B000h base + 24h offset = 4003_B024h
ADC2_SC3 is 4003_C000h base + 24h offset = 4003_C024h
ADC1_SC3 is 400B_B000h base + 24h offset = 400B_B024h
ADC3_SC3 is 400B_C000h base + 24h offset = 400B_C024h

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								CAL	CALF	0		ADCO	AVGE	AVGS	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_SC3 field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 CAL	Calibration CAL begins the calibration sequence when set. This bit stays set while the calibration is in progress and is cleared when the calibration sequence is completed. The CALF bit must be checked to determine the result of the calibration sequence. Once started, the calibration routine cannot be interrupted by writes to the ADC registers or the results will be invalid and the CALF bit will set. Setting the CAL bit will abort any current conversion.
6 CALF	Calibration failed flag CALF displays the result of the calibration sequence. The calibration sequence will fail if ADTRG = 1, any ADC register is written, or any stop mode is entered before the calibration sequence completes. The CALF bit is cleared by writing a 1 to this bit. 0 Calibration completed normally. 1 Calibration failed. ADC accuracy specifications are not guaranteed.
5–4 Reserved	This read-only field is reserved and always has the value zero.
3 ADCO	Continuous conversion enable ADCO enables continuous conversions.

Table continues on the next page...

ADCx_SC3 field descriptions (continued)

Field	Description
	0 One conversion or one set of conversions if the hardware average function is enabled (AVGE=1) after initiating a conversion. 1 Continuous conversions or sets of conversions if the hardware average function is enabled (AVGE=1) after initiating a conversion.
2 AVGE	Hardware average enable AVGE enables the hardware average function of the ADC. 0 Hardware average function disabled. 1 Hardware average function enabled.
1–0 AVGS	Hardware average select AVGS determines how many ADC conversions will be averaged to create the ADC average result. 00 4 samples averaged. 01 8 samples averaged. 10 16 samples averaged. 11 32 samples averaged.

35.3.8 ADC offset correction register (ADCx_OFS)

The ADC offset correction register (OFS) contains the user selected or calibration generated offset error correction value. This register is a 2's complement, left justified, 16-bit value. The value in the offset correction registers (OFS) is subtracted from the conversion and the result is transferred into the result registers (Rn). If the result is above the maximum or below the minimum result value, it is forced to the appropriate limit for the current mode of operation.

Addresses: ADC0_OFS is 4003_B000h base + 28h offset = 4003_B028h
ADC2_OFS is 4003_C000h base + 28h offset = 4003_C028h
ADC1_OFS is 400B_B000h base + 28h offset = 400B_B028h
ADC3_OFS is 400B_C000h base + 28h offset = 400B_C028h

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																OFS															
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	1	0	0

ADCx_OFS field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 OFS	Offset error correction value

35.3.9 ADC plus-side gain register (ADCx_PG)

The plus-side gain register (PG) contains the gain error correction for the plus-side input in differential mode or the overall conversion in single-ended mode. PG, a 16-bit real number in binary format, is the gain adjustment factor, with the radix point fixed between ADPG15 and ADPG14. This register must be written by the user with the value described in the calibration procedure or the gain error specifications may not be met.

Addresses: ADC0_PG is 4003_B000h base + 2Ch offset = 4003_B02Ch
ADC2_PG is 4003_C000h base + 2Ch offset = 4003_C02Ch
ADC1_PG is 400B_B000h base + 2Ch offset = 400B_B02Ch
ADC3_PG is 400B_C000h base + 2Ch offset = 400B_C02Ch

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																PG															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

ADCx_PG field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 PG	Plus-side gain

35.3.10 ADC minus-side gain register (ADCx_MG)

The minus-side gain register (MG) contains the gain error correction for the minus-side input in differential mode. This register is ignored in single-ended mode. MG, a 16-bit real number in binary format, is the gain adjustment factor, with the radix point fixed between ADMG15 and ADMG14. This register must be written by the user with the value described in the calibration procedure or the gain error specifications may not be met.

Addresses: ADC0_MG is 4003_B000h base + 30h offset = 4003_B030h
ADC2_MG is 4003_C000h base + 30h offset = 4003_C030h
ADC1_MG is 400B_B000h base + 30h offset = 400B_B030h
ADC3_MG is 400B_C000h base + 30h offset = 400B_C030h

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																MG															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

ADCx_MG field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 MG	Minus-side gain

35.3.11 ADC plus-side general calibration value register (ADCx_CLPD)

The plus-side general calibration value registers (CLPx) contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLP0[5:0], CLP1[6:0], CLP2[7:0], CLP3[8:0], CLP4[9:0], CLPS[5:0], and CLPD[5:0]. CLPx are automatically set once the self calibration sequence is done (CAL is cleared). If these registers are written by the user after calibration, the linearity error specifications may not be met.

Addresses: ADC0_CLPD is 4003_B000h base + 34h offset = 4003_B034h
ADC2_CLPD is 4003_C000h base + 34h offset = 4003_C034h
ADC1_CLPD is 400B_B000h base + 34h offset = 400B_B034h
ADC3_CLPD is 400B_C000h base + 34h offset = 400B_C034h

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																										CLPD					
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	1	0	1	0

ADCx_CLPD field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

ADCx_CLPD field descriptions (continued)

Field	Description
5–0 CLPD	Calibration value

35.3.12 ADC plus-side general calibration value register (ADCx_CLPS)

For more information, refer to CLPD register description.

Addresses: ADC0_CLPS is 4003_B000h base + 38h offset = 4003_B038h
ADC2_CLPS is 4003_C000h base + 38h offset = 4003_C038h
ADC1_CLPS is 400B_B000h base + 38h offset = 400B_B038h
ADC3_CLPS is 400B_C000h base + 38h offset = 400B_C038h

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																										CLPS					
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	0	0	0	0	0

ADCx_CLPS field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5–0 CLPS	Calibration value

35.3.13 ADC plus-side general calibration value register (ADCx_CLP4)

For more information, refer to CLPD register description.

Addresses: ADC0_CLP4 is 4003_B000h base + 3Ch offset = 4003_B03Ch
ADC2_CLP4 is 4003_C000h base + 3Ch offset = 4003_C03Ch
ADC1_CLP4 is 400B_B000h base + 3Ch offset = 400B_B03Ch
ADC3_CLP4 is 400B_C000h base + 3Ch offset = 400B_C03Ch

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																						CLP4									
W																																
Reset	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	0	0	0	0	0	0	0

ADCx_CLP4 field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value zero.
9–0 CLP4	Calibration value

35.3.14 ADC plus-side general calibration value register (ADCx_CLP3)

For more information, refer to CLPD register description.

Addresses: ADC0_CLP3 is 4003_B000h base + 40h offset = 4003_B040h
ADC2_CLP3 is 4003_C000h base + 40h offset = 4003_C040h
ADC1_CLP3 is 400B_B000h base + 40h offset = 400B_B040h
ADC3_CLP3 is 400B_C000h base + 40h offset = 400B_C040h

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																							CLP3								
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	1	0	0	0	0	0	0	0	0

ADCx_CLP3 field descriptions

Field	Description
31–9 Reserved	This read-only field is reserved and always has the value zero.
8–0 CLP3	Calibration value

35.3.15 ADC plus-side general calibration value register (ADCx_CLP2)

For more information, refer to CLPD register description.

Addresses: ADC0_CLP2 is 4003_B000h base + 44h offset = 4003_B044h
ADC2_CLP2 is 4003_C000h base + 44h offset = 4003_C044h
ADC1_CLP2 is 400B_B000h base + 44h offset = 400B_B044h
ADC3_CLP2 is 400B_C000h base + 44h offset = 400B_C044h

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																								CLP2							
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	1	0	0	0	0	0	0	0

ADCx_CLP2 field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 CLP2	Calibration value

35.3.16 ADC plus-side general calibration value register (ADCx_CLP1)

For more information, refer to CLPD register description.

Addresses: ADC0_CLP1 is 4003_B000h base + 48h offset = 4003_B048h
ADC2_CLP1 is 4003_C000h base + 48h offset = 4003_C048h
ADC1_CLP1 is 400B_B000h base + 48h offset = 400B_B048h
ADC3_CLP1 is 400B_C000h base + 48h offset = 400B_C048h

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																									CLP1						
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	1	0	0	0	0	0	0

ADCx_CLP1 field descriptions

Field	Description
31–7 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

ADCx_CLP1 field descriptions (continued)

Field	Description
6–0 CLP1	Calibration value

35.3.17 ADC plus-side general calibration value register (ADCx_CLP0)

For more information, refer to CLPD register description.

Addresses: ADC0_CLP0 is 4003_B000h base + 4Ch offset = 4003_B04Ch
ADC2_CLP0 is 4003_C000h base + 4Ch offset = 4003_C04Ch
ADC1_CLP0 is 400B_B000h base + 4Ch offset = 400B_B04Ch
ADC3_CLP0 is 400B_C000h base + 4Ch offset = 400B_C04Ch

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																										CLP0					
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	0	0	0	0	0

ADCx_CLP0 field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5–0 CLP0	Calibration value

35.3.18 ADC PGA register (ADCx_PGA)

Addresses: ADC0_PGA is 4003_B000h base + 50h offset = 4003_B050h
ADC2_PGA is 4003_C000h base + 50h offset = 4003_C050h
ADC1_PGA is 400B_B000h base + 50h offset = 400B_B050h
ADC3_PGA is 400B_C000h base + 50h offset = 400B_C050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								PGAEN	0	PGACHPb	PGALPb	PGAG			
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	PGAOFSM	0													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_PGA field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value zero.
23 PGAEN	PGA enable 0 PGA disabled. 1 PGA enabled.
22 Reserved	This read-only field is reserved and always has the value zero.
21 PGACHPb	PGA chopping control 0 Chopping enabled. 1 Chopping disabled.
20 PGALPb	PGA low-power mode control 0 PGA runs in low power mode. 1 PGA runs in normal power mode.
19–16 PGAG	PGA gain setting PGA gain = 2^(PGAG) 0000 1 0001 2 0010 4 0011 8

Table continues on the next page...

ADCx_PGA field descriptions (continued)

Field	Description
	0100 16 0101 32 0110 64 0111 Reserved 1000 Reserved 1001 Reserved 1010 Reserved 1011 Reserved 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved
15 Reserved	This read-only field is reserved and always has the value zero.
14 PGAOFSM	PGA Offset Measurement When this bit is set, the PGA disconnects itself from the external inputs and auto-configures into offset measurement mode. With this bit set, simply run the ADC in recommended settings and enable maximum hardware averaging to get the PGA offset number. The output will be (PGA offset * (64+1)) for the given PGA setting. 0 PGA runs in normal operation. 1 PGA runs in offset measurement mode.
13–0 Reserved	This read-only field is reserved and always has the value zero.

35.3.19 ADC minus-side general calibration value register (ADCx_CLMD)

CLMx contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLM0[5:0], CLM1[6:0], CLM2[7:0], CLM3[8:0], CLM4[9:0], CLMS[5:0], and CLMD[5:0]. CLMx are automatically set once the self calibration sequence is done (CAL is cleared). If these registers are written by the user after calibration, the linearity error specifications may not be met.

Addresses: ADC0_CLMD is 4003_B000h base + 54h offset = 4003_B054h
ADC2_CLMD is 4003_C000h base + 54h offset = 4003_C054h
ADC1_CLMD is 400B_B000h base + 54h offset = 400B_B054h
ADC3_CLMD is 400B_C000h base + 54h offset = 400B_C054h

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																										CLMD					
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	1	0	1	0

ADCx_CLMD field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5–0 CLMD	Calibration value

35.3.20 ADC minus-side general calibration value register (ADCx_CLMS)

For more information, refer to CLMD register description.

Addresses: ADC0_CLMS is 4003_B000h base + 58h offset = 4003_B058h
ADC2_CLMS is 4003_C000h base + 58h offset = 4003_C058h
ADC1_CLMS is 400B_B000h base + 58h offset = 400B_B058h
ADC3_CLMS is 400B_C000h base + 58h offset = 400B_C058h

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																										CLMS					
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	0	0	0	0	0

ADCx_CLMS field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5–0 CLMS	Calibration value

35.3.21 ADC minus-side general calibration value register (ADCx_CLM4)

For more information, refer to CLMD register description.

Addresses: ADC0_CLM4 is 4003_B000h base + 5Ch offset = 4003_B05Ch
ADC2_CLM4 is 4003_C000h base + 5Ch offset = 4003_C05Ch
ADC1_CLM4 is 400B_B000h base + 5Ch offset = 400B_B05Ch
ADC3_CLM4 is 400B_C000h base + 5Ch offset = 400B_C05Ch

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																						CLM4									
W																																
Reset	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	0	0	0	0	0	0	0

ADCx_CLM4 field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value zero.
9–0 CLM4	Calibration value

35.3.22 ADC minus-side general calibration value register (ADCx_CLM3)

For more information, refer to CLMD register description.

Addresses: ADC0_CLM3 is 4003_B000h base + 60h offset = 4003_B060h
ADC2_CLM3 is 4003_C000h base + 60h offset = 4003_C060h
ADC1_CLM3 is 400B_B000h base + 60h offset = 400B_B060h
ADC3_CLM3 is 400B_C000h base + 60h offset = 400B_C060h

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																							CLM3								
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	1	0	0	0	0	0	0	0	0

ADCx_CLM3 field descriptions

Field	Description
31–9 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

ADCx_CLM3 field descriptions (continued)

Field	Description
8–0 CLM3	Calibration value

35.3.23 ADC minus-side general calibration value register (ADCx_CLM2)

For more information, refer to CLMD register description.

Addresses: ADC0_CLM2 is 4003_B000h base + 64h offset = 4003_B064h
ADC2_CLM2 is 4003_C000h base + 64h offset = 4003_C064h
ADC1_CLM2 is 400B_B000h base + 64h offset = 400B_B064h
ADC3_CLM2 is 400B_C000h base + 64h offset = 400B_C064h

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																								CLM2							
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	1	0	0	0	0	0	0	0

ADCx_CLM2 field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 CLM2	Calibration value

35.3.24 ADC minus-side general calibration value register (ADCx_CLM1)

For more information, refer to CLMD register description.

Addresses: ADC0_CLM1 is 4003_B000h base + 68h offset = 4003_B068h
ADC2_CLM1 is 4003_C000h base + 68h offset = 4003_C068h
ADC1_CLM1 is 400B_B000h base + 68h offset = 400B_B068h
ADC3_CLM1 is 400B_C000h base + 68h offset = 400B_C068h

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																									CLM1						
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	1	0	0	0	0	0	0

ADCx_CLM1 field descriptions

Field	Description
31–7 Reserved	This read-only field is reserved and always has the value zero.
6–0 CLM1	Calibration value

35.3.25 ADC minus-side general calibration value register (ADCx_CLM0)

For more information, refer to CLMD register description.

Addresses: ADC0_CLM0 is 4003_B000h base + 6Ch offset = 4003_B06Ch
ADC2_CLM0 is 4003_C000h base + 6Ch offset = 4003_C06Ch
ADC1_CLM0 is 400B_B000h base + 6Ch offset = 400B_B06Ch
ADC3_CLM0 is 400B_C000h base + 6Ch offset = 400B_C06Ch

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																										CLM0					
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	0	0	0	0	0

ADCx_CLM0 field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5–0 CLM0	Calibration value

35.4 Functional description

The ADC module is disabled during reset, in low power stop mode (refer to the Power Management information for details), or when the ADCH bits in SC1n are all high. The module is idle when a conversion has completed and another conversion has not been initiated. When it is idle and the asynchronous clock output enable is disabled (ADACKEN is 0), the module is in its lowest power state. The ADC can perform an analog-to-digital conversion on any of the software selectable channels. All modes perform conversion by a successive approximation algorithm.

To meet accuracy specifications, the ADC module must be calibrated using the on chip calibration function. See Calibration function for details on how to perform calibration.

When the conversion is completed, the result is placed in the data registers (Rn). The respective conversion complete flag (COCO) is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled (AIEN=1).

The ADC module has the capability of automatically comparing the result of a conversion with the contents of the compare value registers. The compare function is enabled by setting the ACFE bit and operates with any of the conversion modes and configurations.

The ADC module has the capability of automatically averaging the result of multiple conversions. The hardware average function is enabled by setting the AVGE bit and operates with any of the conversion modes and configurations.

NOTE

> For the chip specific modes of operation, refer to the Power Management information of this MCU.

35.4.1 PGA functional description

The Programmable Gain Amplifier (PGA) is designed to increase the dynamic range by amplifying low-amplitude signals before they are fed to the 16-bit SAR ADC. The gain of this amplifier is ranged between 1 to 64 in (2^N) steps (1,2,4,8,16,32,64).

This block is designed to work with differential input and output with input signals that range from 0 -1.2 V ± 10 mV. The output common mode of the PGA is determined based on the SAR ADC requirement.

The PGA has only one voltage reference pair. The positive reference used is chip specific and depends on the MCU configuration. Refer to the Chip Configuration chapter on the PGA Voltage Reference specific to this MCU. The ground reference is the analog ground for the PGA.

The ADC PGA register allows to control the PGA gain and modes of operation.

The PGA employs chopping to remove/reduce offset and 1/f noise and offers an offset measurement configuration that aids offset calibration. These two are user programmable features and the details are given below.

PGA Chopping:

The PGA chopping is controlled by PGACHPb bit in PGA register. The default value is “0” (enabled). User can disable the chopping by writing “1” to this register bit. Due to the operation of the PGA chopping, to achieve the specified PGA accuracy, the user must operate the ADC with hardware averaging enabled or must average an even number of samples in software. These samples must be gathered by the ADC with no ADC conversion on a different channel or at a different ADC configuration between the samples. A time delta between the samples controlled by software or hardware trigger (PDB) is allowed. If the PGA is used without averaging or if single conversion is required, chopping should be disabled. While chopping removes the maximum offset and 1/f noise, it mandates averaging to be done. For cases requiring higher effective sampling rate (therefore no averaging) but lower accuracy, the chopping may be disabled and offset calibration may be employed instead. In PGA mode, it is recommended to run the ADC with maximum possible sample rate to get maximum benefit of chopping.

Measurement of PGA offset and offset calibration in software:

The offset of the PGA can be measured by setting PGAOFSM bit in PGA register. When this bit is set, the PGA disconnects itself from the external inputs and auto-configures into offset measurement mode at x64 gain. With this bit set, run the ADC in recommended settings and enable maximum hardware averaging to get the PGA offset number. The output will be (PGA offset * (64+1)) for the given PGA setting. If chopping is enabled during offset measurement, this measures the residual PGA offset after chopping. If chopping is disabled during offset measurement, this measures the non-chopped PGA offset. PGA offset can be calibrated in software with the help of the above measured value. Store the measured offset (X) in memory for the desired setting. Subtract [(X/(64+1))*(G+1)] from the result, where G is the PGA gain during operation.

Digital LPF to improve SNR/ENOB:

For low-frequency and high-accuracy applications, on top of averaging, user may digitally filter the ADC output to cut out-of-band noise to get improvement in SNR/ ENOB from what is specified in device data sheet.

35.4.2 Clock select and divide control

One of four clock sources can be selected as the clock source for the ADC module. This clock source is then divided by a configurable value to generate the input clock to the converter (ADCK). The clock is selected from one of the following sources by means of the ADICLK bits.

- The bus clock. This is the default selection following reset.
- The bus clock divided by two. For higher bus clock rates, this allows a maximum divide by 16 of the bus clock with using the ADIV bits.

- ALTCLK, as defined for this MCU. Refer to the Chip Configuration information.
- The asynchronous clock (ADACK). This clock is generated from a clock source within the ADC module. Note that when the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start (ADACKEN=0), the asynchronous clock is activated at the start of a conversion and shuts off when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated. To avoid the conversion time variability and latency associated with the ADACK clock startup, set ADACKEN=1 and wait the worst case startup time of 5 µs prior to initiating any conversions using the ADACK clock source. Conversions are possible using ADACK as the input clock source while the MCU is in Normal Stop mode. Refer to Power Control for more information.

Whichever clock is selected, its frequency must fall within the specified frequency range for ADCK. If the available clocks are too slow, the ADC may not perform according to specifications. If the available clocks are too fast, the clock must be divided to the appropriate frequency. This divider is specified by the ADIV bits and can be divide-by 1, 2, 4, or 8.

35.4.3 Voltage reference selection

The ADC can be configured to accept one of the two voltage reference pairs as the reference voltage (V_{REFSH} and V_{REFSL}) used for conversions. Each pair contains a positive reference that must be between the minimum Ref Voltage High and V_{DDA}, and a ground reference that must be at the same potential as V_{SSA}. The two pairs are external (V_{REFH} and V_{REFL}) and alternate (V_{ALTH} and V_{ALTL}). These voltage references are selected using the REFSEL bits in the SC2 register. The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration. Refer to the Chip Configuration information on the Voltage References specific to this MCU.

35.4.4 Hardware trigger and channel selects

The ADC module has a selectable asynchronous hardware conversion trigger, ADHWT, that is enabled when the ADTRG bit is set and a hardware trigger select event (ADHWTSn) has occurred. This source is not available on all MCUs. Refer to the Chip Configuration chapter for information on the ADHWT source and the ADHWTSn configurations specific to this MCU.

When a ADHWT source is available and hardware trigger is enabled (ADTRG=1), a conversion is initiated on the rising edge of ADHWT after a hardware trigger select event (ADHWTSn) has occurred. If a conversion is in progress when a rising edge of a trigger occurs, the rising edge is ignored. In continuous convert configuration, only the initial rising edge to launch continuous conversions is observed, and until conversion gets aborted the ADC continues to do conversions on the same ADC status and control register that initiated the conversion. The hardware trigger function operates in conjunction with any of the conversion modes and configurations.

The hardware trigger select event (ADHWTSn) must be set prior to the receipt of the ADHWT signal. If these conditions are not met, the converter may ignore the trigger or use the incorrect configuration. If a hardware trigger select event gets asserted during a conversion, it must stay asserted until the end of current conversion and remain set until the receipt of the ADHWT signal to trigger a new conversion. The channel and status fields selected for the conversion depend on the active trigger select signal (ADHWTSA active selects SC1A; ADHWTSn active selects SC1n).

Note

> Asserting more than one hardware trigger select signal (ADHWTSn) at the same time results in unknown results. To avoid this, select only one hardware trigger select signal (ADHWTSn) prior to the next intended conversion.

When the conversion is completed, the result is placed in the data registers associated with the ADHWTSn received (ADHWTSA active selects RA register; ADHWTSn active selects Rn register). The conversion complete flag associated with the ADHWTSn received (the COCO bit in SC1n register) is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled (AIEN=1).

35.4.5 Conversion control

Conversions can be performed as determined by the CFG1[MODE] bits and the SC1n[DIFF] bit as shown in the description of CFG1[MODE].

Conversions can be initiated by a software or hardware trigger. In addition, the ADC module can be configured for low power operation, long sample time, continuous conversion, hardware average, and automatic compare of the conversion result to a software determined compare value.

35.4.5.1 Initiating conversions

A conversion is initiated:

- Following a write to SC1A register (with ADCH bits not all 1's) if software triggered operation is selected (ADTRG=0).
- Following a hardware trigger (ADHWT) event if hardware triggered operation is selected (ADTRG=1) and a hardware trigger select event (ADHWTSn) has occurred. The channel and status fields selected depend on the active trigger select signal (ADHWTSA active selects SC1A register; ADHWTSn active selects SC1n register; if neither is active, the off condition is selected).

Note

> Selecting more than one hardware trigger select signal (ADHWTSn) prior to a conversion completion will result in unknown results. To avoid this, select only one hardware trigger select signal (ADHWTSn) prior to a conversion completion.

- Following the transfer of the result to the data registers when continuous conversion is enabled (ADCO=1).

If continuous conversions are enabled, a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation (ADTRG=0), continuous conversions begin after SC1A register is written and continue until aborted. In hardware triggered operation (ADTRG=1 and one ADHWTSn event has occurred), continuous conversions begin after a hardware trigger event and continue until aborted.

If hardware averaging is enabled, a new conversion is automatically initiated after the completion of the current conversion until the correct number of conversions is completed. In software triggered operation, conversions begin after SC1A register is written. In hardware triggered operation, conversions begin after a hardware trigger. If continuous conversions are also enabled, a new set of conversions to be averaged are initiated following the last of the selected number of conversions.

35.4.5.2 Completing conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, Rn. If the compare functions are disabled, this is indicated by the setting of the COCO bit in the respective SC1n register. If hardware averaging is enabled, the respective COCO bit sets only if the last of the selected number of conversions is completed. If the compare function is enabled, the respective COCO bit sets and conversion result data is transferred only if the compare condition is true. If both hardware averaging and compare functions are enabled then the respective COCO bit sets only if the last of the selected number of conversions is completed and the compare condition is true. An interrupt is generated if the respective AIEN bit is high at the time that the respective COCO bit is set.

35.4.5.3 Aborting conversions

Any conversion in progress is aborted when:

- Writing to SC1A register while it is actively controlling a conversion, aborts the current conversion. In software trigger mode (ADTRG=0), a write to SC1A register initiates a new conversion (if the ADCH field in SC1A is equal to a value other than all 1s). Writing to any of the SC1(B-n) registers while that specific SC1(B-n) register is actively controlling a conversion aborts the current conversion.The SC1(B-n) registers are not used for software trigger operation and therefore writes to the SC1(B-n) registers do not initiate a new conversion.
- A write to any ADC register besides the SC1A:SC1n registers occurs. This indicates a mode of operation change has occurred and the current conversion is therefore invalid.
- The MCU is reset or enters Low Power Stop modes.
- The MCU enters Normal Stop mode with ADACK not enabled.

When a conversion is aborted, the contents of the data registers, Rn, are not altered. The data registers continue to be the values transferred after the completion of the last successful conversion. If the conversion was aborted by a reset or Low Power Stop modes, RA and R n return to their reset states.

35.4.5.4 Power control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source, but the asynchronous clock output is disabled (ADACKEN=0), the ADACK clock generator also remains in its idle state (disabled) until a conversion is initiated. If the asynchronous clock output is enabled (ADACKEN=1), it remains active regardless of the state of the ADC or the MCU power mode.

Power consumption when the ADC is active can be reduced by setting ADLPC. This results in a lower maximum value for f_{ADCK} .

35.4.5.5 Sample time and total conversion time

For short sample (ADLSMP=0), there is a 2-cycle adder for first conversion over the base sample time of 4 ADCK cycles. For high speed conversions (ADHSC=1), there is an additional 2-cycle adder on any conversion. The table below summarizes sample times for the possible ADC configurations.

ADC Configuration			Sample time (ADCK cycles)	
ADLSMP	ADLSTS	ADHSC	First or Single	Subsequent
0	X	0	6	4
1	00	0	24	
1	01	0	16	
1	10	0	10	
1	11	0	6	
0	X	1	8	6
1	00	1	26	
1	01	1	18	
1	10	1	12	
1	11	1	8	

The total conversion time depends upon: the sample time (as determined by ADLSMP and ADLSTS bits), the MCU bus frequency, the conversion mode (as determined by MODE and SC1n[DIFF] bits), the high speed configuration (ADHSC bit), and the frequency of the conversion clock (f_{ADCK}).

The ADHSC bit is used to configure a higher clock input frequency. This will allow faster overall conversion times. To meet internal ADC timing requirements, the ADHSC bit adds additional ADCK cycles. Conversions with ADHSC = 1 take two more ADCK cycles. ADHSC should be used when the ADCLK exceeds the limit for ADHSC = 0.

After the module becomes active, sampling of the input begins. ADLSMP and ADLSTS select between sample times based on the conversion mode that is selected. When sampling is completed, the converter is isolated from the input channel and a successive approximation algorithm is performed to determine the digital value of the analog signal. The result of the conversion is transferred to Rn upon completion of the conversion algorithm.

If the bus frequency is less than the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled (ADLSMP=0).

The maximum total conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is selectable by the ADICLK bits, and the divide ratio is specified by the ADIV bits.

The maximum total conversion time for all configurations is summarized in the equation below. Refer to the following tables for the variables referenced in the equation.

$$\mathrm{ConversionTime} = \mathrm{SFCAdder} + \mathrm{AverageNum} \times \left(\mathrm{BCT} + \mathrm{LSTAdder} + \mathrm{HSCAdder}\right)$$

Figure 35-157. Conversion time equation

Table 35-177. Single or first continuous time adder (SFCAdder)

ADLSMP	ADACKEN	ADICLK	Single or first continuous time adder (SFCAdder)
1	x	0x, 10	3 ADCK cycles + 5 bus clock cycles
1	1	11	3 ADCK cycles + 5 bus clock cycles[1]
1	0	11	5 μs + 3 ADCK cycles + 5 bus clock cycles
0	x	0x, 10	5 ADCK cycles + 5 bus clock cycles
0	1	11	5 ADCK cycles + 5 bus clock cycles[1]
0	0	11	5 μs + 5 ADCK cycles + 5 bus clock cycles

[1] To achieve this time, ADACKEN must be 1 for at least 5 μs prior to the conversion is initiated.

Table 35-178. Average number factor (AverageNum)

AVGE	AVGS[1:0]	Average number factor (AverageNum)
0	xx	1
1	00	4
1	01	8
1	10	16
1	11	32

Table 35-179. Base Conversion Time (BCT)

Mode	Base conversion time (BCT)
8b s.e.	17 ADCK cycles
9b diff	27 ADCK cycles
10b s.e.	20 ADCK cycles
11b diff	30 ADCK cycles
12b s.e.	20 ADCK cycles

Table continues on the next page...

Table 35-179. Base Conversion Time (BCT) (continued)

Mode	Base conversion time (BCT)
13b diff	30 ADCK cycles
16b s.e.	25 ADCK cycles
16b diff	34 ADCK cycles

Table 35-180. Long sample time adder (LSTAdder)

ADLSMP	ADLSTS	Long sample time adder (LSTAdder)
0	xx	0 ADCK cycles
1	00	20 ADCK cycles
1	01	12 ADCK cycles
1	10	6 ADCK cycles
1	11	2 ADCK cycles

Table 35-181. High Speed Conversion time Adder (HSCAdder)

ADHSC	High Speed Conversion Time Adder (HSCAdder)
0	0 ADCK cycles
1	2 ADCK cycles

Note

The ADCK frequency must be between f_{ADCK} minimum and f_{ADCK} maximum to meet ADC specifications.

35.4.5.6 Conversion time examples

The following examples use Figure 35-157 and the information provided in Table 35-177 through Table 35-181.

35.4.5.6.1 Typical conversion time configuration

A typical configuration for ADC conversion is: 10-bit mode, with the bus clock selected as the input clock source, the input clock divide-by-1 ratio selected, and a bus frequency of 8 MHz, long sample time disabled and high speed conversion disabled. The conversion time for a single conversion is calculated by using Figure 35-157 and the information provided in Table 35-177 through Table 35-181. The table below list the variables of Figure 35-157.

Table 35-182. Typical conversion time

Variable	Time
SFCAdder	5 ADCK cycles + 5 bus clock cycles
AverageNum	1
BCT	20 ADCK cycles
LSTAdder	0
HSCAdder	0

The resulting conversion time is generated using the parameters listed in the proceeding table. Therefore, for a bus clock equal to 8 MHz and an ADCK equal to 8 MHz the resulting conversion time is 3.75 µs.

35.4.5.6.2 Long conversion time configuration

A configuration for long ADC conversion is: 16-bit differential mode with the bus clock selected as the input clock source, the input clock divide-by-8 ratio selected, a bus frequency of 8 MHz, long sample time enabled, configured for longest adder, high speed conversion disabled, and average enabled for 32 conversions. The conversion time for this conversion is calculated by using Figure 35-157 and the information provided in Table 35-177 through Table 35-181. The following table lists the variables of the Figure 35-157.

Table 35-183. Typical conversion time

Variable	Time
SFCAdder	3 ADCK cycles + 5 bus clock cycles
AverageNum	32
BCT	34 ADCK cycles
LSTAdder	20 ADCK cycles
HSCAdder	0

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for bus clock equal to 8 MHz and ADCK equal to 1 MHz, the resulting conversion time is 57.625 µs (AverageNum). This results in a total conversion time of 1.844 ms.

35.4.5.6.3 Short conversion time configuration

A configuration for short ADC conversion is: 8-bit single ended mode with the bus clock selected as the input clock source, the input clock divide-by-1 ratio selected, a bus frequency of 20 MHz, long sample time disabled, and high speed conversion enabled.

The conversion time for this conversion is calculated by using Figure 35-157 and the information provided in Table 35-177 through Table 35-181. The table below list the variables of Figure 35-157.

Table 35-184. Typical conversion time

Variable	Time
SFCAdder	5 ADCK cycles + 5 bus clock cycles
AverageNum	1
BCT	17 ADCK cycles
LSTAdder	0 ADCK cycles
HSCAdder	2

The resulting conversion time is generated using the parameters listed in in the preceding table. Therefore, for bus clock equal to 20 MHz and ADCK equal to 20 MHz, the resulting conversion time is 1.45 µs.

35.4.5.7 Hardware average function

The hardware average function can be enabled (AVGE=1) to perform a hardware average of multiple conversions. The number of conversions is determined by the AVGS[1:0] bits, which select 4, 8, 16, or 32 conversions to be averaged. While the hardware average function is in progress, the ADACT bit will be set.

After the selected input is sampled and converted, the result is placed in an accumulator from which an average is calculated once the selected number of conversions has been completed. When hardware averaging is selected, the completion of a single conversion will not set the COCO bit.

If the compare function is either disabled or evaluates true, after the selected number of conversions are completed, the average conversion result is transferred into the data result registers, Rn, and the COCO bit is set. An ADC interrupt is generated upon the setting of COCO if the respective ADC interrupt is enabled (AIEN=1).

Note

> The hardware average function can perform conversions on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the hardware average is completed if SC1n[AIEN] bit was set.

35.4.6 Automatic compare function

The compare function can be configured to check if the result is less than or greater-than-or-equal-to a single compare value, or if the result falls within or outside a range determined by two compare values. The compare mode is determined by ACFGT, ACREN, and the values in the compare value registers (CV1 and CV2). After the input is sampled and converted, the compare values (CV1 and CV2) are used as described in the following table.There are six compare modes as shown in the following table.

Table 35-185. Compare modes

ACFGT	ACREN	ADCCV1 relative to ADCCV2	Function	Compare mode description
0	0	—	Less than threshold	Compare true if the result is less than the CV1 registers.
1	0	—	Greater than or equal to threshold	Compare true if the result is greater than or equal to CV1 registers.
0	1	Less than or equal	Outside range, not inclusive	Compare true if the result is less than CV1 **Or** the result is greater than CV2.
0	1	Greater than	Inside range, not inclusive	Compare true if the result is less than CV1 **And** the result is greater than CV2.
1	1	Less than or equal	Inside range, inclusive	Compare true if the result is greater than or equal to CV1 **And** the result is less than or equal to CV2.
1	1	Greater than	Outside range, inclusive	Compare true if the result is greater than or equal to CV1 **Or** the result is less than or equal to CV2.

With the ADC range enable bit set, ACREN =1, and if compare value register 1 (CV1 value) is less than or equal to the compare value register 2 (CV2 value), then setting ACFGT will select a trigger-if-inside-compare-range inclusive-of-endpoints function. Clearing ACFGT will select a trigger-if-outside-compare-range, not-inclusive-of-endpoints function.

If CV1 is greater than CV2, setting ACFGT will select a trigger-if-outside-compare-range, inclusive-of-endpoints function. Clearing ACFGT will select a trigger-if-inside-compare-range, not-inclusive-of-endpoints function.

If the condition selected evaluates true, COCO is set.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, COCO is not set and the conversion result data will not be transferred to the result register. If the hardware averaging function is enabled, the compare function compares the averaged result to the compare values. The same compare function definitions apply. An ADC interrupt is generated upon the setting of COCO if the respective ADC interrupt is enabled (AIEN=1).

Note

The compare function can monitor the voltage on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the compare condition is met.

35.4.7 Calibration function

The ADC contains a self-calibration function that is required to achieve the specified accuracy. Calibration must be run, or valid calibration values written, after any reset and before a conversion is initiated. The calibration function sets the offset calibration value, the minus-side calibration values, and the plus-side calibration values. The offset calibration value is automatically stored in the ADC offset correction register (OFS), and the plus-side and minus-side calibration values are automatically stored in the ADC plus-side and minus-side calibration (CLPx and CLMx) registers. The user must configure the ADC correctly prior to calibration, and must generate the plus-side and minus-side gain calibration results and store them in the ADC plus-side gain register (PG) after the calibration function completes.

Prior to calibration, the user must configure the ADC's clock source and frequency, low power configuration, voltage reference selection, sample time, and high speed configuration according to the application's clock source availability and needs. If the application uses the ADC in a wide variety of configurations, the configuration for which the highest accuracy is required should be selected, or multiple calibrations can be done for the different configurations. For best calibration results, it is recommended to set hardware averaging to maximum (AVGE=1, AVGS=11 for average of 32), ADC clock frequency f_{ADCK} less than or equal to 4 MHz, V_{REFH}=V_{DDA}, and to calibrate at nominal voltage and temperature. The input channel, conversion mode continuous function, compare function, resolution mode, and differential/single-ended mode are all ignored during the calibration function.

To initiate calibration, the user sets the CAL bit and the calibration will automatically begin if the ADTRG bit is 0. If ADTRG is 1, the CAL bit will not get set and the calibration fail flag (CALF) will be set. While calibration is active, no ADC register can be written and no stop mode may be entered, or the calibration routine will be aborted causing the CAL bit to clear and the CALF bit to set. At the end of a calibration sequence, the COCO bit of the SC1A register will be set. The AIEN bit can be used to allow an interrupt to occur at the end of a calibration sequence. At the end of the calibration routine, if the CALF bit is not set, the automatic calibration routine completed successfully.

To complete calibration, the user must generate the gain calibration values using the following procedure:

1. Initialize (clear) a 16-bit variable in RAM.
2. Add the plus-side calibration results CLP0, CLP1, CLP2, CLP3, CLP4, and CLPS to the variable.
3. Divide the variable by two.
4. Set the MSB of the variable.
5. The previous two steps can be achieved by setting the carry bit, rotating to the right through the carry bit on the high byte and again on the low byte.
6. Store the value in the plus-side gain calibration register (PG).
7. Repeat the procedure for the minus-side gain calibration value.

When calibration is complete, the user may reconfigure and use the ADC as desired. A second calibration may also be performed if desired by clearing and again setting the CAL bit.

Overall, the calibration routine may take as many as 14k ADCK cycles and 100 bus cycles, depending on the results and the clock source chosen. For an 8 MHz clock source, this length amounts to about 1.7 ms. To reduce this latency, the calibration values (offset, plus-side and minus-side gain, and plus-side and minus-side calibration values) may be stored in flash memory after an initial calibration and recovered prior to the first ADC conversion. This method should reduce the calibration latency to 20 register store operations on all subsequent power, reset, or Low Power Stop mode recoveries.

35.4.8 User defined offset function

The ADC offset correction register (OFS) contains the user selected or calibration generated offset error correction value. This register is a 2’s complement, left justified. The value in the offset correction register (OFS) is subtracted from the conversion and the result is transferred into the result registers (Rn). If the result is above the maximum or below the minimum result value, it is forced to the appropriate limit for the current mode of operation.

The formatting of the ADC offset correction register is different from the data result register (Rn) to preserve the resolution of the calibration value regardless of the conversion mode selected. Lower order bits are ignored in lower resolution modes. For example, in 8-bit single-ended mode, the bits OFS[14:7] are subtracted from D[7:0]; bit OFS[15] indicates the sign (negative numbers are effectively added to the result) and bits

OFS[6:0] are ignored. The same bits are used in 9-bit differential mode since bit OFS[15] indicates the sign bit, which maps to bit D[8]. For 16-bit differential mode, all bits OFS[15:0] are directly subtracted from the conversion result data D[15:0]. In 16-bit single-ended mode, there is no bit in the offset correction register corresponding to the least significant result bit D[0], so odd values (-1 or +1, and so on) cannot be subtracted from the result.

OFS is automatically set according to calibration requirements once the self calibration sequence is done (CAL is cleared). The user may write to OFS to override the calibration result if desired. If the offset correction register is written by the user to a value that is different from the calibration value, the ADC error specifications may not be met. It is recommended that the value generated by the calibration function be stored in memory before overwriting with a user specified value.

Note

> There is an effective limit to the values of offset that can be set by the user. If the magnitude of the offset is too great, the results of the conversions will cap off at the limits.

The offset calibration function may be employed by the user to remove application offsets or DC bias values. The offset correction register, OFS may be written with a number in 2's complement format and this offset will be subtracted from the result (or hardware averaged value). To add an offset, store the negative offset in 2's complement format and the effect will be an addition. An offset correction that results in an out-of-range value will be forced to the minimum or maximum value (the minimum value for single-ended conversions is 0x0000; for a differential conversion it is 0x8000).

To preserve accuracy, the calibrated offset value initially stored in the OFS register must be added to the user defined offset. For applications that may change the offset repeatedly during operation, it is recommended to store the initial offset calibration value in flash so it can be recovered and added to any user offset adjustment value and the sum stored in the OFS register.

35.4.9 Temperature sensor

The ADC module includes a temperature sensor whose output is connected to one of the ADC analog channel inputs. The following equation provides an approximate transfer function of the temperature sensor.

$$\mathrm{Temp} = 25 - \left(\left(V_{\mathrm{TEMP}} - V_{\mathrm{TEMP25}} \right) \div m \right)$$

Figure 35-158. Approximate transfer function of the temperature sensor

where:

- V_{TEMP} is the voltage of the temperature sensor channel at the ambient temperature.
- V_{TEMP25} is the voltage of the temperature sensor channel at 25 °C.
- m is the hot or cold voltage versus temperature slope in V/°C.

For temperature calculations, use the V_{TEMP25} and m values from the ADC Electricals table.

In application code, the user reads the temperature sensor channel, calculates V_{TEMP}, and compares to V_{TEMP25}. If V_{TEMP} is greater than V_{TEMP25} the cold slope value is applied in the preceding equation. If V_{TEMP} is less than V_{TEMP25}, the hot slope value is applied in the preceding equation.

For more information on using the temperature sensor, see the application note titled *Temperature Sensor for the HCS08 Microcontroller Family* (document AN3031).

35.4.10 MCU wait mode operation

Wait mode is a lower power-consumption standby mode from which recovery is fast because the clock sources remain active. If a conversion is in progress when the MCU enters Wait mode, it continues until completion. Conversions can be initiated while the MCU is in Wait mode by means of the hardware trigger or if continuous conversions are enabled.

The bus clock, bus clock divided by two, and ADACK are available as conversion clock sources while in Wait mode. The use of ALTCLK as the conversion clock source in Wait is dependent on the definition of ALTCLK for this MCU. Refer to the Chip Configuration information on ALTCLK specific to this MCU.

If the compare and hardware averaging functions are disabled, a conversion complete event sets the COCO and generates an ADC interrupt to wake the MCU from Wait mode if the respective ADC interrupt is enabled (AIEN=1). If the hardware averaging function is enabled, the COCO will set (and generate an interrupt if enabled) when the selected number of conversions are completed. If the compare function is enabled, the COCO will set (and generate an interrupt if enabled) only if the compare conditions are met. If a single conversion is selected and the compare trigger is not met, the ADC will return to its idle state and cannot wake the MCU from Wait mode unless a new conversion is initiated by the hardware trigger.

35.4.11 MCU Normal Stop mode operation

Stop mode is a low power-consumption standby mode during which most or all clock sources on the MCU are disabled.

35.4.11.1 Normal Stop mode with ADACK disabled

If the asynchronous clock, ADACK, is not selected as the conversion clock, executing a stop instruction aborts the current conversion and places the ADC in its idle state. The contents of the ADC registers, including Rn, are unaffected by Normal Stop mode. After exiting from Normal Stop mode, a software or hardware trigger is required to resume conversions.

35.4.11.2 Normal Stop mode with ADACK enabled

If ADACK is selected as the conversion clock, the ADC continues operation during Normal Stop mode. Refer to the Chip Configuration chapter for configuration information for this MCU.

If a conversion is in progress when the MCU enters Normal Stop mode, it continues until completion. Conversions can be initiated while the MCU is in Normal Stop mode by means of the hardware trigger or if continuous conversions are enabled.

If the compare and hardware averaging functions are disabled, a conversion complete event sets the COCO and generates an ADC interrupt to wake the MCU from Normal Stop mode if the respective ADC interrupt is enabled (AIEN = 1). The result register will contain the data from the first completed conversion that occurred during Normal Stop mode. If the hardware averaging function is enabled, the COCO will set (and generate an interrupt if enabled) when the selected number of conversions are completed. If the compare function is enabled, the COCO will set (and generate an interrupt if enabled) only if the compare conditions are met. If a single conversion is selected and the compare is not true, the ADC will return to its idle state and cannot wake the MCU from Normal Stop mode unless a new conversion is initiated by another hardware trigger.

35.4.12 MCU Low Power Stop mode operation

The ADC module is automatically disabled when the MCU enters Low Power Stop mode. All module registers contain their reset values following exit from Low Power Stop mode. Therefore, the module must be re-enabled and re-configured following exit from Low Power Stop mode.

NOTE

For the chip specific modes of operation, refer to the Power Management information for the device.

35.5 Initialization information

This section gives an example that provides some basic direction on how to initialize and configure the ADC module. You can configure the module for 16-bit, 12-bit, 10-bit, or 8-bit single-ended resolution or 16-bit, 13-bit, 11-bit, or 9-bit differential resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. Refer to Table 35-180, Table 35-181, and Table 35-182 for information used in this example.

Note

Hexadecimal values are designated by a preceding 0x, binary values designated by a preceding %, and decimal values have no preceding character.

35.5.1 ADC module initialization example

This section provides details about the ADC module initialization.

35.5.1.1 Initialization sequence

Before the ADC module can be used to complete conversions, an initialization procedure must be performed. A typical sequence is as follows:

1. Calibrate the ADC by following the calibration instructions in Calibration function.
2. Update the configuration register (CFG) to select the input clock source and the divide ratio used to generate the internal clock, ADCK. This register is also used for selecting sample time and low-power configuration.
3. Update status and control register 2 (SC2) to select the conversion trigger (hardware or software) and compare function options, if enabled.
4. Update status and control register 3 (SC3) to select whether conversions will be continuous or completed only once (ADCO) and to select whether to perform hardware averaging.

5. Update the status and control register (SC1:SC1n) to select whether conversions will be single-ended or differential and to enable or disable conversion complete interrupts. Also, select the input channel on which to perform conversions.
6. Update PGA register (PGA) to enable or disable PGA and configure appropriate gain. This register is also used for selecting power mode and whether the module is chopper stabilized.

35.5.1.2 Pseudo-code example

In this example, the ADC module is set up with interrupts enabled to perform a single 10-bit conversion at low power with a long sample time on input channel 1, where the internal ADCK clock is derived from the bus clock divided by 1.

CFG1 = 0x98 (%10011000)

```
        Bit 7   ADLPC   1      Configures for low power (lowers maximum clock speed.
        Bit 6:5 ADIV    00     Sets the ADCK to the input clock ÷ 1.
        Bit 4   ADLSMP  1      Configures for long sample time.
        Bit 3:2   MODE   10    Selects the single-ended 10-bit conversion, differential 11-
bit conversion.
        Bit 1:0   ADICLK   00    Selects the bus clock.
```

SC2 = 0x00 (%00000000)

```
        Bit 7   ADACT   0       Flag indicates if a conversion is in progress.
        Bit 6   ADTRG   0       Software trigger selected.
        Bit 5   ACFE    0       Compare function disabled.
        Bit 4   ACFGT   0       Not used in this example.
        Bit 3   ACREN   0      Compare range disabled.
        Bit 2   DMAEN   0      DMA request disabled.
        Bit 1:0 REFSEL  00     Selects default voltage reference pin pair (External pins VREFH
and VREFL).
```

SC1A = 0x41 (%01000001)

```
        Bit 7   COCO   0      Read-only flag which is set when a conversion completes.
        Bit 6   AIEN   1      Conversion complete interrupt enabled.
        Bit 5   DIFF   0      Single-ended conversion selected.
        Bit 4:0 ADCH   00001  Input channel 1 selected as ADC input channel.
```

RA = 0xxx

```
        Holds results of conversion.
```

CV = 0xxx

```
        Holds compare value when compare function enabled.
```

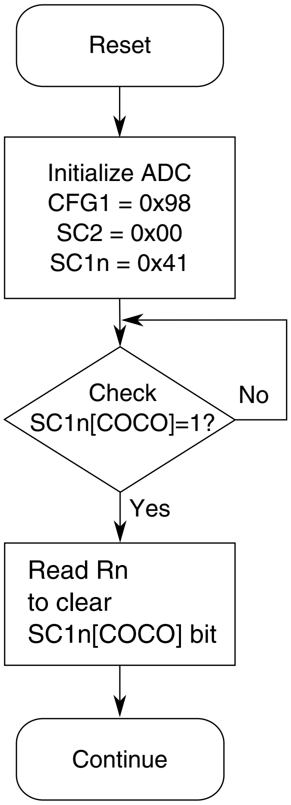


Figure 35-159. Initialization Flowchart for Example

35.6 Application information

This section contains information for using the ADC module in applications. The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an ADC.

35.6.1 External pins and routing

The following sections discuss the external pins associated with the ADC module and how they should be used for best results.

35.6.1.1 Analog supply pins

The ADC module has analog power and ground supplies (V_{DDA} and V_{SSA}) available as separate pins on some devices. V_{SSA} is shared on the same pin as the MCU digital VSS on some devices. On other devices, V_{SSA} and V_{DDA} are shared with the MCU digital supply pins. In these cases, there are separate pads for the analog supplies bonded to the same pin as the corresponding digital supply so that some degree of isolation between the supplies is maintained.

When available on a separate pin, both V_{DDA} and V_{SSA} must be connected to the same voltage potential as their corresponding MCU digital supply (V_{DD} and V_{SS}) and must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

If separate power supplies are used for analog and digital power, the ground connection between these supplies must be at the V_{SSA} pin. This should be the only ground connection between these supplies if possible. The V_{SSA} pin makes a good single point ground location.

35.6.1.2 Analog voltage reference pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs used by the converter, V_{REFSH} and V_{REFSL}. V_{REFSH} is the high reference voltage for the converter. V_{REFSL} is the low reference voltage for the converter.

The ADC can be configured to accept one of two voltage reference pairs for V_{REFSH} and V_{REFSL}. Each pair contains a positive reference and a ground reference. The two pairs are external (V_{REFH} and V_{REFL}) and alternate (V_{ALTH} and V_{ALTL}). These voltage references are selected using the REFSEL bits in the SC2 register. The alternate (V_{ALTH} and V_{ALTL}) voltage reference pair may select additional external pins or internal sources depending on MCU configuration. Refer to the Chip Configuration information on the Voltage References specific to this MCU.

In some packages, the external or alternate pairs are connected in the package to V_{DDA} and V_{SSA}, respectively. One of these positive references may be shared on the same pin as V_{DDA} on some devices. One of these ground references may be shared on the same pin as V_{SSA} on some devices.

If externally available, the positive reference may be connected to the same potential as V_{DDA} or may be driven by an external source to a level between the minimum Ref Voltage High and the V_{DDA} potential (the positive reference must never exceed V_{DDA}). If externally available, the ground reference must be connected to the same voltage potential as V_{SSA}. The voltage reference pairs must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

AC current in the form of current spikes required to supply charge to the capacitor array at each successive approximation step is drawn through the V_{REFH} and V_{REFL} loop. The best external component to meet this current demand is a 0.1 μF capacitor with good high frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as near as possible to the package pins. Resistance in the path is not recommended because the current causes a voltage drop that could result in conversion errors. Inductance in this path must be minimum (parasitic only).

35.6.1.3 Analog input pins

The external analog inputs are typically shared with digital I/O pins on MCU devices.

Empirical data shows that capacitors on the analog inputs improve performance in the presence of noise or when the source impedance is high. Use of 0.01 μF capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used they must be placed as near as possible to the package pins and be referenced to V_{SSA}.

For proper conversion, the input voltage must fall between V_{REFH} and V_{REFL}. If the input is equal to or exceeds V_{REFH}, the converter circuit converts the signal to 0xFFF (full scale 12-bit representation), 0x3FF (full scale 10-bit representation) or 0xFF (full scale 8-bit representation). If the input is equal to or less than V_{REFL}, the converter circuit converts it to 0x000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. There is a brief current associated with V_{REFL} when the sampling capacitor is charging.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins should not be transitioning during conversions.

35.6.2 Sources of error

Several sources of error exist for A/D conversions. These are discussed in the following sections.

35.6.2.1 Sampling error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy.

$$\text{RAS} + \text{RADIN} = \text{SC} / (\text{FMAX} * \text{NUMTAU} * \text{CADIN})$$

Figure 35-160. Sampling equation

Where:

RAS = External analog source resistance

SC = Number of ADCK cycles used during sample window

CADIN = Internal ADC input capacitance

NUMTAU = -ln(LSBERR / 2^N)

LSBERR = value of acceptable sampling error in LSBs

N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode or 16 in 16-bit mode

Higher source resistances or higher-accuracy sampling is possible by setting ADLSMP and changing the ADLSTS bits (to increase the sample window) or decreasing ADCK frequency to increase sample time.

35.6.2.2 Pin leakage error

Leakage on the I/O pins can cause conversion error if the external analog source resistance (R_{AS}) is high. If this error cannot be tolerated by the application, keep R_{AS} lower than V_{REFH} / (4 × I_{LEAK} × 2^N) for less than 1/4 LSB leakage error (N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode, or 16 in 16-bit mode).

35.6.2.3 Noise-induced errors

System noise that occurs during the sample or conversion process can affect the accuracy of the conversion. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1 μF low-ESR capacitor from V_{REFH} to V_{REFL}.
- There is a 0.1 μF low-ESR capacitor from V_{DDA} to V_{SSA}.
- If inductive isolation is used from the primary supply, an additional 1 μF capacitor is placed from V_{DDA} to V_{SSA}.
- V_{SSA} (and V_{REFL}, if connected) is connected to V_{SS} at a quiet point in the ground plane.
- Operate the MCU in Wait or Normal Stop mode before initiating (hardware triggered conversions) or immediately after initiating (hardware or software triggered conversions) the ADC conversion.

- For software triggered conversions, immediately follow the write to the SC1 register with a wait instruction or stop instruction.
- For Normal Stop mode operation, select ADACK as the clock source. Operation in Normal Stop reduces V_{DD} noise but increases effective conversion time due to stop recovery.

- There is no I/O switching, input or output, on the MCU during the conversion.

There are some situations where external system activity causes radiated or conducted noise emissions or excessive V_{DD} noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in Wait or Normal Stop or I/O activity cannot be halted, these recommended actions may reduce the effect of noise on the accuracy:

- Place a 0.01 μF capacitor (C_{AS}) on the selected input channel to V_{REFL} or V_{SSA} (this improves noise issues, but affects the sample rate based on the external analog source resistance).
- Average the result by converting the analog input many times in succession and dividing the sum of the results. Four samples are required to eliminate the effect of a 1 LSB, one-time error.
- Reduce the effect of synchronous noise by operating off the asynchronous clock (ADACK) and averaging. Noise that is synchronous to ADCK cannot be averaged out.

35.6.2.4 Code width and quantization error

The ADC quantizes the ideal straight-line transfer function into 65536 steps (in 16-bit mode). Each step ideally has the same height (1 code) and width. The width is defined as the delta between the transition points to one code and the next. The ideal code width for an N bit converter (in this case N can be 16, 12, 10, or 8), defined as 1 LSB, is:

$$1\text{LSB}=\left(V_{REFH}\right)/2^{N}$$

Figure 35-161. Ideal code width for an N bit converter

There is an inherent quantization error due to the digitization of the result. For 8-bit, 10-bit, or 12-bit conversions, the code transitions when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be ± 1/2 LSB in 8-bit, 10-bit, or 12-bit modes. As a consequence, however, the code width of the first (0x000) conversion is only 1/2 LSB and the code width of the last (0xFF or 0x3FF) is 1.5 LSB.

For 16-bit conversions, the code transitions only after the full code width is present, so the quantization error is -1 LSB to 0 LSB and the code width of each step is 1 LSB.

35.6.2.5 Linearity errors

The ADC may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors, but the system designers should be aware of them because they affect overall accuracy. These errors are:

- Zero-scale error (E_{ZS}) (sometimes called offset): This error is defined as the difference between the actual code width of the first conversion and the ideal code width (1/2 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode). If the first conversion is 0x001, the difference between the actual 0x001 code width and its ideal (1 LSB) is used.
- Full-scale error (E_{FS}): This error is defined as the difference between the actual code width of the last conversion and the ideal code width (1.5 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode). If the last conversion is 0x3FE, the difference between the actual 0x3FE code width and its ideal (1 LSB) is used.
- Differential non-linearity (DNL): This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral non-linearity (INL): This error is defined as the highest-value the (absolute value of the) running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total unadjusted error (TUE): This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function and includes all forms of error.

35.6.2.6 Code jitter, non-monotonicity, and missing codes

Analog-to-digital converters are susceptible to three special forms of error. These are code jitter, non-monotonicity, and missing codes.

Code jitter is when, at certain points, a given input voltage converts to one of two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the converter yields the lower code (and vice-versa). However, even small amounts of system noise can cause the converter to be indeterminate (between two codes) for a range of input voltages around the transition voltage.

This error may be reduced by repeatedly sampling the input and averaging the result. Additionally, the techniques discussed in Noise-induced errors reduces this error.

Non-monotonicity is defined as when, except for code jitter, the converter converts to a lower code for a higher input voltage. Missing codes are those values never converted for any input value.

In 8-bit or 10-bit mode, the ADC is guaranteed to be monotonic and have no missing codes.

Chapter 36
Comparator (CMP)

36.1 Introduction

NOTE

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

The Comparator module (CMP) 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 Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal provided by the 6-bit DAC. The mux circuit is designed to operate across the full range of the supply voltage.

The 6-bit DAC is 64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed. The 64-tap resistor ladder network divides the supply reference Vin into 64 voltage level. A 6-bit digital signal input selects output voltage level, which varies from Vin to Vin/64. Vin can be selected from two voltage sources, V_{in1} and V_{in2}. The 6-bit DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

36.2 CMP Features

The CMP has the following features:

- Operates 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 comparator output
- Selectable inversion on comparator output
- Comparator output may be:
    - Sampled
    - Windowed (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
- Support DMA transfer
    - A comparison event can be selected to trigger a DMA transfer.
- Functional in all modes of operation.
- The window and filter functions are not available in Stop, VLPS, LLS and VLLSx modes.

36.3 6-bit DAC Key Features

- 6-bit resolution
- Selectable supply reference source
- Power down mode to conserve power when it is not being used
- Output can be routed to internal comparator input

36.4 ANMUX Key Features

- Two 8 to 1 channel mux
- Operates the entire supply range

36.5 CMP, DAC, and ANMUX Diagram

The following figure shows the block diagram for the High Speed Comparator, Digital to Analog Converter, and Analog MUX modules.

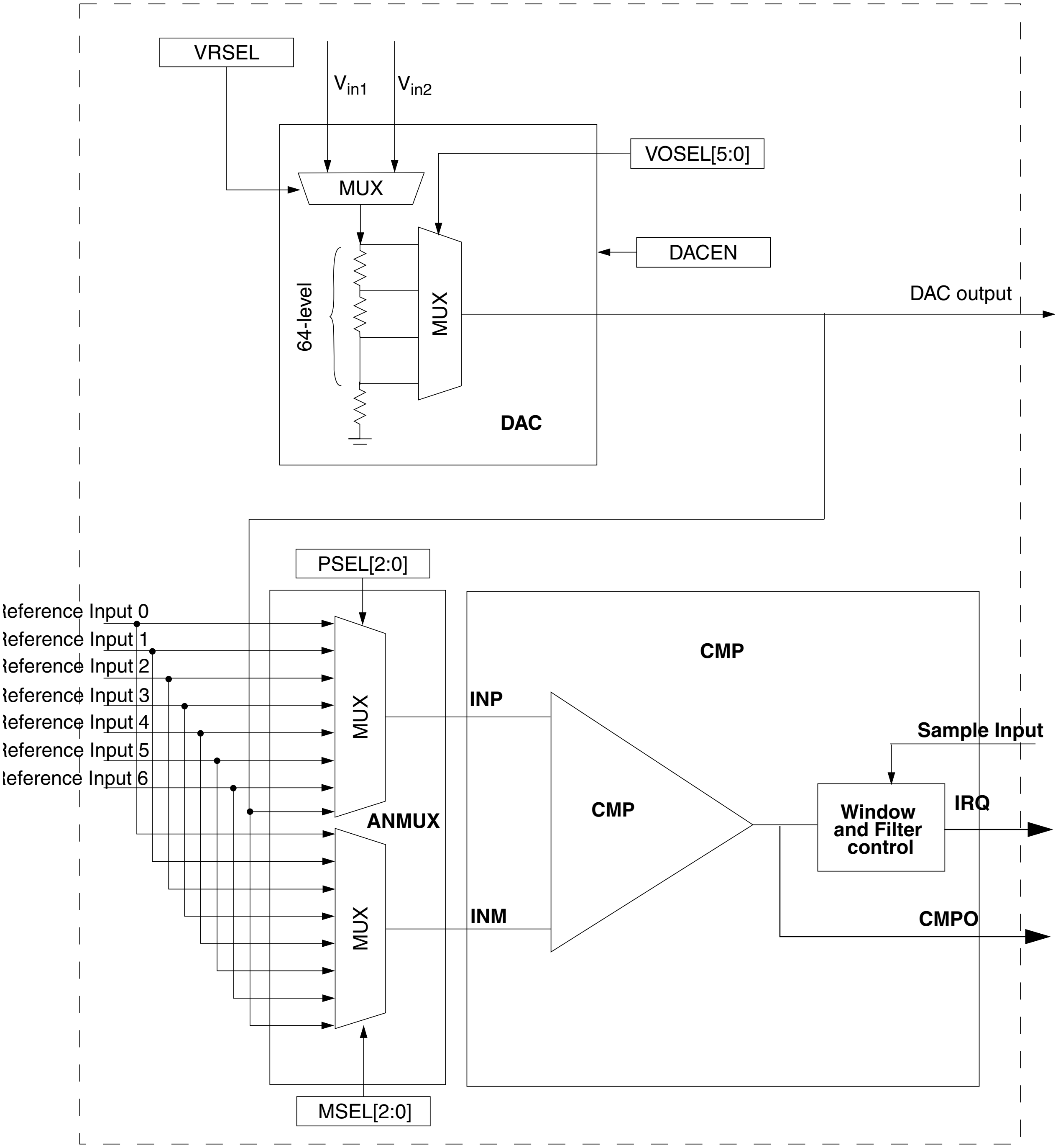


Figure 36-1. CMP, DAC and ANMUX Blocks Diagram

36.6 CMP Block Diagram

The following figure shows the block diagram for the Comparator module.

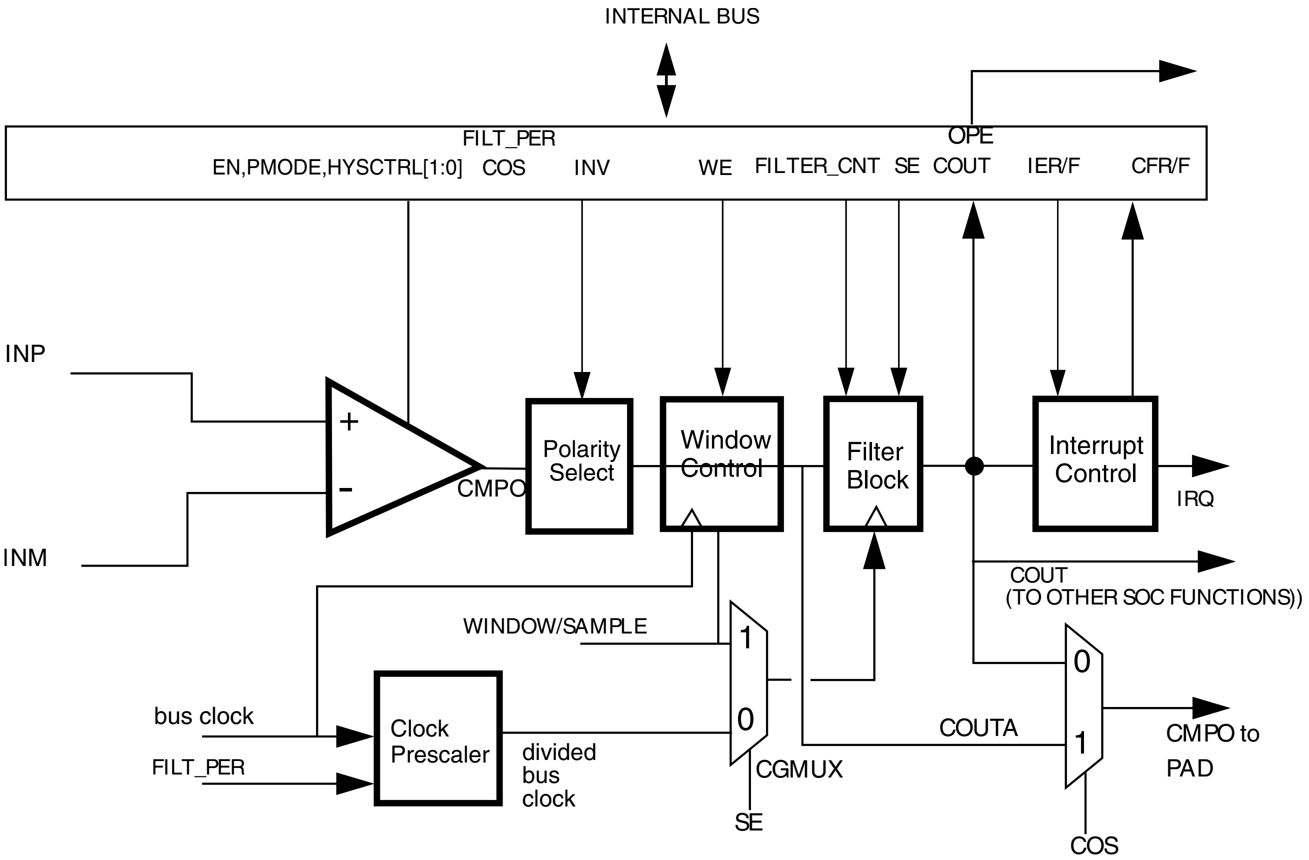


Figure 36-2. Comparator Module Block Diagram

In the CMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- If CR1[WE] = 1, the comparator output will be 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.
- The Filter Block acts as a simple sampler if the filter is bypassed and CR0[FILTER_CNT] is set to 0x01.
- The Filter Block filters based on multiple samples when the filter is bypassed and CR0[FILTER_CNT] is set greater than 0x01.
    - If CR1[SE] = 1, the external SAMPLE input is used as sampling clock
    - IF CR1[SE] = 0, the divided bus clock is used as sampling clock

- If enabled, the Filter Block will incur up to 1 bus clock additional latency penalty on COUT due to the fact that COUT (which is crossing clock domain boundaries) must be resynchronized to the bus clock.
- CR1[WE] and CR1[SE] are mutually exclusive.

36.7 Memory Map/Register Definitions

CMP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_3000	CMP Control Register 0 (CMP0_CR0)	8	R/W	00h	36.7.1/ 975
4007_3001	CMP Control Register 1 (CMP0_CR1)	8	R/W	00h	36.7.2/ 976
4007_3002	CMP Filter Period Register (CMP0_FPR)	8	R/W	00h	36.7.3/ 978
4007_3003	CMP Status and Control Register (CMP0_SCR)	8	R/W	00h	36.7.4/ 978
4007_3004	DAC Control Register (CMP0_DACCR)	8	R/W	00h	36.7.5/ 980
4007_3005	MUX Control Register (CMP0_MUXCR)	8	R/W	00h	36.7.6/ 980
4007_3008	CMP Control Register 0 (CMP1_CR0)	8	R/W	00h	36.7.1/ 975
4007_3009	CMP Control Register 1 (CMP1_CR1)	8	R/W	00h	36.7.2/ 976
4007_300A	CMP Filter Period Register (CMP1_FPR)	8	R/W	00h	36.7.3/ 978
4007_300B	CMP Status and Control Register (CMP1_SCR)	8	R/W	00h	36.7.4/ 978
4007_300C	DAC Control Register (CMP1_DACCR)	8	R/W	00h	36.7.5/ 980
4007_300D	MUX Control Register (CMP1_MUXCR)	8	R/W	00h	36.7.6/ 980
4007_3010	CMP Control Register 0 (CMP2_CR0)	8	R/W	00h	36.7.1/ 975
4007_3011	CMP Control Register 1 (CMP2_CR1)	8	R/W	00h	36.7.2/ 976

Table continues on the next page...

CMP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_3012	CMP Filter Period Register (CMP2_FPR)	8	R/W	00h	36.7.3/ 978
4007_3013	CMP Status and Control Register (CMP2_SCR)	8	R/W	00h	36.7.4/ 978
4007_3014	DAC Control Register (CMP2_DACCR)	8	R/W	00h	36.7.5/ 980
4007_3015	MUX Control Register (CMP2_MUXCR)	8	R/W	00h	36.7.6/ 980
4007_3018	CMP Control Register 0 (CMP3_CR0)	8	R/W	00h	36.7.1/ 975
4007_3019	CMP Control Register 1 (CMP3_CR1)	8	R/W	00h	36.7.2/ 976
4007_301A	CMP Filter Period Register (CMP3_FPR)	8	R/W	00h	36.7.3/ 978
4007_301B	CMP Status and Control Register (CMP3_SCR)	8	R/W	00h	36.7.4/ 978
4007_301C	DAC Control Register (CMP3_DACCR)	8	R/W	00h	36.7.5/ 980
4007_301D	MUX Control Register (CMP3_MUXCR)	8	R/W	00h	36.7.6/ 980

36.7.1 CMP Control Register 0 (CMPx_CR0)

Addresses: CMP0_CR0 is 4007_3000h base + 0h offset = 4007_3000h
CMP1_CR0 is 4007_3008h base + 0h offset = 4007_3008h
CMP2_CR0 is 4007_3010h base + 0h offset = 4007_3010h
CMP3_CR0 is 4007_3018h base + 0h offset = 4007_3018h

Bit	7	6	5	4	3	2	1	0
Read	0	FILTER_CNT			0	0	HYSTCTR	
Write								
Reset	0	0	0	0	0	0	0	0

CMPx_CR0 field descriptions

Field	Description
7 Reserved	This read-only field is reserved and always has the value zero.
6–4 FILTER_CNT	Filter Sample Count

Table continues on the next page...

CMPx_CR0 field descriptions (continued)

Field	Description
	These bits represent 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 reference the Functional Description. 000 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. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3 Reserved	This read-only field is reserved and always has the value zero.
2 Reserved	This read-only field is reserved and always has the value zero.
1–0 HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level is device-specific. See the device's data sheet for the exact values. 00 Level 0 01 Level 1 10 Level 2 11 Level 3

36.7.2 CMP Control Register 1 (CMPx_CR1)

Addresses: CMP0_CR1 is 4007_3000h base + 1h offset = 4007_3001h
CMP1_CR1 is 4007_3008h base + 1h offset = 4007_3009h
CMP2_CR1 is 4007_3010h base + 1h offset = 4007_3011h
CMP3_CR1 is 4007_3018h base + 1h offset = 4007_3019h

Bit	7	6	5	4	3	2	1	0
Read	SE	WE	0	PMODE	INV	COS	OPE	EN
Write			▨					
Reset	0	0	0	0	0	0	0	0

CMPx_CR1 field descriptions

Field	Description
7 SE	Sample Enable

Table continues on the next page...

CMPx_CR1 field descriptions (continued)

Field	Description
	At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Sampling mode not selected. 1 Sampling mode selected.
6 WE	Windowing Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Windowing mode not selected. 1 Windowing mode selected.
5 Reserved	This read-only field is reserved and always has the value zero.
4 PMODE	Power Mode Select Refer to the device data sheet's CMP electrical specifications table for details on the impact of the modes below. 0 Low Speed (LS) comparison mode selected. In this mode, CMP has slower output propagation delay and lower current consumption. 1 High Speed (HS) comparison mode selected. In this mode, CMP has faster output propagation delay and higher current consumption.
3 INV	Comparator INVERT This bit allows you to select the polarity of the analog comparator function. It is also driven to the COUT output (on both the device pin and as SCR[COUT]) when CR1[OPE]=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
2 COS	Comparator Output Select 0 Set CMPO to equal COUT (filtered comparator output). 1 Set CMPO to equal COUTA (unfiltered comparator output).
1 OPE	Comparator Output Pin Enable 0 The comparator output (CMPO) is not available on the associated CMPO output pin. 1 The comparator output (CMPO) is available on the associated CMPO output pin.
0 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, it remains in the off state, and consumes no power. When you select the same input from analog mux to the positive and negative port, the comparator is disabled automatically. 0 Analog Comparator disabled. 1 Analog Comparator enabled.

36.7.3 CMP Filter Period Register (CMPx_FPR)

Addresses: CMP0_FPR is 4007_3000h base + 2h offset = 4007_3002h
CMP1_FPR is 4007_3008h base + 2h offset = 4007_300Ah
CMP2_FPR is 4007_3010h base + 2h offset = 4007_3012h
CMP3_FPR is 4007_3018h base + 2h offset = 4007_301Ah

Bit	7	6	5	4	3	2	1	0
Read	FILT_PER							
Write								
Reset	0	0	0	0	0	0	0	0

CMPx_FPR field descriptions

Field	Description
7–0 FILT_PER	Filter Sample Period When CR1[SE] is equal to zero, this field specifies the sampling period, in bus clock cycles, of the comparator output filter. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional Description. This field has no effect when CR1[SE] is equal to one. In that case, the external SAMPLE signal is used to determine the sampling period.

36.7.4 CMP Status and Control Register (CMPx_SCR)

Addresses: CMP0_SCR is 4007_3000h base + 3h offset = 4007_3003h
CMP1_SCR is 4007_3008h base + 3h offset = 4007_300Bh
CMP2_SCR is 4007_3010h base + 3h offset = 4007_3013h
CMP3_SCR is 4007_3018h base + 3h offset = 4007_301Bh

Bit	7	6	5	4	3	2	1	0
Read	0	DMAEN	0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

CMPx_SCR field descriptions

Field	Description
7 Reserved	This read-only field is reserved and always has the value zero.
6 DMAEN	DMA Enable Control The DMAEN bit enables the DMA transfer triggered from the CMP module. When this bit is set, a DMA request is asserted when the CFR or CFF bit is set.

Table continues on the next page...

CMPx_SCR field descriptions (continued)

Field	Description
	0 DMA disabled. 1 DMA enabled.
5 Reserved	This read-only field is reserved and always has the value zero.
4 IER	Comparator Interrupt Enable Rising The IER bit enables the CFR interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFR bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 IEF	Comparator Interrupt Enable Falling The IEF bit enables the CFF interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFF bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
2 CFR	Analog Comparator Flag Rising During normal operation, the CFR bit is set when a rising edge on COUT has been detected. The CFR bit is cleared by writing a logic one to the bit. During Stop modes, CFR is level sensitive. 0 Rising edge on COUT has not been detected. 1 Rising edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling During normal operation, the CFF bit is set when a falling edge on COUT has been detected. The CFF bit is cleared by writing a logic one to the bit. During Stop modes, CFF is level senstive. 0 Falling edge on COUT has not been detected. 1 Falling edge on COUT has occurred.
0 COUT	Analog Comparator Output Reading the COUT bit will return the current value of the analog comparator output. The register bit is reset to zero and will read as CR1[INV] when the Analog Comparator module is disabled (CR1[EN] = 0). Writes to this bit are ignored.

36.7.5 DAC Control Register (CMPx_DACCR)

Addresses: CMP0_DACCR is 4007_3000h base + 4h offset = 4007_3004h
CMP1_DACCR is 4007_3008h base + 4h offset = 4007_300Ch
CMP2_DACCR is 4007_3010h base + 4h offset = 4007_3014h
CMP3_DACCR is 4007_3018h base + 4h offset = 4007_301Ch

Bit	7	6	5	4	3	2	1	0
Read Write	DACEN	VRSEL	VOSEL					
Reset	0	0	0	0	0	0	0	0

CMPx_DACCR field descriptions

Field	Description
7 DACEN	DAC Enable This bit is used to enable the DAC. When the DAC is disabled, it is powered down to conserve power. 0 DAC is disabled. 1 DAC is enabled.
6 VRSEL	Supply Voltage Reference Source Select 0 V_{in1} is selected as resistor ladder network supply reference Vin. 1 V_{in2} is selected as resistor ladder network supply reference Vin.
5–0 VOSEL	DAC Output Voltage Select This bit selects an output voltage from one of 64 distinct levels. `DACO = (Vin/64) * (VOSEL[5:0] + 1)`, so the DACO range is from Vin/64 to Vin.

36.7.6 MUX Control Register (CMPx_MUXCR)

Addresses: CMP0_MUXCR is 4007_3000h base + 5h offset = 4007_3005h
CMP1_MUXCR is 4007_3008h base + 5h offset = 4007_300Dh
CMP2_MUXCR is 4007_3010h base + 5h offset = 4007_3015h
CMP3_MUXCR is 4007_3018h base + 5h offset = 4007_301Dh

Bit	7	6	5	4	3	2	1	0
Read	0		PSEL			MSEL		
Write	▨	▨						
Reset	0	0	0	0	0	0	0	0

CMPx_MUXCR field descriptions

Field	Description
7–6 Reserved	This read-only field is reserved and always has the value zero.
5–3 PSEL	Plus Input MUX Control Determines which input is selected for the plus input of the comparator. For INx inputs, refer to CMP, DAC and ANMUX Blocks Diagram. **NOTE:** When an inappropriate operation selects the same input for both MUXes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7
2–0 MSEL	Minus Input MUX Control Determines which input is selected for the minus input of the comparator. For INx inputs, refer to CMP, DAC and ANMUX Blocks Diagram. **NOTE:** When an inappropriate operation selects the same input for both MUXes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 000 IN0 001 IN1 010 IN2 011 IN3 100 IN4 101 IN5 110 IN6 111 IN7

36.8 CMP Functional Description

The Comparator can be used to compare two analog input voltages applied to INP and INM. The analog comparator output (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 CR1[INV] = 1.

The SCR[IER], SCR[IEF] bits are used to select the condition which will cause the comparator module to assert an interrupt to the processor. SCR[CFF] is set on a falling edge and SCR[CFR] is set on rising edge of the comparator output. The (optionally filtered) comparator output can be read directly through the SCR[COUT] bit.

36.8.1 CMP Functional Modes

There are three main sub-blocks to the comparator module: the comparator itself, the window function and the filter function. The filter, CR0[FILTER_CNT] can be clocked from an internally or external clock source. Additionally, the filter is programmable with respect to how many samples must agree before a change on the output is registered. In the simplest case, only 1 sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CR1[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 CR1[WE]. When set, the comparator output is sampled only when the WINDOW input signal is equal to one. 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 36-36. Comparator Sample/Filter Controls

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
1	0	X	X	X	X	**Disabled** Refer to the Disabled Mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode** Refer to the Continuous Mode (#s 2A & 2B).
2B	1	0	0	X	0x00	
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode** Refer to the Sampled, Non-Filtered Mode (#s 3A & 3B).
3B	1	0	0	0x01	> 0x00	
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode** Refer to the Sampled, Filtered Mode (#s 4A & 4B).
4B	1	0	0	> 0x01	> 0x00	
5A	1	1	0	0x00	X	**Windowed mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA Refer to the Windowed Mode (#s 5A & 5B).
5B	1	1	0	X	0x00	

Table continues on the next page...

Table 36-36. Comparator Sample/Filter Controls (continued)

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
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 FILT_PER to generate COUT. Refer to 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. Refer to the Windowed/Filtered Mode (#7).
All other combinations of CR1[EN], CR1[WE], CR1[SE], CR0[FILTER_CNT], and FPR[FILT_PER] 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 should generally 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 should be changed only after setting CR1[SE]=0 and CR0[FILTER_CNT]=0x00. This has the effect of resetting the filter to a known state.

36.8.1.1 Disabled Mode (# 1)

In disabled mode, the analog comparator is non-functional and consumes no power. The output of the analog comparator block (CMPO) is zero in this mode.

36.8.1.2 Continuous Mode (#s 2A & 2B)

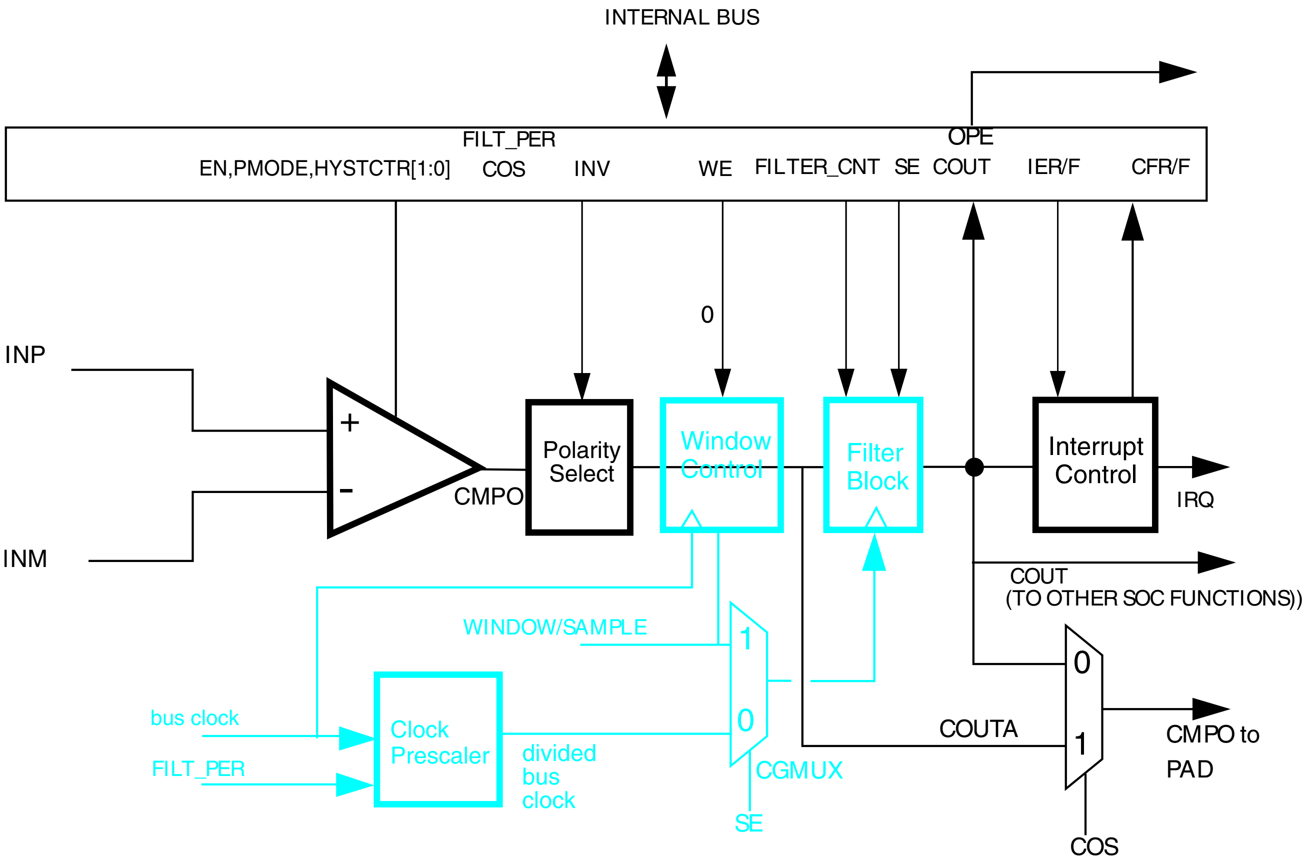


Figure 36-33. Comparator Operation in Continuous Mode

NOTE

Refer to 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. SCR[COUT] is updated continuously. The path from comparator inputs pins to output pin is operating in combinational (unclocked) mode. COUT and COUTA are identical.

For control configurations which result in disabling the Filter Block, refer to Filter Block Bypass Logic diagram.

36.8.1.3 Sampled, Non-Filtered Mode (#s 3A & 3B)

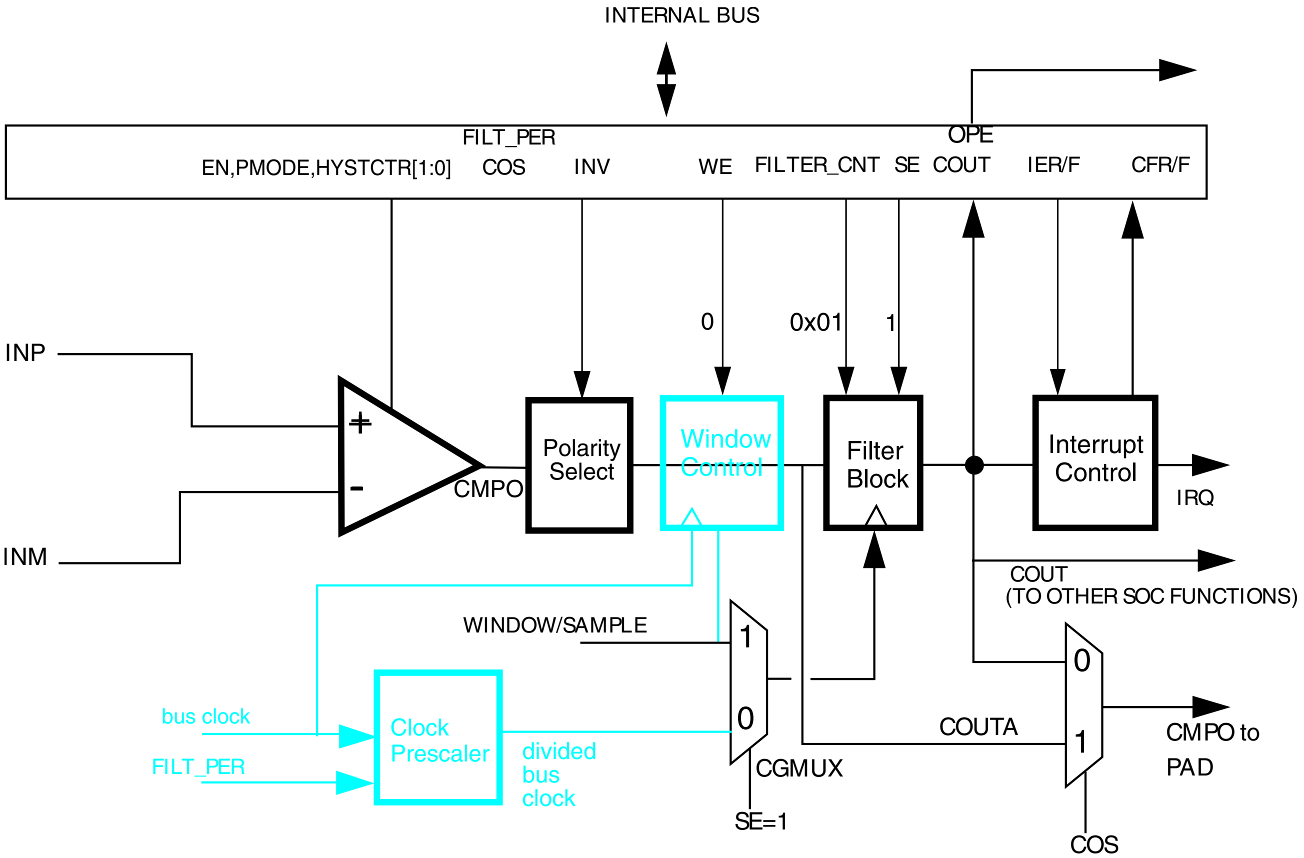


Figure 36-34. Sampled, Non-Filtered (# 3A): Sampling point externally driven

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).

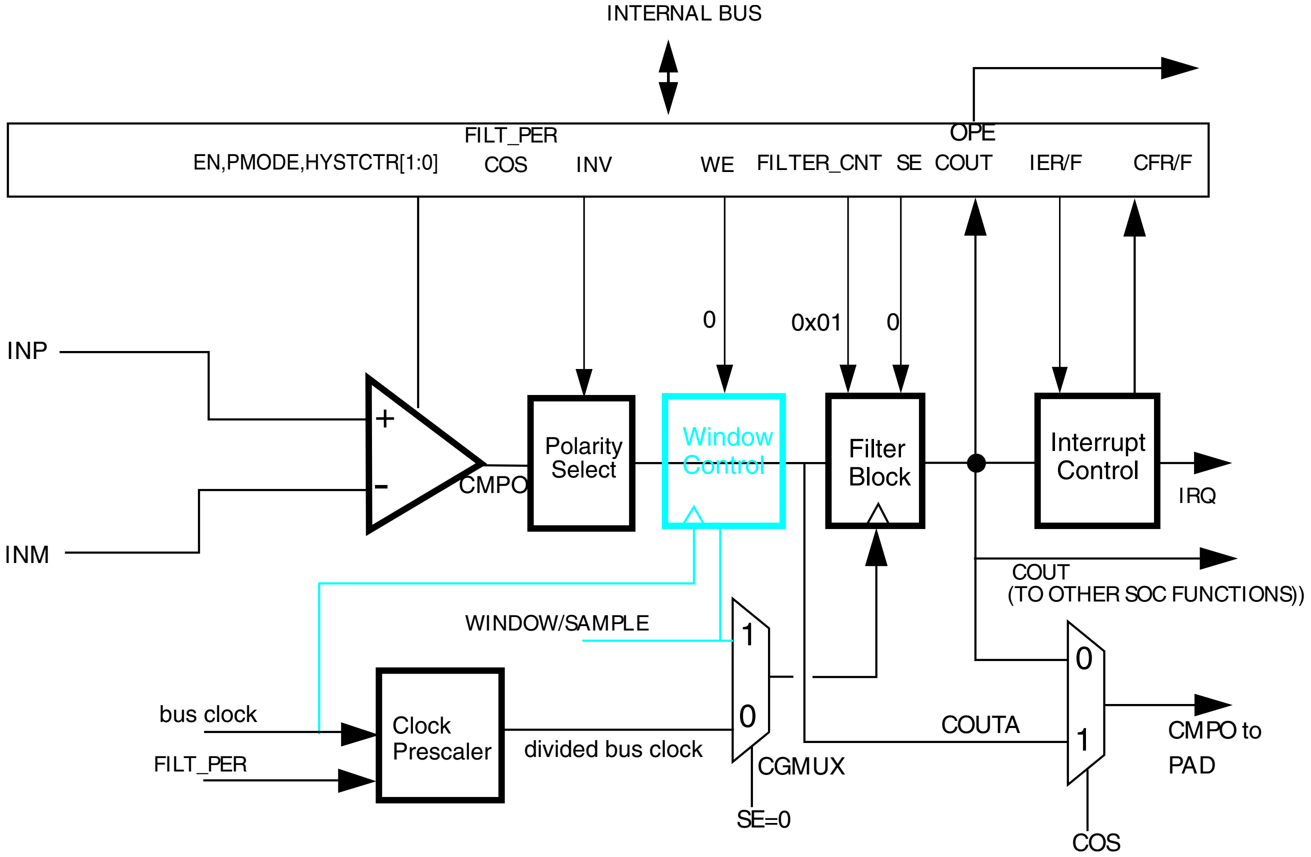


Figure 36-35. Sampled, Non-Filtered (# 3B): Sampling interval internally derived

36.8.1.4 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 CR0[FILTER_CNT] is now greater than 1, which activates filter operation.

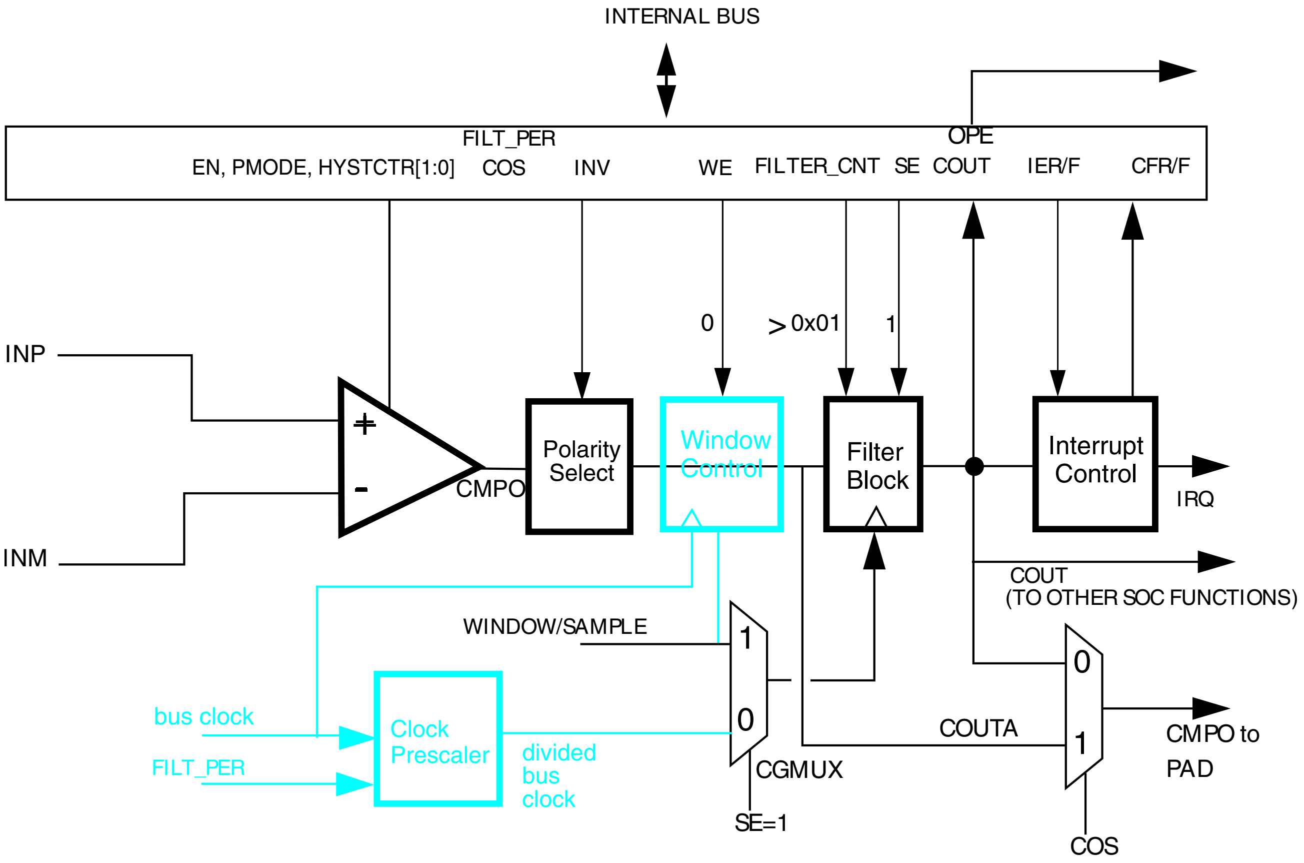


Figure 36-36. Sampled, Filtered (# 4A): Sampling point externally driven

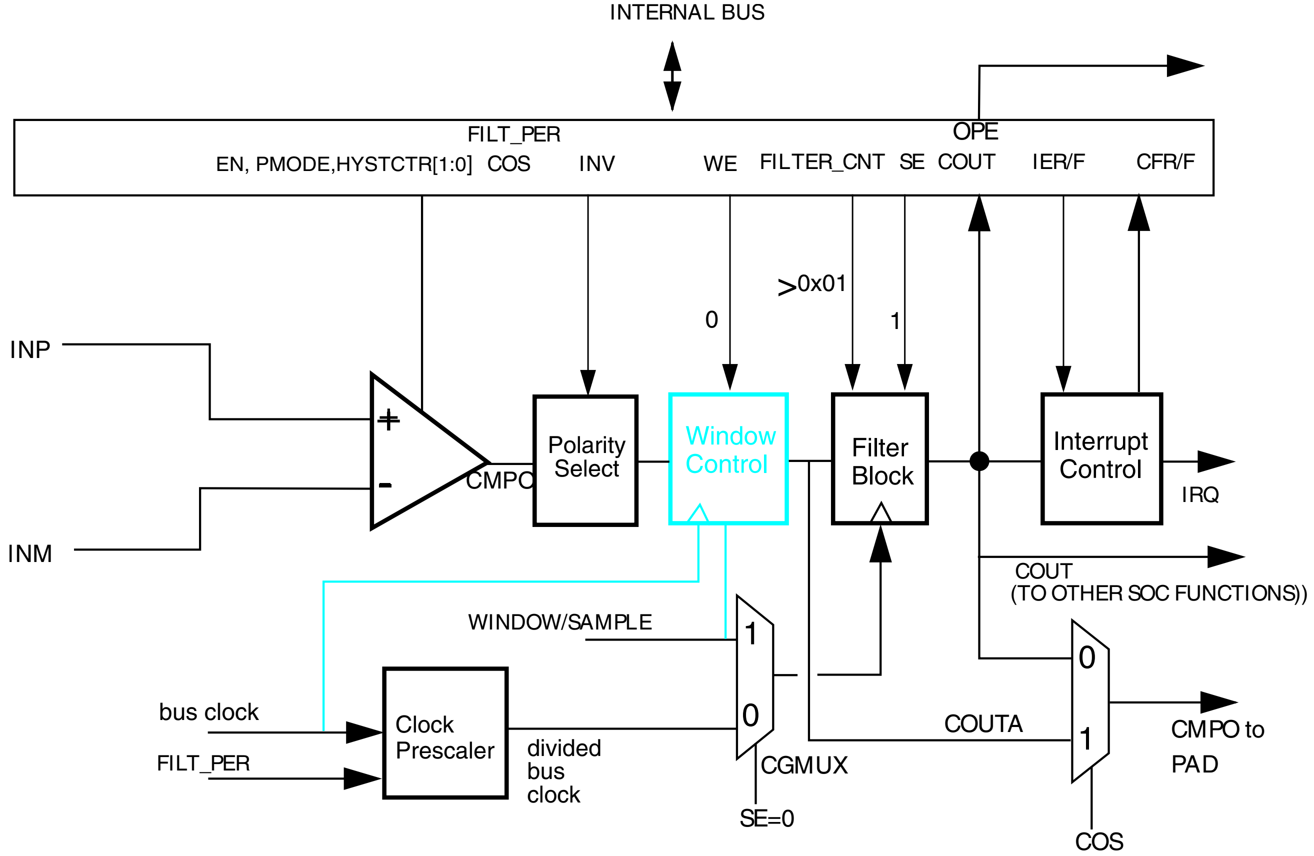


Figure 36-37. Sampled, Filtered (# 4B): Sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that CR0[FILTER_CNT] is now greater than 1, which activates filter operation.

36.8.1.5 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". Note that 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.

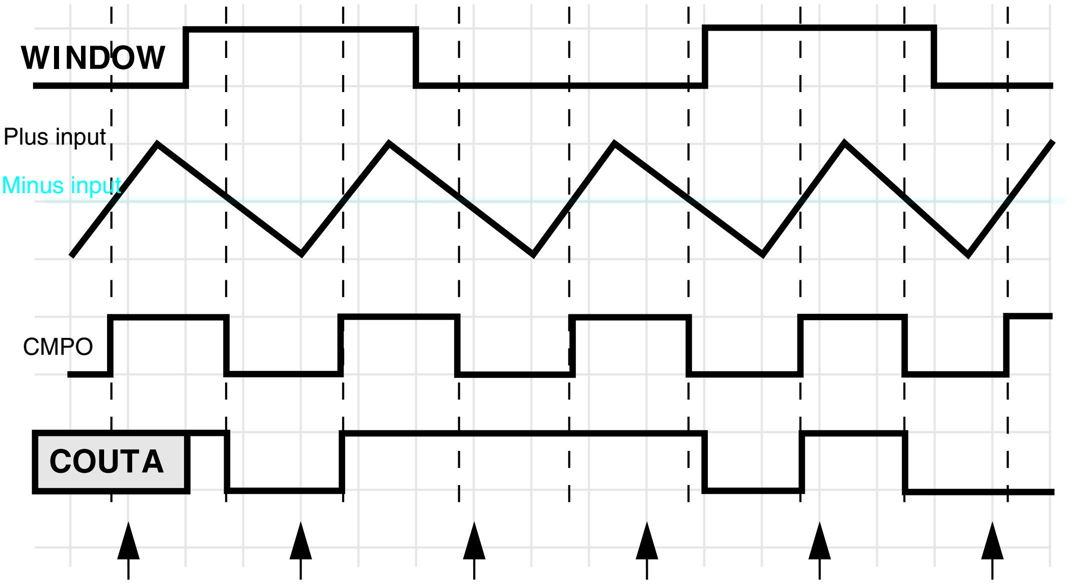


Figure 36-38. Windowed Mode Operation

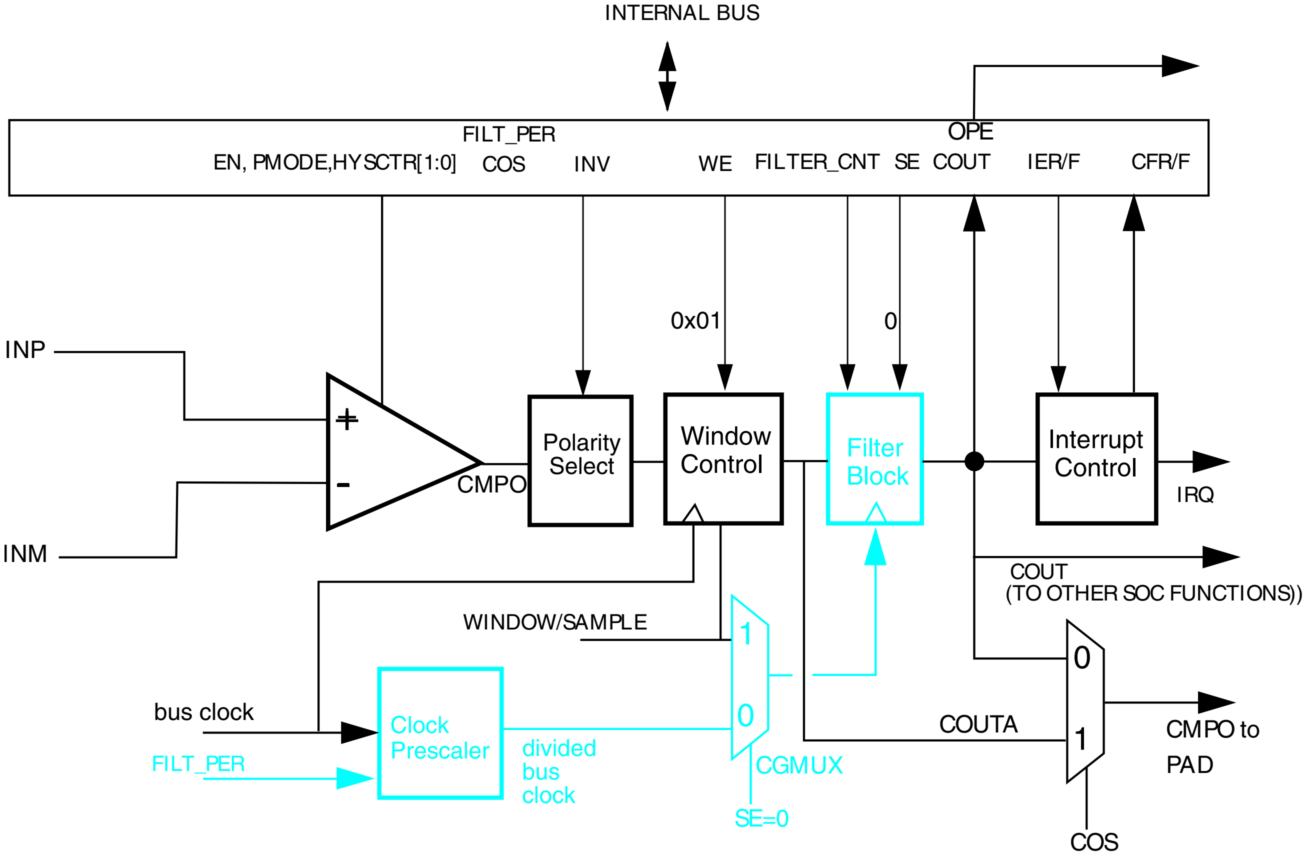


Figure 36-39. Windowed Mode

For control configurations which result in disabling the Filter Block, refer to Filter Block Bypass Logic diagram.

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.

36.8.1.6 Windowed/Resampled Mode (# 6)

The following figure uses the same input stimulus shown in Figure 36-38, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows. Again, prop delays and latency is ignored for clarity's sake. This example was generated solely to demonstrate operation of the comparator in windowing / 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.

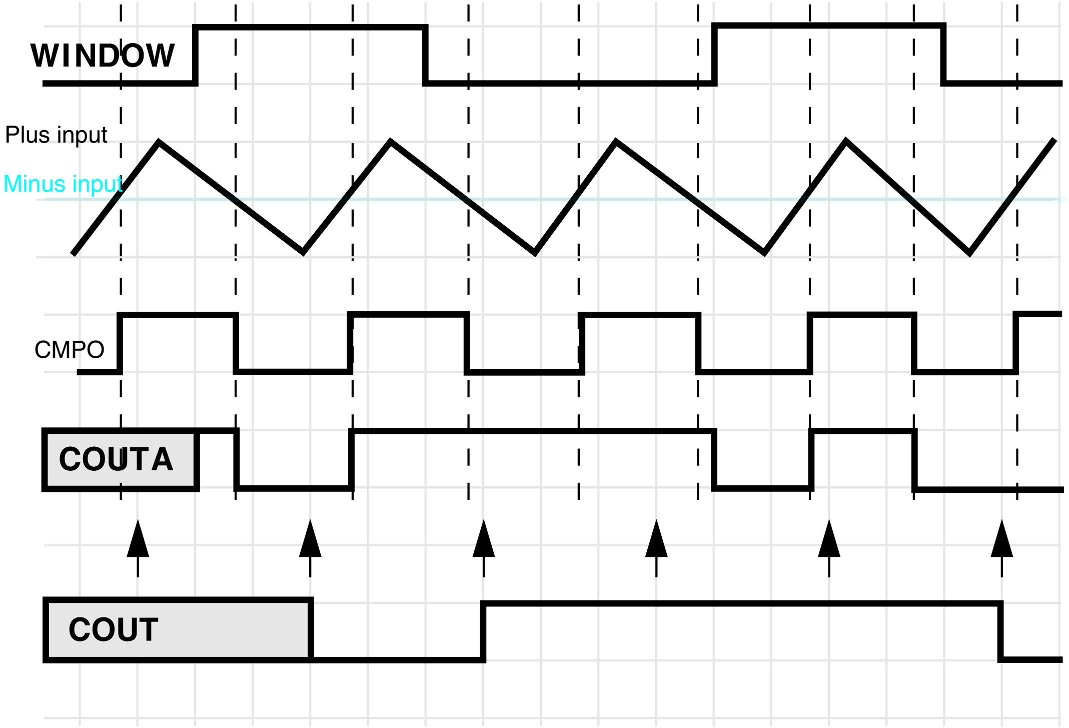


Figure 36-40. Windowed / Resampled Mode Operation

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 CR0[FILTER_CNT] must be exactly one.

36.8.1.7 Windowed/Filtered Mode (#7)

This is the most complex mode of operation for the comparator block, as it utilizes 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 + ((CR0[FILTER_CNT] X FPR[FILT_PER]) + 1) X 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.

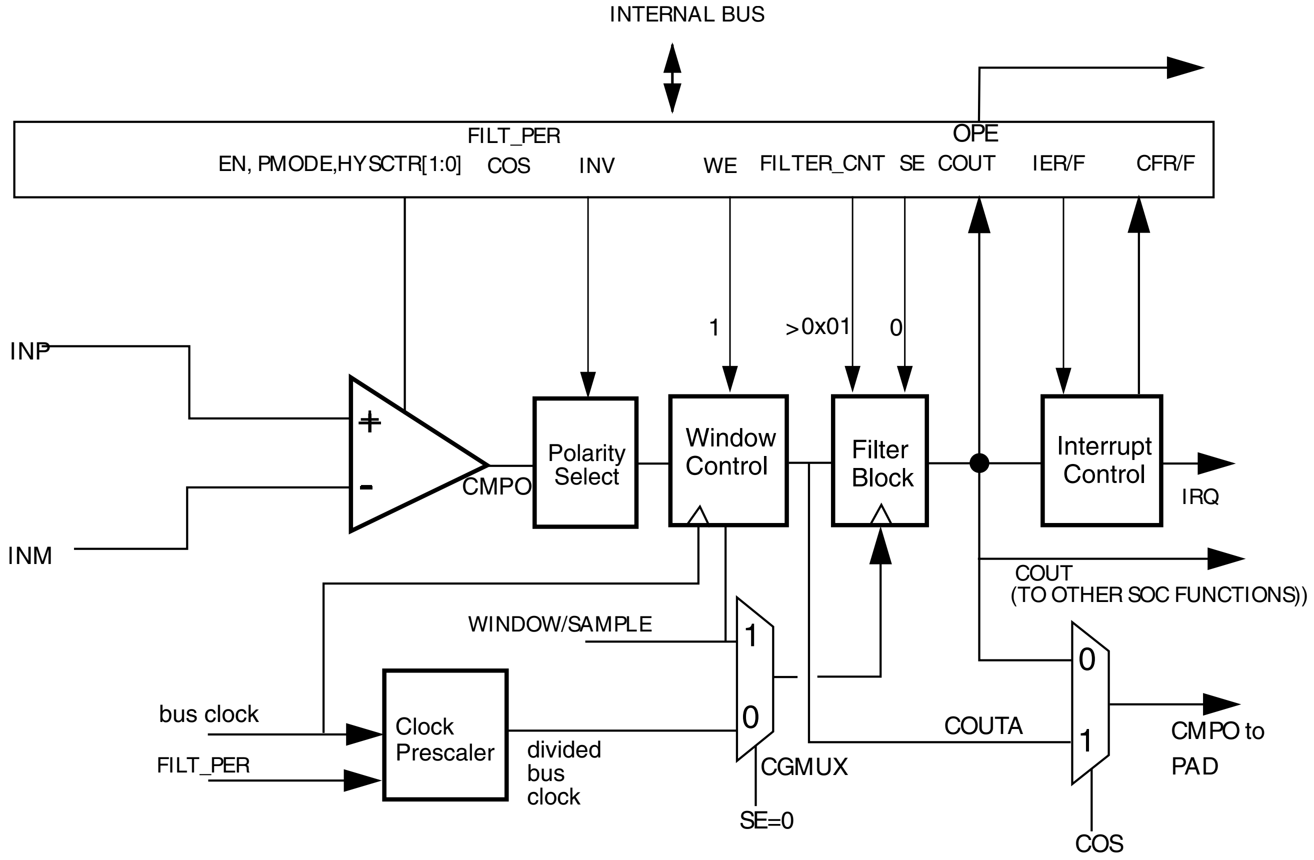


Figure 36-41. Windowed/Filtered Mode

36.8.2 Power Modes

36.8.2.1 Wait Mode Operation

During Wait and VLPW modes and if enabled, the CMP continues to operate normally. Also, if enabled, a CMP interrupt can wake the MCU.

36.8.2.2 Stop Mode Operation

Subject to platform-specific clock restrictions, the MCU is brought out of stop when a compare event occurs and the corresponding interrupt is enabled. Similarly, if CR1[OPE] is enabled, the comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. In stop modes, the comparator can be operational in both high speed (HS) comparison mode (CR1[PMODE] = 1) and low speed (LS) comparison mode (CR1[PMODE] = 0), but it is recommended to use the low speed mode to minimize power consumption.

If stop is exited with a reset, all comparator registers are put into their reset state.

36.8.2.3 Low-Leakage Mode Operation

When the chip is in low-leakage modes, the CMP module is partially functional and is limited to low speed mode (regardless of the CR1[PMODE] bit's setting). Windowed, sampled, and filtered modes are not supported. The CMP output pin is latched and does not reflect the compare output state.

The positive- and negative-input voltage can be from external pins or the DAC output. The MCU can be brought out of the low-leakage mode if a compare event occurs and the CMP interrupt is enabled. After wakeup from low-leakage modes, the CMP module is in the reset state except for the SCR[CFF] and SCR[CFR] flags.

36.8.3 Startup and Operation

A typical startup sequence is as follows.

The time required to stabilize COUT will be 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. Power on delay of the comparators are available from data sheets. The windowing function has a maximum of 1 bus clock period delay. Filter delay is specified in Low Pass Filter.

During operation, the propagation delay of the selected data paths must always be considered. It can take many bus clock cycles for COUT and the CFR/CFF status bits 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 will initially be equal to zero until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic one.

36.8.4 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 FPR[FILT_PER], 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.

36.8.4.1 Enabling Filter Modes

Filter Modes are enabled by setting CR0[FILTER_CNT] greater than 0x01 and (setting FPR[FILT_PER] to a non-zero value OR setting CR1[SE]=1). If using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT_PER] bus clock cycles.

The filter output will be at logic zero when first initalized, and will subsequently change when CR0[FILTER_CNT] consecutive samples all agree that the output value has changed. Said another way, SCR[COUT] will be zero for some initial period, even when COUTA is at logic one.

Setting both CR1[SE] and FPR[FILT_PER] 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 CR0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CR1[SE]=1, the filter takes samples of COUTA on each positive transition of the sample input. The output state of the filter changes when CR0[FILTER_CNT] consecutive samples all agree that the output value has changed.

36.8.4.2 Latency Issues

The FPR[FILT_PER] value (or SAMPLE period) should be set such that the sampling period is just larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The CR0[FILTER_CNT] value should 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 CR0[FILTER_CNT] power.

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.

The values of FPR[FILT_PER] (or SAMPLE period) and CR0[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 CR0[FILTER_CNT] power.

Table 36-37. Comparator Sample/Filter Maximum Latencies

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	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} + (FPR[FILT_PER] x T_{per}) + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (CR0[FILTER_CNT] x T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (CR0[FILTER_CNT] x FPR[FILT_PER] 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} + (FPR[FILT_PER] x T_{per}) + 2T_{per}

Table continues on the next page...

Table 36-37. Comparator Sample/Filter Maximum Latencies (continued)

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation	Maximum Latency[1]
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (CR0[FILTER_CNT] x FPR[FILT_PER] 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.

36.9 CMP Interrupts

The CMP module is capable of generating an interrupt on either the rising or falling edge of the comparator output (or both). The interrupt request is asserted when both SCR[IER] bit and SCR[CFR] are set. It is also asserted when both SCR[IEF] bit and SCR[CFF] are set. The interrupt is de-asserted by clearing either SCR[IER] or SCR[CFR] for a rising edge interrupt, or SCR[IEF] and SCR[CFF] for a falling edge interrupt.

36.10 CMP DMA Support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support (set SCR[DMAEN]) enables and the interrupt enables (set SCR[IER] or SCR[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 dma_done signal that de-asserts the dma_request and clears the flag to allow a subsequent change on comparator output to occur and force another DMA request.

36.11 Digital to Analog Converter Block Diagram

The following figure shows the block diagram of the DAC module. It contains a 64-tap resistor ladder network and a 64-to-1 multiplexer, which selects an output voltage from one of 64 distinct levels that outputs from DACO. It is controlled through DAC Control register (DACCR). Its supply reference source can be selected from two sources V_{in1} and V_{in2}. The module can be powered down (disabled) when it is not used. When in disable mode, DACO is connected to the analog ground.

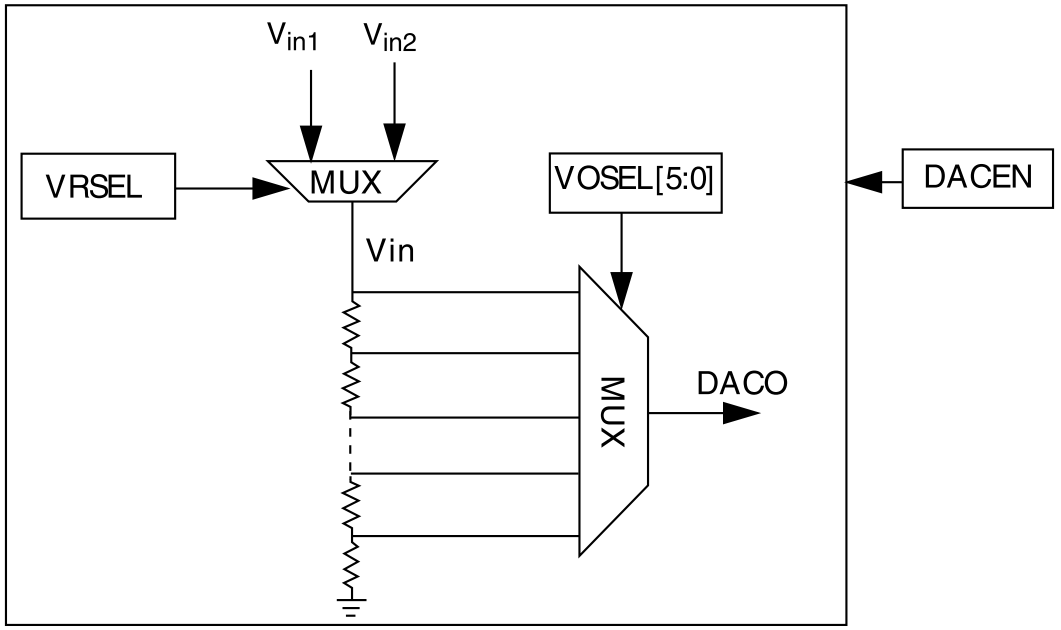


Figure 36-42. 6-bit DAC Block Diagram

36.12 DAC Functional Description

This section provides DAC functional description.

36.12.1 Voltage Reference Source Select

- V_{in1} should be used to connect to the primary voltage source as supply reference of 64 tap resistor ladder
- V_{in2} should be used to connect to alternate voltage source (or primary source if alternate voltage source is not available)

36.13 DAC Resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

36.14 DAC Clocks

This module has a single clock input, the bus clock.

36.15 DAC Interrupts

This module has no interrupts.

Chapter 37
12-bit Digital-to-Analog Converter (DAC)

37.1 Introduction

NOTE

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

The 12-bit digital-to-analog converter (DAC) is a low power general purpose DAC. The output of this DAC can be placed on an external pin or set as one of the inputs to the analog comparator, Op-Amps, ADC, or other peripherals.

37.2 Features

The DAC module features include:

- On-chip programmable reference generator output (voltage output from 1/4096 Vin to Vin, step is 1/4096 Vin)
- Vin can be selected from two reference sources
- Static operation in Normal Stop mode
- 16-word data buffer supported with configurable watermark and multiple operation modes
- DMA support

37.3 Block Diagram

The block diagram of the DAC module is as follows:

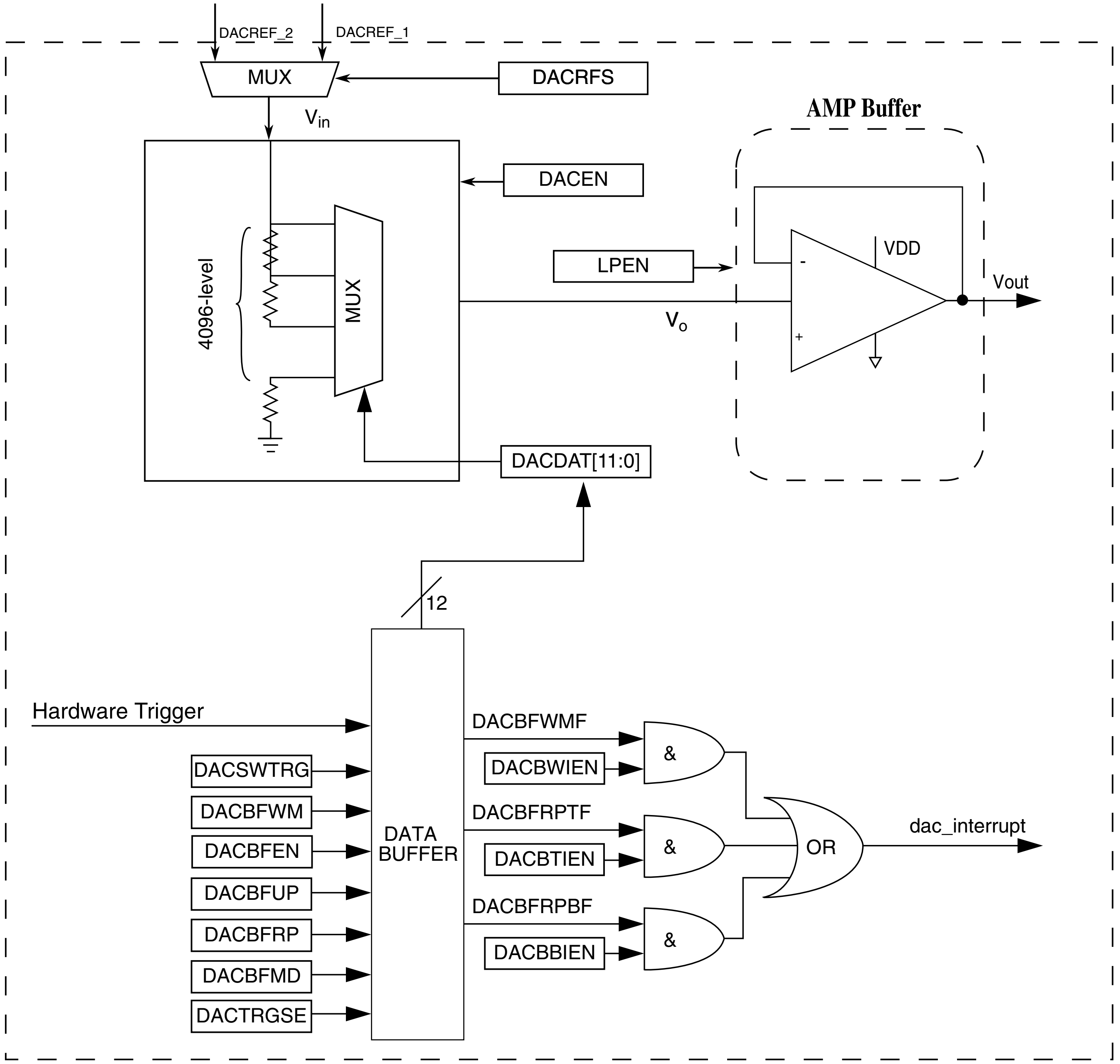


Figure 37-1. DAC Block Diagram

37.4 Memory Map/Register Definition

The DAC has registers to control analog comparator and programmable voltage divider to perform the digital-to-analog functions.

The address of a register is the sum of a base address and an address offset. The base address is defined at the chip level. The address offset is defined at the module level.

NOTE

Do not use 32/16-bit access to DACx_Cn(n=0, 1, 2) and DACx_SR register.

DAC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400C_C000	DAC Data Low Register (DAC0_DAT0L)	8	R/W	00h	37.4.1/ 1005
400C_C001	DAC Data High Register (DAC0_DAT0H)	8	R/W	00h	37.4.2/ 1005
400C_C002	DAC Data Low Register (DAC0_DAT1L)	8	R/W	00h	37.4.1/ 1005
400C_C003	DAC Data High Register (DAC0_DAT1H)	8	R/W	00h	37.4.2/ 1005
400C_C004	DAC Data Low Register (DAC0_DAT2L)	8	R/W	00h	37.4.1/ 1005
400C_C005	DAC Data High Register (DAC0_DAT2H)	8	R/W	00h	37.4.2/ 1005
400C_C006	DAC Data Low Register (DAC0_DAT3L)	8	R/W	00h	37.4.1/ 1005
400C_C007	DAC Data High Register (DAC0_DAT3H)	8	R/W	00h	37.4.2/ 1005
400C_C008	DAC Data Low Register (DAC0_DAT4L)	8	R/W	00h	37.4.1/ 1005
400C_C009	DAC Data High Register (DAC0_DAT4H)	8	R/W	00h	37.4.2/ 1005
400C_C00A	DAC Data Low Register (DAC0_DAT5L)	8	R/W	00h	37.4.1/ 1005
400C_C00B	DAC Data High Register (DAC0_DAT5H)	8	R/W	00h	37.4.2/ 1005
400C_C00C	DAC Data Low Register (DAC0_DAT6L)	8	R/W	00h	37.4.1/ 1005

Table continues on the next page...

DAC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400C_C00D	DAC Data High Register (DAC0_DAT6H)	8	R/W	00h	37.4.2/ 1005
400C_C00E	DAC Data Low Register (DAC0_DAT7L)	8	R/W	00h	37.4.1/ 1005
400C_C00F	DAC Data High Register (DAC0_DAT7H)	8	R/W	00h	37.4.2/ 1005
400C_C010	DAC Data Low Register (DAC0_DAT8L)	8	R/W	00h	37.4.1/ 1005
400C_C011	DAC Data High Register (DAC0_DAT8H)	8	R/W	00h	37.4.2/ 1005
400C_C012	DAC Data Low Register (DAC0_DAT9L)	8	R/W	00h	37.4.1/ 1005
400C_C013	DAC Data High Register (DAC0_DAT9H)	8	R/W	00h	37.4.2/ 1005
400C_C014	DAC Data Low Register (DAC0_DAT10L)	8	R/W	00h	37.4.1/ 1005
400C_C015	DAC Data High Register (DAC0_DAT10H)	8	R/W	00h	37.4.2/ 1005
400C_C016	DAC Data Low Register (DAC0_DAT11L)	8	R/W	00h	37.4.1/ 1005
400C_C017	DAC Data High Register (DAC0_DAT11H)	8	R/W	00h	37.4.2/ 1005
400C_C018	DAC Data Low Register (DAC0_DAT12L)	8	R/W	00h	37.4.1/ 1005
400C_C019	DAC Data High Register (DAC0_DAT12H)	8	R/W	00h	37.4.2/ 1005
400C_C01A	DAC Data Low Register (DAC0_DAT13L)	8	R/W	00h	37.4.1/ 1005
400C_C01B	DAC Data High Register (DAC0_DAT13H)	8	R/W	00h	37.4.2/ 1005
400C_C01C	DAC Data Low Register (DAC0_DAT14L)	8	R/W	00h	37.4.1/ 1005
400C_C01D	DAC Data High Register (DAC0_DAT14H)	8	R/W	00h	37.4.2/ 1005
400C_C01E	DAC Data Low Register (DAC0_DAT15L)	8	R/W	00h	37.4.1/ 1005
400C_C01F	DAC Data High Register (DAC0_DAT15H)	8	R/W	00h	37.4.2/ 1005
400C_C020	DAC Status Register (DAC0_SR)	8	R	02h	37.4.3/ 1006

Table continues on the next page...

DAC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400C_C021	DAC Control Register (DAC0_C0)	8	R/W	00h	37.4.4/ 1006
400C_C022	DAC Control Register 1 (DAC0_C1)	8	R/W	00h	37.4.5/ 1008
400C_C023	DAC Control Register 2 (DAC0_C2)	8	R/W	0Fh	37.4.6/ 1009
400C_D000	DAC Data Low Register (DAC1_DAT0L)	8	R/W	00h	37.4.1/ 1005
400C_D001	DAC Data High Register (DAC1_DAT0H)	8	R/W	00h	37.4.2/ 1005
400C_D002	DAC Data Low Register (DAC1_DAT1L)	8	R/W	00h	37.4.1/ 1005
400C_D003	DAC Data High Register (DAC1_DAT1H)	8	R/W	00h	37.4.2/ 1005
400C_D004	DAC Data Low Register (DAC1_DAT2L)	8	R/W	00h	37.4.1/ 1005
400C_D005	DAC Data High Register (DAC1_DAT2H)	8	R/W	00h	37.4.2/ 1005
400C_D006	DAC Data Low Register (DAC1_DAT3L)	8	R/W	00h	37.4.1/ 1005
400C_D007	DAC Data High Register (DAC1_DAT3H)	8	R/W	00h	37.4.2/ 1005
400C_D008	DAC Data Low Register (DAC1_DAT4L)	8	R/W	00h	37.4.1/ 1005
400C_D009	DAC Data High Register (DAC1_DAT4H)	8	R/W	00h	37.4.2/ 1005
400C_D00A	DAC Data Low Register (DAC1_DAT5L)	8	R/W	00h	37.4.1/ 1005
400C_D00B	DAC Data High Register (DAC1_DAT5H)	8	R/W	00h	37.4.2/ 1005
400C_D00C	DAC Data Low Register (DAC1_DAT6L)	8	R/W	00h	37.4.1/ 1005
400C_D00D	DAC Data High Register (DAC1_DAT6H)	8	R/W	00h	37.4.2/ 1005
400C_D00E	DAC Data Low Register (DAC1_DAT7L)	8	R/W	00h	37.4.1/ 1005
400C_D00F	DAC Data High Register (DAC1_DAT7H)	8	R/W	00h	37.4.2/ 1005
400C_D010	DAC Data Low Register (DAC1_DAT8L)	8	R/W	00h	37.4.1/ 1005

Table continues on the next page...

DAC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400C_D011	DAC Data High Register (DAC1_DAT8H)	8	R/W	00h	37.4.2/ 1005
400C_D012	DAC Data Low Register (DAC1_DAT9L)	8	R/W	00h	37.4.1/ 1005
400C_D013	DAC Data High Register (DAC1_DAT9H)	8	R/W	00h	37.4.2/ 1005
400C_D014	DAC Data Low Register (DAC1_DAT10L)	8	R/W	00h	37.4.1/ 1005
400C_D015	DAC Data High Register (DAC1_DAT10H)	8	R/W	00h	37.4.2/ 1005
400C_D016	DAC Data Low Register (DAC1_DAT11L)	8	R/W	00h	37.4.1/ 1005
400C_D017	DAC Data High Register (DAC1_DAT11H)	8	R/W	00h	37.4.2/ 1005
400C_D018	DAC Data Low Register (DAC1_DAT12L)	8	R/W	00h	37.4.1/ 1005
400C_D019	DAC Data High Register (DAC1_DAT12H)	8	R/W	00h	37.4.2/ 1005
400C_D01A	DAC Data Low Register (DAC1_DAT13L)	8	R/W	00h	37.4.1/ 1005
400C_D01B	DAC Data High Register (DAC1_DAT13H)	8	R/W	00h	37.4.2/ 1005
400C_D01C	DAC Data Low Register (DAC1_DAT14L)	8	R/W	00h	37.4.1/ 1005
400C_D01D	DAC Data High Register (DAC1_DAT14H)	8	R/W	00h	37.4.2/ 1005
400C_D01E	DAC Data Low Register (DAC1_DAT15L)	8	R/W	00h	37.4.1/ 1005
400C_D01F	DAC Data High Register (DAC1_DAT15H)	8	R/W	00h	37.4.2/ 1005
400C_D020	DAC Status Register (DAC1_SR)	8	R	02h	37.4.3/ 1006
400C_D021	DAC Control Register (DAC1_C0)	8	R/W	00h	37.4.4/ 1006
400C_D022	DAC Control Register 1 (DAC1_C1)	8	R/W	00h	37.4.5/ 1008
400C_D023	DAC Control Register 2 (DAC1_C2)	8	R/W	0Fh	37.4.6/ 1009

37.4.1 DAC Data Low Register (DACx_DATL)

Addresses: 400C_C000h base + 0h offset + (2d × *n*), where *n* = 0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	DATA[7:0]							
Write								
Reset	0	0	0	0	0	0	0	0

DACx_DAT*n*L field descriptions

Field	Description
7–0 DATA[7:0]	When the DAC Buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula. Vout = Vin * (1 + DACDAT0[11:0])/4096 When the DAC Buffer is enabled, DATA is mapped to the 16-word buffer.

37.4.2 DAC Data High Register (DACx_DATH)

Addresses: 400C_C000h base + 1h offset + (2d × *n*), where *n* = 0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	0				DATA[11:8]			
Write	▨	▨	▨	▨				
Reset	0	0	0	0	0	0	0	0

DACx_DAT*n*H field descriptions

Field	Description
7–4 Reserved	This read-only field is reserved and always has the value zero.
3–0 DATA[11:8]	When the DAC Buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula. Vout = Vin * (1 + DACDAT0[11:0])/4096 When the DAC Buffer is enabled, DATA[11:0] is mapped to the 16-word buffer.

37.4.3 DAC Status Register (DACx_SR)

If DMA is enabled, the flags can be cleared automatically by DMA when the DMA request is done. Write zero to a bit to clear it. Writing one has no effect. After reset DACBFRPTF is set and can be cleared by software, if needed. The flags are set only when the data buffer status is changed.

Addresses: DAC0_SR is 400C_C000h base + 20h offset = 400C_C020h
DAC1_SR is 400C_D000h base + 20h offset = 400C_D020h

Bit	7	6	5	4	3	2	1	0
Read	0					DACBFWMF	DACBFRPTF	DACBFRPBF
Write								
Reset	0	0	0	0	0	0	1	0

DACx_SR field descriptions

Field	Description
7–3 Reserved	This read-only field is reserved and always has the value zero. Reserved
2 DACBFWMF	DAC buffer watermark flag 0 The DAC buffer read pointer has not reached the watermark level. 1 The DAC buffer read pointer has reached the watermark level.
1 DACBFRPTF	DAC buffer read pointer top position flag 0 The DAC buffer read pointer is not zero. 1 The DAC buffer read pointer is zero.
0 DACBFRPBF	DAC buffer read pointer bottom position flag 0 The DAC buffer read pointer is not equal to the DACBFUP. 1 The DAC buffer read pointer is equal to the DACBFUP.

37.4.4 DAC Control Register (DACx_C0)

Addresses: DAC0_C0 is 400C_C000h base + 21h offset = 400C_C021h
DAC1_C0 is 400C_D000h base + 21h offset = 400C_D021h

Bit	7	6	5	4	3	2	1	0
Read	DACEN	DACRFS	DACTRGSEL	0	LPEN	DACBWIEN	DACBTIEN	DACBBIEN
Write				DACSWTRG				
Reset	0	0	0	0	0	0	0	0

DACx_C0 field descriptions

Field	Description
7 DACEN	DAC enable The DACEN bit starts the Programmable Reference Generator operation. 0 The DAC system is disabled. 1 The DAC system is enabled.
6 DACRFS	DAC Reference Select 0 The DAC selets DACREF_1 as the reference voltage. 1 The DAC selets DACREF_2 as the reference voltage.
5 DACTRGSEL	DAC trigger select 0 The DAC hardware trigger is selected. 1 The DAC software trigger is selected.
4 DACSWTRG	DAC software trigger Active high. This is a write-only bit, read it always be 0. If DAC software trigger is selected and buffer enabled, write 1 to this bit will advance the buffer read pointer once. 0 The DAC soft trigger is not valid. 1 The DAC soft trigger is valid.
3 LPEN	DAC low power control Refer to the device data sheet's 12-bit DAC electrical characteristics for details on the impact of the modes below. 0 high power mode. 1 low power mode.
2 DACBWIEN	DAC buffer watermark interrupt enable 0 The DAC buffer watermark interrupt is disabled. 1 The DAC buffer watermark interrupt is enabled.
1 DACBTIEN	DAC buffer read pointer top flag interrupt enable 0 The DAC buffer read pointer top flag interrupt is disabled. 1 The DAC buffer read pointer top flag interrupt is enabled.
0 DACBBIEN	DAC buffer read pointer bottom flag interrupt enable 0 The DAC buffer read pointer bottom flag interrupt is disabled. 1 The DAC buffer read pointer bottom flag interrupt is enabled.

37.4.5 DAC Control Register 1 (DACx_C1)

Addresses: DAC0_C1 is 400C_C000h base + 22h offset = 400C_C022h
DAC1_C1 is 400C_D000h base + 22h offset = 400C_D022h

Bit	7	6	5	4	3	2	1	0
Read	DMAEN	0		DACBFWM		DACBFMD		DACBFEN
Write								
Reset	0	0	0	0	0	0	0	0

DACx_C1 field descriptions

Field	Description
7 DMAEN	DMA enable select 0 DMA disabled. 1 DMA enabled. When DMA enabled, DMA request will be generated by original interrupts. And interrupts will not be presented on this module at the same time.
6–5 Reserved	This read-only field is reserved and always has the value zero.
4–3 DACBFWM	DAC buffer watermark select This bitfield controls when the DAC buffer watermark flag will be set. When the DAC buffer read pointer reaches the word defined by this bitfield, from 1 to 4 words away from the upper limit (DACBUP), the DAC buffer watermark flag will be set. This allows user configuration of the watermark interrupt. 00 1 word 01 2 words 10 3 words 11 4 words
2–1 DACBFMD	DAC buffer work mode select 00 Normal Mode 01 Swing Mode 10 One-Time Scan Mode 11 Reserved
0 DACBFEN	DAC buffer enable 0 Buffer read pointer disabled. The converted data is always the first word of the buffer. 1 Buffer read pointer enabled. The converted data is the word that the read pointer points to. It means converted data can be from any word of the buffer.

37.4.6 DAC Control Register 2 (DACx_C2)

Addresses: DAC0_C2 is 400C_C000h base + 23h offset = 400C_C023h
DAC1_C2 is 400C_D000h base + 23h offset = 400C_D023h

Bit	7	6	5	4	3	2	1	0
Read Write	DACBFRP				DACBFUP			
Reset	0	0	0	0	1	1	1	1

DACx_C2 field descriptions

Field	Description
7–4 DACBFRP	DAC buffer read pointer These 4 bits keep the current value of the buffer read pointer.
3–0 DACBFUP	DAC buffer upper limit These 4 bits select the buffer's upper limit. The buffer read pointer cannot exceed it.

37.5 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 DACRFS bit of C0 register. Refer to the module introduction for information on the source 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 $V_{in}/4096$ to V_{in}, and the step is $V_{in}/4096$.

37.5.1 DAC Data Buffer Operation

When the DAC is enabled and the buffer is not enabled, the DAC module always converts the data in DAT0 to analog output voltage.

When both the DAC and the buffer are enabled, the DAC converts the data in the data buffer to analog output voltage. The data buffer read pointer advances to the next word in the event the hardware trigger or the software trigger occurs. Refer to the PDB Module Interconnections section in Chip Configuration chapter for the hardware trigger connection.

The data buffer can be configured to operate in normal mode, swing mode or one-time scan mode. When the buffer operation is switched from one mode to another, the read pointer does not change. The read pointer can be set to any value between "0" and DACBFUP by writing DACBFRP in C2 register.

37.5.1.1 DAC Data Buffer Interrupts

There are several interrupts and associated flags that can be configured for the DAC buffer. The DAC read pointer bottom position flag is set when the DAC buffer read pointer reaches the DAC buffer upper limit. (DACBFRP = DACBFUP). The DAC read pointer top position flag is set when the DAC read pointer is equal to the start position, 0. Finally, the DAC buffer watermark flag is set when the DAC buffer read pointer has reached the position defined by the DAC watermark select bit field. The DAC watermark select (DACBFWM) can be used to generate an interrupt when the DAC buffer Read pointer is between 1 to 4 words from the DAC buffer upper limit.

37.5.1.2 Buffer Normal Mode

This is the default mode. The buffer works as a circular buffer. The read pointer increases by one, every time when the trigger occurs. When the read pointer reaches the upper limit, it goes to the zero directly in the next trigger event.

37.5.1.3 Buffer Swing Mode

This mode is similar to the normal mode. But when the read pointer reaches the upper limit, it does not go to the zero. It will descend by one in the next trigger events until zero is reached.

37.5.1.4 Buffer One-time Scan Mode

The read pointer increases by one every time when the trigger occurs. When it reaches the upper limit, it stops at there. If read pointer is reset to the address other than the upper limit, it will increase to the upper address and stop at there again.

Note

> If the software set the read pointer to the upper limit, the read pointer will not advance in this mode.

37.5.2 DMA Operation

When DMA is enabled, interrupt requests are not generated. DMA requests are generated instead. DMA done signal clears the DMA request.

The status register flags are still set and are cleared automatically when the DMA completes.

37.5.3 Resets

During reset the DAC is configured in the default mode. DAC is disabled.

37.5.4 Low Power Mode Operation

This section describes the wait mode and the stop mode operation of the DAC module.

37.5.4.1 Wait Mode Operation

In wait mode, the DAC will operate normally if enabled.

37.5.4.2 Stop Mode Operation

The DAC continues to operate in Normal Stop mode if enabled, the output voltage will hold the value before stop.

In Low Power Stop modes, the DAC is fully shut-down.

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.

Chapter 38
Voltage Reference (VREFV1)

38.1 Introduction

The VREFV1 Voltage Reference is intended to supply an accurate voltage output that can be trimmed in 0.5 mV steps. The VREFV1 can be used in applications to provide a reference voltage to external devices or used internally as a reference to analog peripherals such as the ADC, DAC, or CMP. The voltage reference has two operating modes that provide different levels of supply rejection and power consumption.

The following figure is a block diagram of the Voltage Reference.

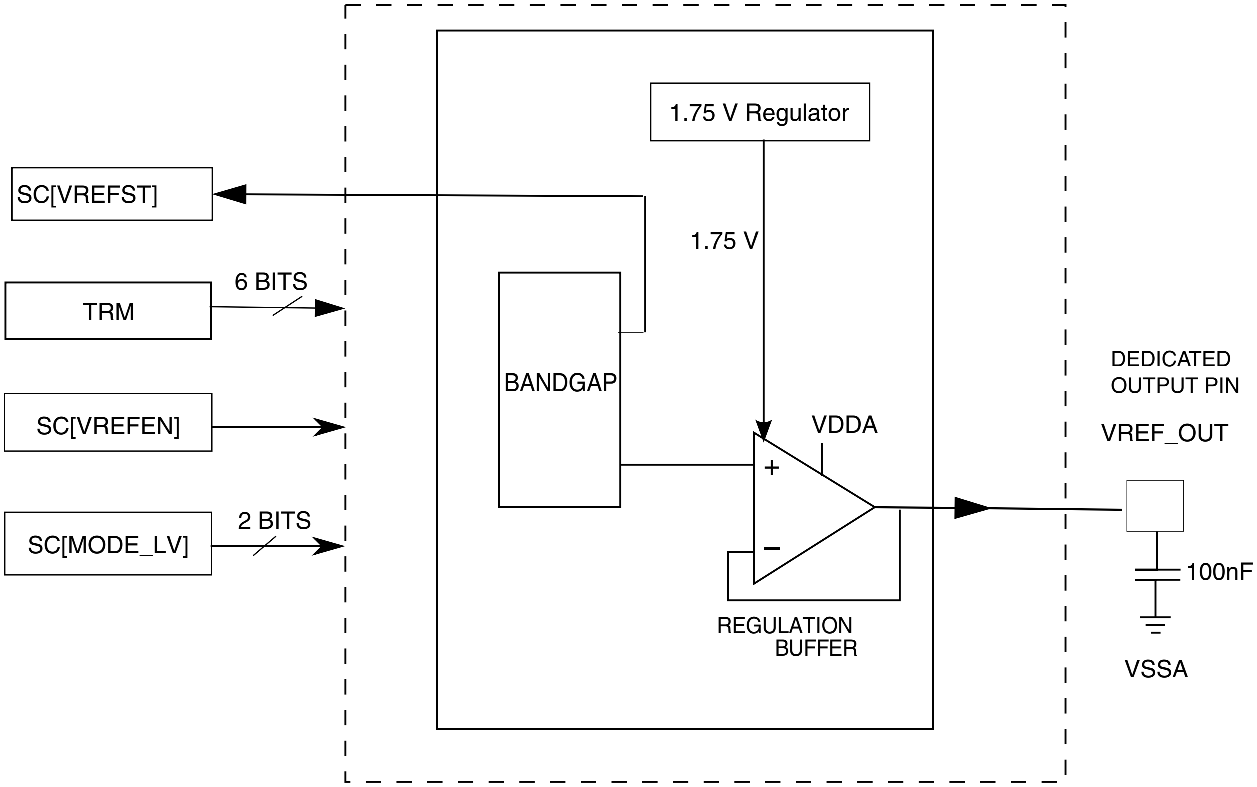


Figure 38-1. Voltage reference block diagram

38.1.1 Overview

The Voltage Reference provides a buffered reference voltage with high output current for use as an external reference. In addition, the buffered reference is available internally for use with on chip peripherals such as ADCs and DACs. Refer to the chip configuration chapter for a description of these options. The reference voltage is output on a dedicated output pin when the VREF is enabled. The Voltage Reference output can be trimmed with a resolution of 0.5mV by means of the TRM register TRIM[5:0] bitfield.

38.1.2 Features

The Voltage Reference has the following features:

- Programmable trim register with 0.5 mV steps, automatically loaded with factory trimmed value upon reset
- Programmable buffer mode selection:
  - Off
  - Bandgap enabled/standby (output buffer disabled)
  - Tight-regulation buffer mode (output buffer enabled)
- 1.2 V output at room temperature
- Dedicated output pin, VREF_OUT
- Load regulation in tight-regulation mode

38.1.3 Modes of Operation

The Voltage Reference continues normal operation in Run, Wait, and Stop modes. The Voltage Reference can also run in Very Low Power Run (VLPR), Very Low Power Wait (VLPW) and Very Low Power Stop (VLPS). The VREF regulator is not available in any Very Low Power modes and must be disabled (SC[REGEN]=0) before entering these modes. Note however that the accuracy of the output voltage will be reduced (by as much as several mVs) when the VREF regulator is not used.

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.

38.1.4 VREF Signal Descriptions

The following table shows the Voltage Reference signals properties.

Table 38-1. VREF Signal Descriptions

Signal	Description	I/O
VREF_OUT	Internally-generated Voltage Reference output	O

NOTE

- In Disable mode, the status of the VREF_OUT signal is high-impedence.

38.2 Memory Map and Register Definition

VREF memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_4000	VREF Trim Register (VREF_TRM)	8	R/W	Undefined	38.2.1/ 1016
4007_4001	VREF Status and Control Register (VREF_SC)	8	R/W	00h	38.2.2/ 1017

38.2.1 VREF Trim Register (VREF_TRM)

This register contains bits that contain the trim data for the Voltage Reference.

Address: VREF_TRM is 4007_4000h base + 0h offset = 4007_4000h

Bit	7	6	5	4	3	2	1	0
Read	Reserved	0	TRIM					
Write		▨						
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

VREF_TRM field descriptions

Field	Description
7 Reserved	This field is reserved. Upon reset this value is loaded with a factory trim value. This bit must always be written with the original reset value.
6 Reserved	This read-only field is reserved and always has the value zero.
5–0 TRIM	Trim bits Upon reset this value is loaded with a factory trim value. These bits change the resulting VREF by approximately ± 0.5 mV for each step. **NOTE:** Min = minimum and max = maximum voltage reference output. For minimum and maximum voltage reference output values, refer to the Data Sheet for this chip. 000000 Min .... 111111 Max

38.2.2 VREF Status and Control Register (VREF_SC)

This register contains the control bits used to enable the internal voltage reference and to select the VREF mode to be used.

Address: VREF_SC is 4007_4000h base + 1h offset = 4007_4001h

Bit	7	6	5	4	3	2	1	0
Read	VREFEN	REGEN	Reserved	0	0	VREFST	MODE_LV	
Write								
Reset	0	0	0	0	0	0	0	0

VREF_SC field descriptions

Field	Description
7 VREFEN	Internal Voltage Reference enable This bit is used to enable the bandgap reference within the Voltage Reference module. **NOTE:** After the VREF is enabled, turning off the clock to the VREF module via the corresponding clock gate register will not disable the VREF. VREF must be disabled via this VREFEN bit. 0 The module is disabled. 1 The module is enabled.
6 REGEN	Regulator enable This bit is used to enable the internal 1.75 V VREF regulator to produce a constant internal voltage supply in order to reduce the sensitivity to external supply noise and variation. The VREF regulator must not be enabled when entering VLPR, VLPW or VLPS modes. 0 Internal 1.75 V regulator is disabled. 1 Internal 1.75 V regulator is enabled.
5 Reserved	This field is reserved. This bit must always be written to zero.
4 Reserved	This read-only field is reserved and always has the value zero.
3 Reserved	This read-only field is reserved and always has the value zero.
2 VREFST	Internal Voltage Reference has settled This bit indicates that the bandgap reference within the Voltage Reference module has completed its startup and stabilization. 0 The bandgap is disabled or not ready. 1 The bandgap is ready.
1–0 MODE_LV	Buffer Mode selection These bits select the buffer modes for the Voltage Reference module.

Table continues on the next page...

VREF_SC field descriptions (continued)

Field	Description
	00 Bandgap on only, for stabilization and startup 01 Reserved 10 Tight-regulation buffer enabled 11 Reserved

38.3 Functional Description

The Voltage Reference is a bandgap buffer system. Unity gain amplifiers are used.

The VREF_OUT signal is available as an internal reference when it is enabled. A 100 nF capacitor must be connected between VREF_OUT and VSSA.

The following table shows all possible function configurations of the Voltage Reference.

Table 38-5. Voltage Reference function configurations

SC[VREFEN]	SC[MODE_LV]	Configuration	Functionality
0	X	Voltage Reference disabled	Off
1	00	Voltage Reference enabled, only the VREF bandgap is on	Startup and standby
1	01	Reserved	Reserved
1	10	Voltage Reference enabled, VREF bandgap and tight-regulation buffer on	VREF_OUT available for internal and external use. 100 nF capacitor is required.
1	11	Reserved	Reserved

38.3.1 Voltage Reference Disabled, SC[VREFEN] = 0

When SC[VREFEN] = 0, the Voltage Reference is disabled, all bandgap and buffers are disabled. The Voltage Reference is in off mode.

38.3.2 Voltage Reference Enabled, SC[VREFEN] = 1

When SC[VREFEN] = 1, the Voltage Reference is enabled, and different modes can be set by the SC[MODE_LV] bits.

38.3.2.1 SC[MODE_LV]=00

The internal bandgap is enabled to generate an accurate 1.2 V output that can be trimmed with the TRM register's TRIM[5:0] bitfield. The bandgap requires some time for startup and stabilization. SC[VREFST] can be monitored to determine if the stabilization and startup is complete.

The output buffer is disabled in this mode, and there is no buffered voltage output. The Voltage Reference is in standby mode. If this mode is first selected and the tight regulation buffer mode is subsequently enabled, there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet.

38.3.2.2 SC[MODE_LV] = 01

Reserved

38.3.2.3 SC[MODE_LV] = 10

The tight regulation buffer is enabled to generate a buffered 1.2 V voltage to VREF_OUT. If this mode is entered from the standby mode (SC[MODE_LV] = 00, SC[VREFEN] = 1) there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled then you must wait the longer of Tstup or until SC[VREFST] = 1.

38.3.2.4 SC[MODE_LV] = 11

Reserved

38.4 Initialization/Application Information

The Voltage Reference requires some time for startup and stabilization. After SC[VREFEN] = 1, SC[VREFST] can be monitored to determine if the stabilization and startup of the VREF bandgap is complete.

When the Voltage Reference is already enabled and stabilized, changing SC[MODE_LV] will not clear SC[VREFST] but there will be some startup time before the output voltage at the VREF_OUT pin has settled. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. Also, there will be some settling time when a step change of the load current is applied to the VREF_OUT pin.

When the 1.75V VREF regulator is disabled, the VREF_OUT voltage will be more sensitive to supply voltage variation. It is recommended to use this regulator to achieve optimum VREF_OUT performance.

Chapter 39
Programmable Delay Block (PDB)

39.1 Introduction

NOTE

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

The programmable delay block (PDB) provides controllable delays from either an internal or an external trigger, or a programmable interval tick, to the hardware trigger inputs of ADCs and/or generates the interval triggers to DACs, so that the precise timing between ADC conversions and/or DAC updates can be achieved. The PDB can optionally provides pulse outputs (Pulse-Out's) that are used as the sample window in the CMP block.

39.1.1 Features

- Up to 15 trigger input sources and software trigger source
- Up to eight configurable PDB channels for ADC hardware trigger
  - One PDB channel is associated with one ADC.
  - One trigger output for ADC hardware trigger and up to eight pre-trigger outputs for ADC trigger select per PDB channel
  - Trigger outputs can be enabled or disabled independently.
  - One 16-bit delay register per pre-trigger output
  - Optional bypass of the delay registers of the pre-trigger outputs
  - Operation in One-Shot or Continuous modes

  - Optional back-to-back mode operation, which enables the ADC conversions complete to trigger the next PDB channel
  - One programmable delay interrupt
  - One sequence error interrupt
  - One channel flag and one sequence error flag per pre-trigger
  - DMA support
- Up to eight DAC interval triggers
  - One interval trigger output per DAC
  - One 16-bit delay interval register per DAC trigger output
  - Optional bypass the delay interval trigger registers
  - Optional external triggers
- Up to eight pulse outputs (pulse-out's)
  - Pulse-out's can be enabled or disabled independently.
  - Programmable pulse width

NOTE

The number of PDB input and output triggers are chip-specific. Refer to the Chip Configuration information for details.

39.1.2 Implementation

In this chapter, the following letters refers to the number of output triggers.

- *N* — Total available number of PDB channels.
- *n* — PDB channel number, valid from 0 to *N*-1.
- *M* — Total available pre-trigger per PDB channel.
- *m* — Pre-trigger number, valid from 0 to *M*-1.
- *X* — Total number of DAC interval triggers.
- *x* — DAC interval trigger output number, valid from 0 to *X*-1.

- *Y* — Total number of Pulse-Out's.
- *y* — Pulse-Out number, valid value is 0 to *Y*-1.

NOTE

The number of module output triggers to core are chip-specific. For module to core output triggers implementation, refer to the Chip Configuration information.

39.1.3 Back-to-back Acknowledgement Connections

PDB back-to-back operation acknowledgment connections are chip-specific. For implementation, refer to the Chip Configuration information.

39.1.4 DAC External Trigger Input Connections

The implementation of DAC external trigger inputs is chip-specific. Refer to the Chip Configuration information for details.

39.1.5 Block Diagram

This diagram illustrates the major components of the PDB.

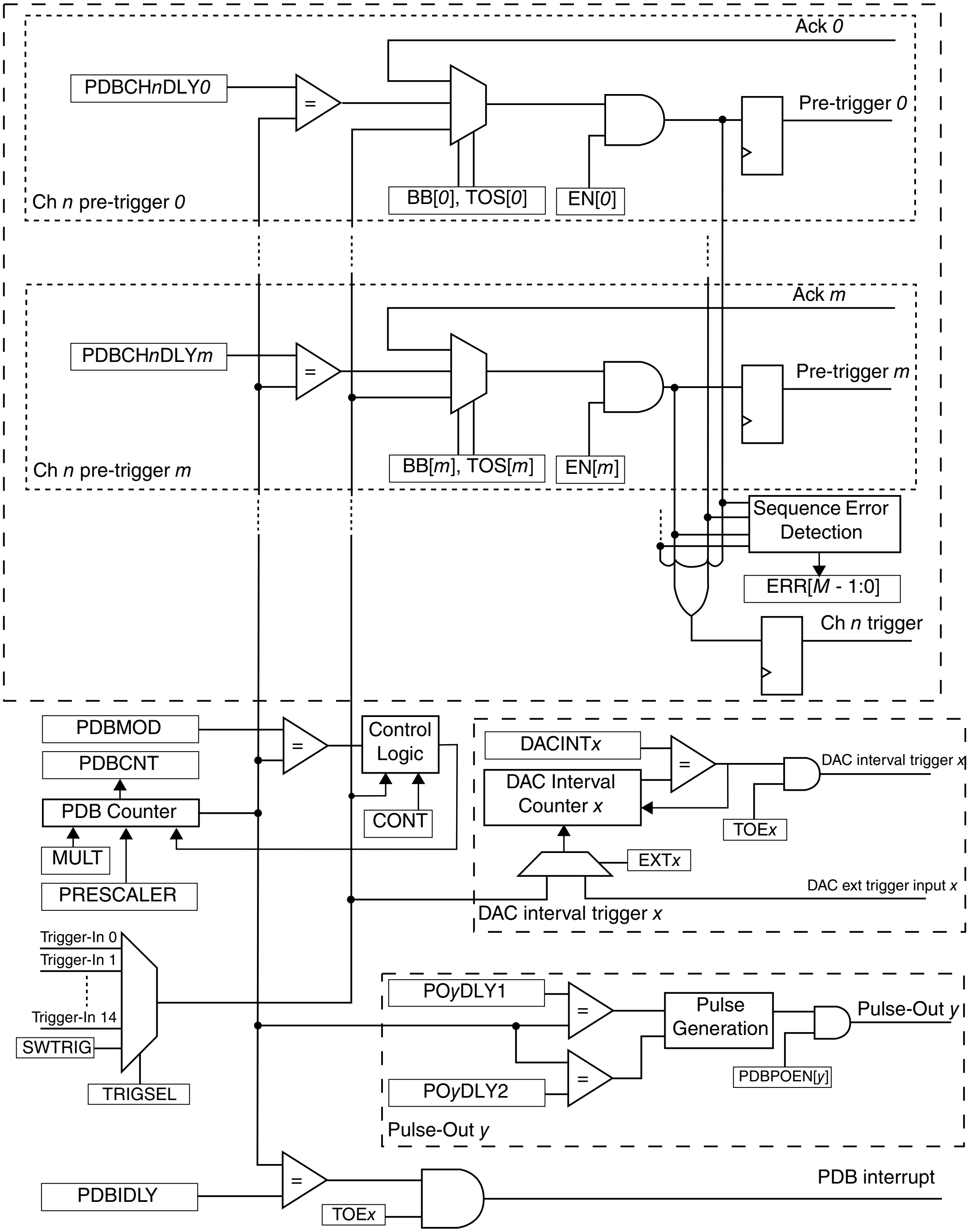


Figure 39-1. PDB Block Diagram

In this diagram, only one PDB channel *n*, one DAC interval trigger *x*, and one Pulse-Out *y* is shown. The PDB enable control logic and the sequence error interrupt logic is not shown.

39.1.6 Modes of Operation

PDB ADC trigger operates in the following modes.

Disabled: Counter is off, all pre-trigger and trigger outputs are low if PDB is not in back-to-back operation of Bypass mode.

Debug: Counter is paused when processor is in debug mode, the counter for dac trigger also paused in Debug mode.

Enabled One-Shot: Counter is enabled and restarted at count zero upon receiving a positive edge on the selected trigger input source or software trigger is selected and SC[SWTRIG] is written with 1. In each PDB channel, an enabled pre-trigger asserts once per trigger input event; the trigger output asserts whenever any of pre-triggers is asserted.

Enabled Continuous: Counter is enabled and restarted at count zero. The counter is rolled over to zero again when the count reaches the value specified in the modulus register, and the counting is restarted. This enables a continuous stream of pre-triggers/ trigger outputs as a result of a single trigger input event.

Enabled Bypassed: The pre-trigger and trigger outputs assert immediately after a positive edge on the selected trigger input source or software trigger is selected and SC[SWTRIG] is written with 1, that is the delay registers are bypassed. It is possible to bypass any one or more of the delay registers; therefore this mode can be used in conjunction with One-Shot or Continuous mode.

39.2 PDB Signal Descriptions

This table shows the detailed description of the external signal.

Table 39-1. PDB Signal Descriptions

Signal	Description	I/O
EXTRG	External trigger input source. If the PDB is enabled and external trigger input source is selected, a positive edge on the EXTRG signal resets and starts the counter.	I

39.3 Memory Map and Register Definition

PDB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_6000	Status and Control Register (PDB0_SC)	32	R/W	0000_0000h	39.3.1/ 1027
4003_6004	Modulus Register (PDB0_MOD)	32	R/W	0000_FFFFh	39.3.2/ 1030
4003_6008	Counter Register (PDB0_CNT)	32	R	0000_0000h	39.3.3/ 1030
4003_600C	Interrupt Delay Register (PDB0_IDLY)	32	R/W	0000_FFFFh	39.3.4/ 1031
4003_6010	Channel n Control Register 1 (PDB0_CH0C1)	32	R/W	0000_0000h	39.3.5/ 1031
4003_6014	Channel n Status Register (PDB0_CH0S)	32	w1c	0000_0000h	39.3.6/ 1032
4003_6018	Channel n Delay 0 Register (PDB0_CH0DLY0)	32	R/W	0000_0000h	39.3.7/ 1033
4003_601C	Channel n Delay 1 Register (PDB0_CH0DLY1)	32	R/W	0000_0000h	39.3.8/ 1034
4003_6038	Channel n Control Register 1 (PDB0_CH1C1)	32	R/W	0000_0000h	39.3.5/ 1031
4003_603C	Channel n Status Register (PDB0_CH1S)	32	w1c	0000_0000h	39.3.6/ 1032
4003_6040	Channel n Delay 0 Register (PDB0_CH1DLY0)	32	R/W	0000_0000h	39.3.7/ 1033
4003_6044	Channel n Delay 1 Register (PDB0_CH1DLY1)	32	R/W	0000_0000h	39.3.8/ 1034
4003_6060	Channel n Control Register 1 (PDB0_CH2C1)	32	R/W	0000_0000h	39.3.5/ 1031
4003_6064	Channel n Status Register (PDB0_CH2S)	32	w1c	0000_0000h	39.3.6/ 1032
4003_6068	Channel n Delay 0 Register (PDB0_CH2DLY0)	32	R/W	0000_0000h	39.3.7/ 1033
4003_606C	Channel n Delay 1 Register (PDB0_CH2DLY1)	32	R/W	0000_0000h	39.3.8/ 1034
4003_6088	Channel n Control Register 1 (PDB0_CH3C1)	32	R/W	0000_0000h	39.3.5/ 1031
4003_608C	Channel n Status Register (PDB0_CH3S)	32	w1c	0000_0000h	39.3.6/ 1032
4003_6090	Channel n Delay 0 Register (PDB0_CH3DLY0)	32	R/W	0000_0000h	39.3.7/ 1033
4003_6094	Channel n Delay 1 Register (PDB0_CH3DLY1)	32	R/W	0000_0000h	39.3.8/ 1034

Table continues on the next page...

PDB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_6150	DAC Interval Trigger n Control Register (PDB0_DACINTC0)	32	R/W	0000_0000h	39.3.9/ 1034
4003_6154	DAC Interval n Register (PDB0_DACINT0)	32	R/W	0000_0000h	39.3.10/ 1035
4003_6158	DAC Interval Trigger n Control Register (PDB0_DACINTC1)	32	R/W	0000_0000h	39.3.9/ 1034
4003_615C	DAC Interval n Register (PDB0_DACINT1)	32	R/W	0000_0000h	39.3.10/ 1035
4003_6190	Pulse-Out n Enable Register (PDB0_POEN)	32	R/W	0000_0000h	39.3.11/ 1035
4003_6194	Pulse-Out n Delay Register (PDB0_PO0DLY)	32	R/W	0000_0000h	39.3.12/ 1036
4003_6198	Pulse-Out n Delay Register (PDB0_PO1DLY)	32	R/W	0000_0000h	39.3.12/ 1036
4003_619C	Pulse-Out n Delay Register (PDB0_PO2DLY)	32	R/W	0000_0000h	39.3.12/ 1036
4003_61A0	Pulse-Out n Delay Register (PDB0_PO3DLY)	32	R/W	0000_0000h	39.3.12/ 1036

39.3.1 Status and Control Register (PDBx_SC)

Addresses: PDB0_SC is 4003_6000h base + 0h offset = 4003_6000h

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>R</td><td colspan="12">0</td><td colspan="2" rowspan="2">LDMOD</td><td rowspan="2">PDBEIE</td><td>0</td></tr>
<tr><td>W</td><td colspan="12"></td><td>SWTRIG</td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

<table>
<tr><td>Bit</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td rowspan="2">DMAEN</td><td colspan="3" rowspan="2">PRESCALER</td><td colspan="4" rowspan="2">TRGSEL</td><td rowspan="2">PDBEN</td><td rowspan="2">PDBIF</td><td rowspan="2">PDBIE</td><td>0</td><td colspan="2" rowspan="2">MULT</td><td rowspan="2">CONT</td><td rowspan="2">LDOK</td></tr>
<tr><td>W</td><td></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

PDBx_SC field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–18 LDMOD	Load Mode Select Selects the mode to load the MOD, IDLY, CH*n*DLY*m*, INT*x*, and PO*y*DLY registers, after 1 is written to LDOK. 00 The internal registers are loaded with the values from their buffers immediately after 1 is written to LDOK. 01 The internal registers are loaded with the values from their buffers when the PDB counter reaches the MOD register value after 1 is written to LDOK. 10 The internal registers are loaded with the values from their buffers when a trigger input event is detected after 1 is written to LDOK. 11 The internal registers are loaded with the values from their buffers when either the PDB counter reaches the MOD register value or a trigger input event is detected, after 1 is written to LDOK.
17 PDBEIE	PDB Sequence Error Interrupt Enable This bit enables the PDB sequence error interrupt. When this bit is set, any of the PDB channel sequence error flags generates a PDB sequence error interrupt. 0 PDB sequence error interrupt disabled. 1 PDB sequence error interrupt enabled.
16 SWTRIG	Software Trigger When PDB is enabled and the software trigger is selected as the trigger input source, writing 1 to this bit reset and restarts the counter. Writing 0 to this bit has no effect. Reading this bit results 0.
15 DMAEN	DMA Enable When DMA is enabled, the PDBIF flag generates a DMA request instead of an interrupt. 0 DMA disabled 1 DMA enabled
14–12 PRESCALER	Prescaler Divider Select 000 Counting uses the peripheral clock divided by multiplication factor selected by MULT. 001 Counting uses the peripheral clock divided by twice of the multiplication factor selected by MULT. 010 Counting uses the peripheral clock divided by four times of the multiplication factor selected by MULT. 011 Counting uses the peripheral clock divided by eight times of the multiplication factor selected by MULT. 100 Counting uses the peripheral clock divided by 16 times of the multiplication factor selected by MULT. 101 Counting uses the peripheral clock divided by 32 times of the multiplication factor selected by MULT. 110 Counting uses the peripheral clock divided by 64 times of the multiplication factor selected by MULT. 111 Counting uses the peripheral clock divided by 128 times of the multiplication factor selected by MULT.
11–8 TRGSEL	Trigger Input Source Select

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
	Selects the trigger input source for the PDB. The trigger input source can be internal or external (EXTRG pin), or the software trigger. 0000 Trigger-In 0 is selected 0001 Trigger-In 1 is selected 0010 Trigger-In 2 is selected 0011 Trigger-In 3 is selected 0100 Trigger-In 4 is selected 0101 Trigger-In 5 is selected 0110 Trigger-In 6 is selected 0111 Trigger-In 7 is selected 1000 Trigger-In 8 is selected 1001 Trigger-In 9 is selected 1010 Trigger-In 10 is selected 1011 Trigger-In 11 is selected 1100 Trigger-In 12 is selected 1101 Trigger-In 13 is selected 1110 Trigger-In 14 is selected 1111 Software trigger is selected
7 PDBEN	PDB Enable 0 PDB disabled. Counter is off. 1 PDB enabled
6 PDBIF	PDB Interrupt Flag This bit is set when the counter value is equal to the IDLY register. Writing zero clears this bit.
5 PDBIE	PDB Interrupt Enable. This bit enables the PDB interrupt. When this bit is set and DMAEN is cleared, PDBIF generates a PDB interrupt. 0 PDB interrupt disabled 1 PDB interrupt enabled
4 Reserved	This read-only field is reserved and always has the value zero.
3–2 MULT	Multiplication Factor Select for Prescaler This bit selects the multiplication factor of the prescaler divider for the counter clock. 00 Multiplication factor is 1 01 Multiplication factor is 10 10 Multiplication factor is 20 11 Multiplication factor is 40
1 CONT	Continuous Mode Enable This bit enables the PDB operation in Continuous mode. 0 PDB operation in One-Shot mode 1 PDB operation in Continuous mode

Table continues on the next page...

PDBx_SC field descriptions (continued)

Field	Description
0 LDOK	Load OK Writing 1 to this bit updates the internal registers of MOD, IDLY, CHnDLYm, DACINTx, and POyDLY with the values written to their buffers. The MOD, IDLY, CHnDLYm, DACINTx, and POyDLY will take effect according to the LDMOD. After 1 is written to LDOK bit, the values in the buffers of above registers are not effective and the buffers cannot be written until the values in buffers are loaded into their internal registers. LDOK can be written only when PDBEN is set or it can be written at the same time with PDBEN being written to 1. It is automatically cleared when the values in buffers are loaded into the internal registers or the PDBEN is cleared. Writing 0 to it has no effect.

39.3.2 Modulus Register (PDBx_MOD)

Addresses: PDB0_MOD is 4003_6000h base + 4h offset = 4003_6004h

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																MOD															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

PDBx_MOD field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 MOD	PDB Modulus. These bits specify the period of the counter. When the counter reaches this value, it will be reset back to zero. If the PDB is in Continuous mode, the count begins anew. Reading these bits returns the value of internal register that is effective for the current cycle of PDB.

39.3.3 Counter Register (PDBx_CNT)

Addresses: PDB0_CNT is 4003_6000h base + 8h offset = 4003_6008h

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																CNT															
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

PDBx_CNT field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 CNT	PDB Counter These read-only bits contain the current value of the counter.

39.3.4 Interrupt Delay Register (PDBx_IDLY)

Addresses: PDB0_IDLY is 4003_6000h base + Ch offset = 4003_600Ch

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																IDLY															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

PDBx_IDLY field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 IDLY	PDB Interrupt Delay These bits specify the delay value to schedule the PDB interrupt. It can be used to schedule an independent interrupt at some point in the PDB cycle. If enabled, a PDB interrupt is generated, when the counter is equal to the IDLY. Reading these bits returns the value of internal register that is effective for the current cycle of the PDB.

39.3.5 Channel n Control Register 1 (PDBx_CHC1)

Each PDB channel has one Control Register, CHnC1. The bits in this register control the functionality of each PDB channel operation.

Addresses: PDB0_CH0C1 is 4003_6000h base + 10h offset = 4003_6010h
PDB0_CH1C1 is 4003_6000h base + 38h offset = 4003_6038h
PDB0_CH2C1 is 4003_6000h base + 60h offset = 4003_6060h
PDB0_CH3C1 is 4003_6000h base + 88h offset = 4003_6088h

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								BB								TOS								EN							
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

PDBx_CH*n*C1 field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value zero.
23–16 BB	PDB Channel Pre-Trigger Back-to-Back Operation Enable These bits enable the PDB ADC pre-trigger operation as back-to-back mode. Only lower M pre-trigger bits are implemented in this MCU. Back-to-back operation enables the ADC conversions complete to trigger the next PDB channel pre-trigger and trigger output, so that the ADC conversions can be triggered on next set of configuration and results registers. Application code must only enable the back-to-back operation of the PDB pre-triggers at the leading of the back-to-back connection chain. 0 PDB channel's corresponding pre-trigger back-to-back operation disabled. 1 PDB channel's corresponding pre-trigger back-to-back operation enabled.
15–8 TOS	PDB Channel Pre-Trigger Output Select These bits select the PDB ADC pre-trigger outputs. Only lower M pre-trigger bits are implemented in this MCU. 0 PDB channel's corresponding pre-trigger is in bypassed mode. The pre-trigger asserts one peripheral clock cycle after a rising edge is detected on selected trigger input source or software trigger is selected and SWTRIG is written with 1. 1 PDB channel's corresponding pre-trigger asserts when the counter reaches the channel delay register plus one peripheral clock cycle after a rising edge is detected on selected trigger input source or software trigger is selected and SETRIG is written with 1.
7–0 EN	PDB Channel Pre-Trigger Enable These bits enable the PDB ADC pre-trigger outputs. Only lower M pre-trigger bits are implemented in this MCU. 0 PDB channel's corresponding pre-trigger disabled. 1 PDB channel's corresponding pre-trigger enabled.

39.3.6 Channel n Status Register (PDBx_CHS)

Addresses: PDB0_CH0S is 4003_6000h base + 14h offset = 4003_6014h
PDB0_CH1S is 4003_6000h base + 3Ch offset = 4003_603Ch
PDB0_CH2S is 4003_6000h base + 64h offset = 4003_6064h
PDB0_CH3S is 4003_6000h base + 8Ch offset = 4003_608Ch

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								CF								0								ERR							
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

PDBx_CH*n*S field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value zero.
23–16 CF	PDB Channel Flags The CF[*m*] bit is set when the PDB counter matches the CH*n*DLY*m*. Write 0 to clear these bits.
15–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 ERR	PDB Channel Sequence Error Flags Only the lower M bits are implemented in this MCU. 0 Sequence error not detected on PDB channel's corresponding pre-trigger. 1 Sequence error detected on PDB channel's corresponding pre-trigger. ADCn block can be triggered for a conversion by one pre-trigger from PDB channel *n*. When one conversion, which is triggered by one of the pre-triggers from PDB channel *n*, is in progress, new trigger from PDB channel's corresponding pre-trigger m cannot be accepted by ADCn, and ERR[m] is set. Writing 1's to clear the sequence error flags.

39.3.7 Channel n Delay 0 Register (PDBx_CHDLY0)

Addresses: PDB0_CH0DLY0 is 4003_6000h base + 18h offset = 4003_6018h
PDB0_CH1DLY0 is 4003_6000h base + 40h offset = 4003_6040h
PDB0_CH2DLY0 is 4003_6000h base + 68h offset = 4003_6068h
PDB0_CH3DLY0 is 4003_6000h base + 90h offset = 4003_6090h

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																DLY															
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

PDBx_CH*n*DLY0 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 DLY	PDB Channel Delay These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

39.3.8 Channel n Delay 1 Register (PDBx_CHDLY1)

Addresses: PDB0_CH0DLY1 is 4003_6000h base + 1Ch offset = 4003_601Ch
PDB0_CH1DLY1 is 4003_6000h base + 44h offset = 4003_6044h
PDB0_CH2DLY1 is 4003_6000h base + 6Ch offset = 4003_606Ch
PDB0_CH3DLY1 is 4003_6000h base + 94h offset = 4003_6094h

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																DLY															
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

PDBx_CH*n*DLY1 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 DLY	PDB Channel Delay These bits specify the delay value for the channel's corresponding pre-trigger. The pre-trigger asserts when the counter is equal to DLY. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

39.3.9 DAC Interval Trigger n Control Register (PDBx_DACINTC*n*)

Addresses: PDB0_DACINTC0 is 4003_6000h base + 150h offset = 4003_6150h
PDB0_DACINTC1 is 4003_6000h base + 158h offset = 4003_6158h

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																														EXT	TOE
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

PDBx_DACINTC*n* field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value zero.
1 EXT	DAC External Trigger Input Enable This bit enables the external trigger for DAC interval counter.

Table continues on the next page...

PDBx_DACINTC*n* field descriptions (continued)

Field	Description
	0 DAC external trigger input disabled. DAC interval counter is reset and started counting when a rising edge is detected on selected trigger input source or software trigger is selected and SWTRIG is written with 1. 1 DAC external trigger input enabled. DAC interval counter is bypassed and DAC external trigger input triggers the DAC interval trigger.
0 TOE	DAC Interval Trigger Enable This bit enables the DAC interval trigger. 0 DAC interval trigger disabled. 1 DAC interval trigger enabled.

39.3.10 DAC Interval n Register (PDBx_DACINT*n*)

Addresses: PDB0_DACINT0 is 4003_6000h base + 154h offset = 4003_6154h
PDB0_DACINT1 is 4003_6000h base + 15Ch offset = 4003_615Ch

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																INT															
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

PDBx_DACINT*n* field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 INT	DAC Interval These bits specify the interval value for DAC interval trigger. DAC interval trigger triggers DAC[1:0] update when the DAC interval counter is equal to the DACINT. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

39.3.11 Pulse-Out n Enable Register (PDBx_POEN)

Addresses: PDB0_POEN is 4003_6000h base + 190h offset = 4003_6190h

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																								POEN							
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

PDBx_POEN field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 POEN	PDB Pulse-Out Enable These bits enable the pulse output. Only lower Y bits are implemented in this MCU. 0 PDB Pulse-Out disabled 1 PDB Pulse-Out enabled

39.3.12 Pulse-Out n Delay Register (PDBx_PODLY)

Addresses: PDB0_PO0DLY is 4003_6000h base + 194h offset = 4003_6194h
PDB0_PO1DLY is 4003_6000h base + 198h offset = 4003_6198h
PDB0_PO2DLY is 4003_6000h base + 19Ch offset = 4003_619Ch
PDB0_PO3DLY is 4003_6000h base + 1A0h offset = 4003_61A0h

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	DLY1																DLY2															
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

PDBx_PO*n*DLY field descriptions

Field	Description
31–16 DLY1	PDB Pulse-Out Delay 1 These bits specify the delay 1 value for the PDB Pulse-Out. Pulse-Out goes high when the PDB counter is equal to the DLY1. Reading these bits returns the value of internal register that is effective for the current PDB cycle.
15–0 DLY2	PDB Pulse-Out Delay 2 These bits specify the delay 2 value for the PDB Pulse-Out. Pulse-Out goes low when the PDB counter is equal to the DLY2. Reading these bits returns the value of internal register that is effective for the current PDB cycle.

39.4 Functional Description

39.4.1 PDB Pre-trigger and Trigger Outputs

The PDB contains a counter whose output is compared against several different digital values. If the PDB is enabled, a trigger input event will reset the counter and make it start to count. A trigger input event is defined as a rising edge being detected on selected trigger input source or software trigger being selected and SC[SWTRIG] is written with 1. For each channel, delay *m* determines the time between assertion of the trigger input event to the point at which changes in the pre-trigger *m* output signal is initiated. The time is defined as:

- Trigger input event to pre-trigger *m* = (prescaler X multiplication factor X delay *m*) + 2 peripheral clock cycles
- Add one additional peripheral clock cycle to determine the time at which the channel trigger output change.

Each channel is associated with one ADC block. PDB channel n pre-trigger outputs 0 to *M* and trigger output is connected to ADC hardware trigger select and hardware trigger inputs. The pre-triggers are used to precondition the ADC block prior to the actual trigger. The ADC contains *M* sets of configuration and result registers, allowing it to operate in a ping-pong fashion, alternating conversions between *M* different analog sources. The pre-trigger outputs are used to specify which signal will be sampled next. When pre-trigger *m* is asserted, the ADC conversion is triggered with set *m* of the configuration and result registers.

The waveforms shown in the following diagram illuminate the pre-trigger and trigger outputs of PDB channel *n*. The delays can be independently set via the CH*n*DLY*m* registers. And the pre-triggers can be enabled or disabled in CH*n*C1[EN[*m*]].

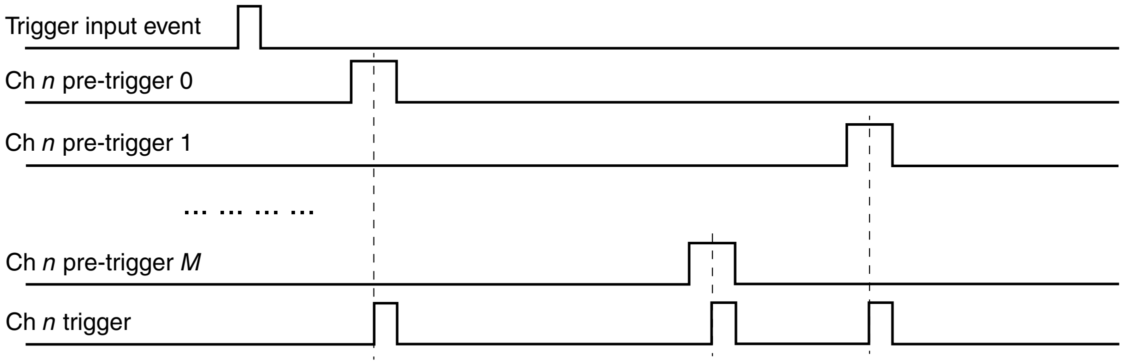


Figure 39-74. Pre-trigger and Trigger Outputs

The delay in CH*n*DLY*m* register can be optionally bypassed, if CH*n*C1[TOS[*m*]] is cleared. In this case, when the trigger input event occurs, the pre-trigger *m* is asserted after two peripheral clock cycles.

The PDB can be configured in back-to-back (B2B) operation. B2B operation enables the ADC conversions complete to trigger the next PDB channel pre-trigger and trigger outputs, so that the ADC conversions can be triggered on next set of configuration and results registers. When B2B is enabled by setting CH*n*C1[BB[*m*]], the delay *m* is ignored and the pre-trigger *m* is asserted two peripheral cycles after the acknowledgment *m* is received. The acknowledgment connections in this MCU is described in Back-to-back Acknowledgement Connections.

When an ADC conversion, which is triggered by one of the pre-triggers from PDB channel *n*, is in progress and ADC*n*SC1[COCO] is not set, a new trigger from PDB channel *n* pre-trigger *m* cannot be accepted by ADC*n*. Therefore every time when one PDB channel *n* pre-trigger and trigger output starts an ADC conversion, an internal lock associated with the corresponding pre-trigger is activated. The lock becomes inactive when the corresponding ADC*n*SC1[COCO] is set, or the corresponding PDB pre-trigger is disabled, or the PDB is disabled. The channel *n* trigger output is suppressed when any of the locks of the pre-triggers in channel *n* is active. If a new pre-trigger *m* asserts when there is active lock in the PDB channel *n*, a register flag bit, CH*n*S[ERR[*m*]], associated with the pre-trigger *m* is set. If SC[PDBEIE] is set, the sequence error interrupt is generated. Sequence error is typically happened because the delay *m* is set too short and the pre-trigger *m* asserts before the previous triggered ADC conversion is completed.

When the PDB counter reaches the value set in IDLY register, the SC[PDBIF] flag is set. A PDB interrupt can be generated if SC[PDBIE] is set and SC[DMAEN] is cleared. If SC[DMAEN] is set, PDB requests a DMA transfer when SC[PDBIF] is set.

The modulus value in MOD register, is used to reset the counter back to zero at the end of the count. If SC[CONT] bit is set, the counter will then resume a new count. Otherwise, the counter operation will cease until the next trigger input event occurs.

39.4.2 PDB Trigger Input Source Selection

The PDB has up to 15 trigger input sources, namely Trigger-In 0 to 14. They are connected to on-chip or off-chip event sources. The PDB can be triggered by software through the SC[SWTRIG]. SC[TRIGSEL] bits select the active trigger input source or software trigger.

For the trigger input sources implemented in this MCU, refer to Chip Configuration information.

39.4.3 DAC Interval Trigger Outputs

PDB can generate the interval triggers for DACs to update their outputs periodically. DAC interval counter *x* is reset and started when a trigger input event occurs if DACINTC*x*[EXT] is cleared. When the interval counter *x* is equal to the value set in DACINT*x* register, the DAC interval trigger *x* output generates a pulse of one peripheral clock cycle width to update the DAC*x*. If DACINTC*x*[EXT] is set, the DAC interval counter is bypassed and the interval trigger output *x* generates a pulse following the detection of a rising edge on the DAC external trigger input. The counter and interval trigger can be disabled by clearing the DACINTC*x*[TOE].

DAC interval counters are also reset when the PDB counter reaches the MOD register value, therefore when the PDB counter rolls over to zero, the DAC interval counters starts anew.

Together, the DAC interval trigger pulse and the ADC pre-trigger/trigger pulses allow precise timing of DAC updates and ADC measurements. This is outlined in the typical use case described in the following diagram.

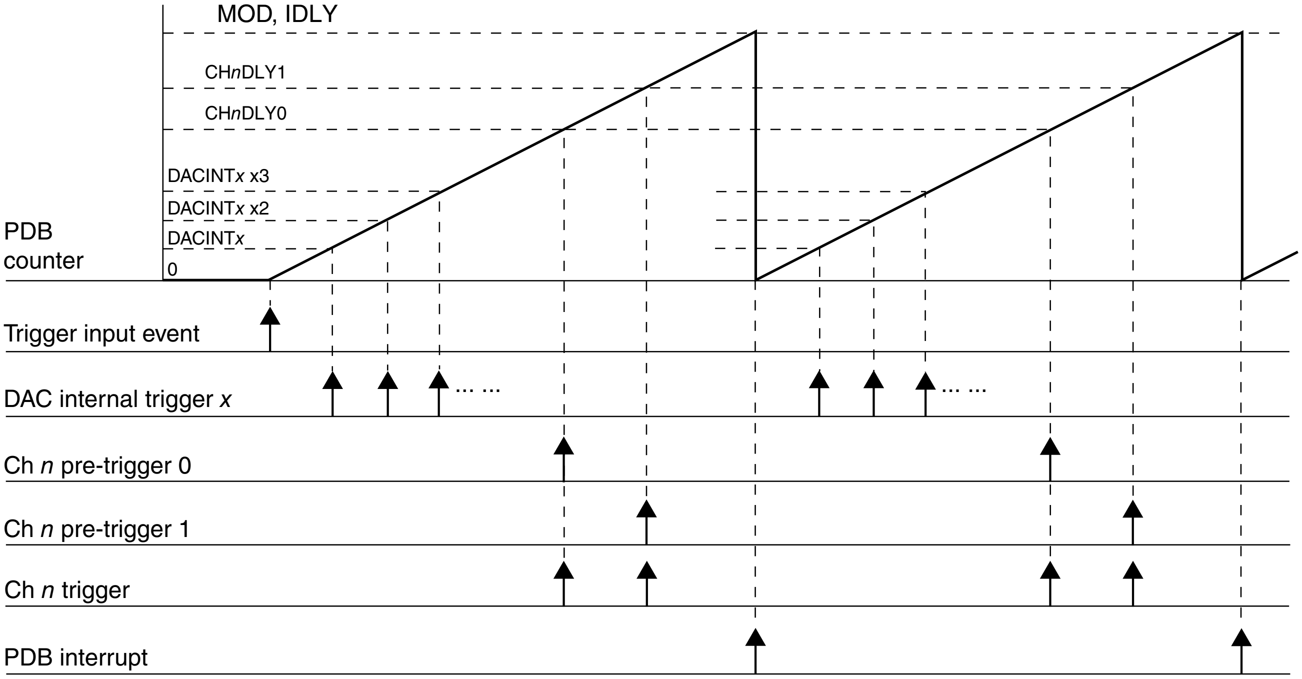


Figure 39-75. PDB ADC Triggers and DAC Interval Triggers Use Case

NOTE

> Because the DAC interval counters share the prescaler with PDB counter, PDB must be enabled if the DAC interval trigger outputs are used in the applications.

39.4.4 Pulse-Out's

PDB can generate pulse outputs of configurable width. When PDB counter reaches the value set in PO*y*DLY[DLY1], the Pulse-Out goes high; when the counter reaches PO*y*DLY[DLY2], it goes low. PO*y*DLY[DLY2] can be set either greater or less than PO*y*DLY[DLY1].

Because the PDB counter is shared by both ADC pre-trigger/trigger outputs and Pulse-Out generation, they have the same time base.

The pulse-out connections implemented in this MCU are described in the device's Chip Configuration details.

39.4.5 Updating the Delay Registers

The following registers control the timing of the PDB operation; and in some of the applications, they may need to become effective at the same time.

- PDB Modulus Register (MOD)
- PDB Interrupt Delay Register (IDLY)
- PDB Channel *n* Delay *m* Register (CH*n*DLY*m*)
- DAC Interval *x* Register (DACINT*x*)
- PDB Pulse-Out *y* Delay Register (PO*y*DLY)

The internal registers of them are buffered and any values written to them are written first to their buffers. The circumstances that cause their internal registers to be updated with the values from the buffers are summarized as below table.

Table 39-76. Circumstances of Update to the Delay Registers

SC[LDMOD]	Update to the Delay Registers
00	The internal registers are loaded with the values from their buffers immediately after 1 is written to SC[LDOK].
01	The PDB counter reaches the MOD register value after 1 is written to SC[LDOK].
10	A trigger input event is detected after 1 is written to SC[LDOK].
11	Either the PDB counter reaches the MOD register value, or a trigger input event is detected, after 1 is written to SC[LDOK].

After 1 is written to SC[LDOK], the buffers cannot be written until the values in buffers are loaded into their internal registers. SC[LDOK] is self-cleared when the internal registers are loaded, so the application code can read it to determine the updates of the internal registers.

The following diagrams show the cases of the internal registers being updated with SC[LDMOD] is 00 and x1.

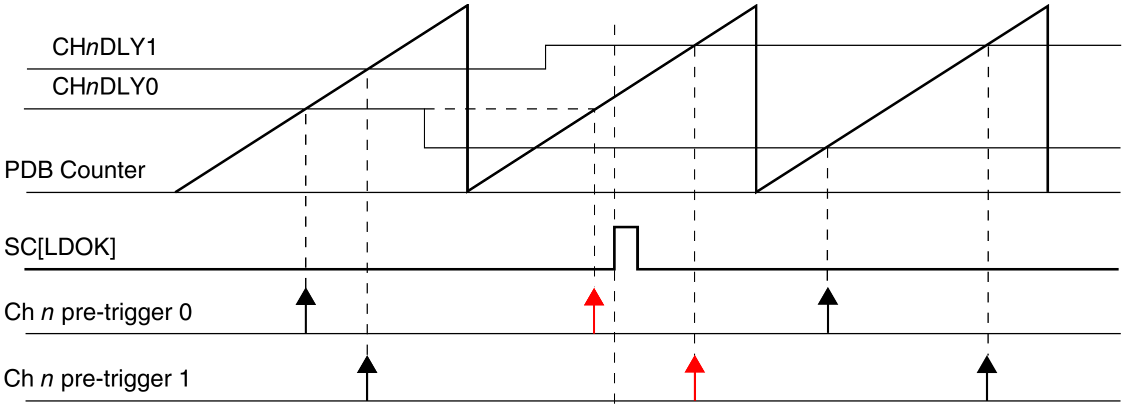


Figure 39-76. Registers Update with SC[LDMOD] = 00

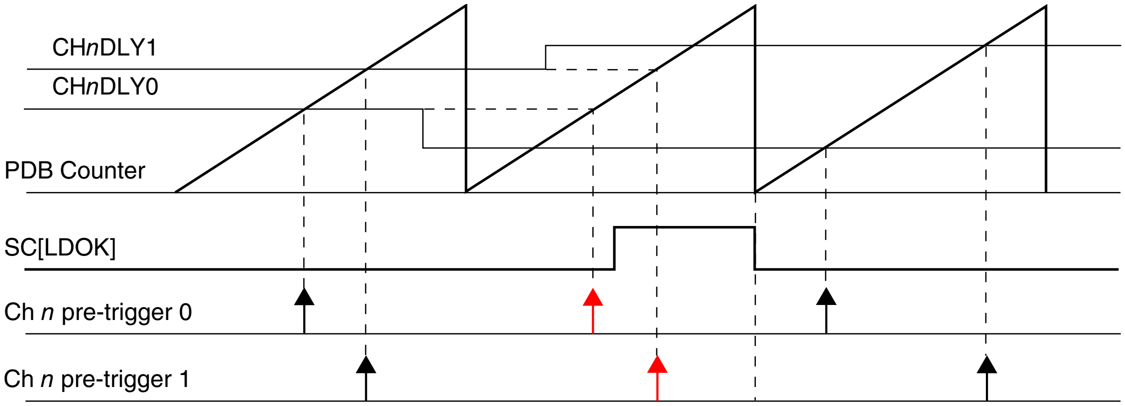


Figure 39-77. Registers Update with SC[LDMOD] = x1

39.4.6 Interrupts

PDB can generate two interrupts, PDB interrupt and PDB sequence error interrupt. The following table summarizes the interrupts.

Table 39-77. PDB Interrupt Summary

Interrupt	Flags	Enable Bit
PDB Interrupt	SC[PDBIF]	SC[PDBIE] = 1 and SC[DMAEN] = 0
PDB Sequence Error Interrupt	CH*n*S[ERR*m*]	SC[PDBEIE] = 1

39.4.7 DMA

If SC[DMAEN] is set, PDB can generate DMA transfer request when SC[PDBIF] is set. When DMA is enabled, the PDB interrupt will not be issued.

39.5 Application Information

39.5.1 Impact of Using the Prescaler and Multiplication Factor on Timing Resolution

Use of prescaler and multiplication factor greater than 1 limits the count/delay accuracy in terms of peripheral clock cycles (to the modulus of the prescaler X multiplication factor). If the multiplication factor is set to 1 and the prescaler is set to 2 then the only values of total peripheral clocks that can be detected are even values; if prescaler is set to 4 then the only values of total peripheral clocks that can be decoded as detected are mod(4) and so forth. If the applications need a really long delay value and use 128, then the resolution would be limited to 128 peripheral clock cycles.

Therefore, use the lowest possible prescaler and multiplication factor for a given application.

Chapter 40
FlexTimer (FTM)

40.1 Introduction

NOTE

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

The FlexTimer Module (FTM) is a two to eight channel timer which supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter.

40.1.1 FlexTimer Philosophy

The FlexTimer is built upon a very simple timer (HCS08 Timer PWM Module – TPM) used for many years on Freescales 8 bit microcontrollers. The FlexTimer extends the functionality to meet the demands of motor control, digital lighting solutions and power conversion yet providing low cost and backwards compatibility with the TPM module.

Several key enhancements are made; signed up counter, deadtime insertion hardware, fault control inputs, enhanced triggering functionality and initialization and polarity control.

All the features common with the TPM module have fully backwards compatible register assignments and the FlexTimer can use code on the same core platform without change to perform the same functions.

Motor control and power conversion features have been added through a dedicated set of registers and defaults turn off all new features. The new features such as hardware deadtime insertion, polarity, fault control and output forcing and masking greatly reduce loading on the execution software and are usually each controlled by a group of registers.

FlexTimer input triggers can be from comparators, ADC or other sub modules to initiate timer functions automatically. These triggers can be linked in a variety of ways during integration of the sub modules so please note carefully the options available for used FlexTimer configuration.

Several FlexTimers may be synchronized to provide a larger timer with their counters incrementing in unison, assuming the initialization, the input clocks, the initial and final counting values are the same in each FlexTimer.

All main user access registers are buffered to ease the load on the executing software. A number of trigger options exist to determine which registers are updated with this user defined data.

40.1.2 Features

The FTM features include:

- FTM source clock is selectable
  - Source clock can be the system clock, the fixed frequency clock, or an external clock
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than the system clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- FTM has a 16-bit counter
  - It can be a free-running counter or a counter with initial and final value
  - The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- In input capture mode
  - the capture can occur on rising edges, falling edges or both edges
  - an input filter can be selected for some channels
- In output compare mode the output signal can be set, cleared, or toggled on match
- All channels can be configured for center-aligned PWM mode

- Each pair of channels can be combined to generate a PWM signal (with independent control of both edges of PWM signal)
- The FTM channels can operate as pairs with equal outputs, pairs with complementary outputs, or independent channels (with independent outputs)
- The deadtime insertion is available for each complementary pair
- Generation of triggers (match trigger)
- Software control of PWM outputs
- Up to 4 fault inputs for global fault control
- The polarity of each channel is configurable
- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- The generation of an interrupt when the fault condition is detected
- Synchronized loading of write buffered FTM registers
- Write protection for critical registers
- Backwards compatible with TPM
- Testing of input captures for a stuck at zero and one conditions
- Dual edge capture for pulse and period width measurement
- Quadrature decoder with input filters, relative position counting and interrupt on position count or capture of position count on external event

40.1.3 Modes of Operation

When the MCU is in an active BDM mode, the FTM temporarily suspends all counting until the MCU returns to normal user operating mode. During stop mode, all FTM input clocks are stopped, so the FTM is effectively disabled until clocks resume. During wait mode, the FTM continues to operate normally. If the FTM does not need to produce a real time reference or provide the interrupt sources needed to wake the MCU from wait mode, the power can then be saved by disabling FTM functions before entering wait mode.

40.1.4 Block Diagram

The FTM uses one input/output (I/O) pin per channel, CHn (FTM channel (n)) where n is the channel number (0–7).

The following figure shows the FTM structure. The central component of the FTM is the 16-bit counter with programmable initial and final values and its counting can be up or up-down.

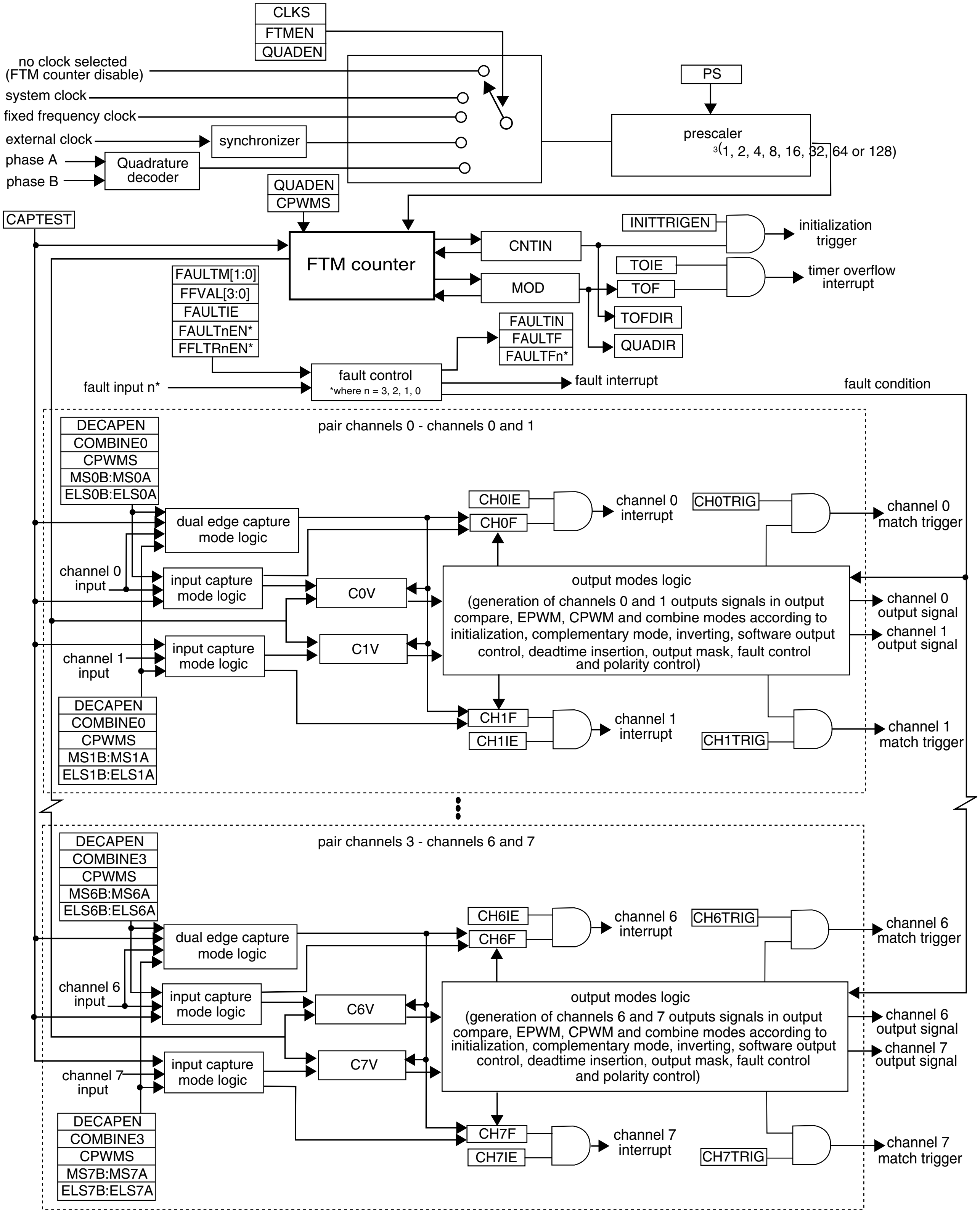


Figure 40-1. FTM Block Diagram

40.2 FTM Signal Descriptions

Table 40-1 shows the user-accessible signals for the FTM.

Table 40-1. FTM Signal Descriptions

Signal	Description	I/O
EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I
CHn	FTM channel (n), where n can be 7-0	I/O
FAULTj	Fault input (j), where j can be 3-0	I
PHA	Quadrature decoder phase A input. Input pin associated with quadrature decoder phase A.	I
PHB	Quadrature decoder phase B input. Input pin associated with quadrature decoder phase B.	I

40.2.1 EXTCLK — FTM External Clock

The external clock input signal is used as the FTM counter clock if selected by CLKS[1:0] bits in the SC register. This clock signal must not exceed 1/4 of system clock frequency. The FTM counter prescaler selection and settings are also used when an external clock is selected.

40.2.2 CHn — FTM Channel (n) I/O Pin

Each FTM 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.

40.2.3 FAULTj — FTM Fault Input

The fault input signals are used to control the CHn channel output state. If a fault is detected, the FAULTj signal is asserted and the channel output is put in a safe state. The behavior of the fault logic is defined by the FAULTM[1:0] control bits in the MODE register and FAULTEN bit in the COMBINEm register. Note that each FAULTj input may affect all channels selectively since FAULTM[1:0] and FAULTEN control bits are defined for each pair of channels. Since there are several FAULTj inputs, maximum of 4 for the FTM module, each one of these inputs is activated by the FAULTjEN bit in the FLTCTRL register.

40.2.4 PHA — FTM Quadrature Decoder Phase A Input

The quadrature decoder phase A input is used as the quadrature decoder mode is selected. The phase A input signal is one of the signals that control the FTM counter increment or decrement in the quadrature decoder mode (Quadrature Decoder Mode).

40.2.5 PHB — FTM Quadrature Decoder Phase B Input

The quadrature decoder phase B input is used as the quadrature decoder mode is selected. The phase B input signal is one of the signals that control the FTM counter increment or decrement in the quadrature decoder mode (Quadrature Decoder Mode).

40.3 Memory Map and Register Definition

This section provides a detailed description of all FTM registers.

40.3.1 Module Memory Map

This section presents a high-level summary of the FTM registers and how they are mapped.

The first set has the original TPM registers.

The second set has the FTM specific registers. Any second set registers (or bits within these registers) that are used by an unavailable function in the FTM configuration remain in the memory map and in the reset value, so they have no active function.

Note

> Do not write to the FTM specific registers (second set registers) when FTMEN = 0.

40.3.2 Register Descriptions

This section consists of register descriptions in address order.

Accesses to reserved addresses result in transfer errors. Registers for absent channels are considered reserved.

FTM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8000	Status and Control (FTM0_SC)	32	R/W	0000_0000h	40.3.3/ 1058
4003_8004	Counter (FTM0_CNT)	32	R/W	0000_0000h	40.3.4/ 1059
4003_8008	Modulo (FTM0_MOD)	32	R/W	0000_0000h	40.3.5/ 1060
4003_800C	Channel (n) Status and Control (FTM0_C0SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_8010	Channel (n) Value (FTM0_C0V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_8014	Channel (n) Status and Control (FTM0_C1SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_8018	Channel (n) Value (FTM0_C1V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_801C	Channel (n) Status and Control (FTM0_C2SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_8020	Channel (n) Value (FTM0_C2V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_8024	Channel (n) Status and Control (FTM0_C3SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_8028	Channel (n) Value (FTM0_C3V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_802C	Channel (n) Status and Control (FTM0_C4SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_8030	Channel (n) Value (FTM0_C4V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_8034	Channel (n) Status and Control (FTM0_C5SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_8038	Channel (n) Value (FTM0_C5V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_803C	Channel (n) Status and Control (FTM0_C6SC)	32	R/W	0000_0000h	40.3.6/ 1061

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8040	Channel (n) Value (FTM0_C6V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_8044	Channel (n) Status and Control (FTM0_C7SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_8048	Channel (n) Value (FTM0_C7V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_804C	Counter Initial Value (FTM0_CNTIN)	32	R/W	0000_0000h	40.3.8/ 1065
4003_8050	Capture and Compare Status (FTM0_STATUS)	32	R/W	0000_0000h	40.3.9/ 1065
4003_8054	Features Mode Selection (FTM0_MODE)	32	R/W	0000_0004h	40.3.10/ 1068
4003_8058	Synchronization (FTM0_SYNC)	32	R/W	0000_0000h	40.3.11/ 1069
4003_805C	Initial State for Channels Output (FTM0_OUTINIT)	32	R/W	0000_0000h	40.3.12/ 1072
4003_8060	Output Mask (FTM0_OUTMASK)	32	R/W	0000_0000h	40.3.13/ 1073
4003_8064	Function for Linked Channels (FTM0_COMBINE)	32	R/W	0000_0000h	40.3.14/ 1075
4003_8068	Deadtime Insertion Control (FTM0_DEADTIME)	32	R/W	0000_0000h	40.3.15/ 1080
4003_806C	FTM External Trigger (FTM0_EXTTRIG)	32	R/W	0000_0000h	40.3.16/ 1081
4003_8070	Channels Polarity (FTM0_POL)	32	R/W	0000_0000h	40.3.17/ 1083
4003_8074	Fault Mode Status (FTM0_FMS)	32	R/W	0000_0000h	40.3.18/ 1085
4003_8078	Input Capture Filter Control (FTM0_FILTER)	32	R/W	0000_0000h	40.3.19/ 1087
4003_807C	Fault Control (FTM0_FLTCTRL)	32	R/W	0000_0000h	40.3.20/ 1089
4003_8080	Quadrature Decoder Control and Status (FTM0_QDCTRL)	32	R/W	0000_0000h	40.3.21/ 1091
4003_8084	Configuration (FTM0_CONF)	32	R/W	0000_0000h	40.3.22/ 1093
4003_8088	FTM Fault Input Polarity (FTM0_FLTPOL)	32	R/W	0000_0000h	40.3.23/ 1094
4003_808C	Synchronization Configuration (FTM0_SYNCONF)	32	R/W	0000_0000h	40.3.24/ 1096

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_8090	FTM Inverting Control (FTM0_INVCTRL)	32	R/W	0000_0000h	40.3.25/ 1098
4003_8094	FTM Software Output Control (FTM0_SWOCTRL)	32	R/W	0000_0000h	40.3.26/ 1099
4003_8098	FTM PWM Load (FTM0_PWMLOAD)	32	R/W	0000_0000h	40.3.27/ 1102
4003_9000	Status and Control (FTM1_SC)	32	R/W	0000_0000h	40.3.3/ 1058
4003_9004	Counter (FTM1_CNT)	32	R/W	0000_0000h	40.3.4/ 1059
4003_9008	Modulo (FTM1_MOD)	32	R/W	0000_0000h	40.3.5/ 1060
4003_900C	Channel (n) Status and Control (FTM1_C0SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9010	Channel (n) Value (FTM1_C0V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_9014	Channel (n) Status and Control (FTM1_C1SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9018	Channel (n) Value (FTM1_C1V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_901C	Channel (n) Status and Control (FTM1_C2SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9020	Channel (n) Value (FTM1_C2V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_9024	Channel (n) Status and Control (FTM1_C3SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9028	Channel (n) Value (FTM1_C3V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_902C	Channel (n) Status and Control (FTM1_C4SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9030	Channel (n) Value (FTM1_C4V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_9034	Channel (n) Status and Control (FTM1_C5SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9038	Channel (n) Value (FTM1_C5V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_903C	Channel (n) Status and Control (FTM1_C6SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9040	Channel (n) Value (FTM1_C6V)	32	R/W	0000_0000h	40.3.7/ 1064

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_9044	Channel (n) Status and Control (FTM1_C7SC)	32	R/W	0000_0000h	40.3.6/ 1061
4003_9048	Channel (n) Value (FTM1_C7V)	32	R/W	0000_0000h	40.3.7/ 1064
4003_904C	Counter Initial Value (FTM1_CNTIN)	32	R/W	0000_0000h	40.3.8/ 1065
4003_9050	Capture and Compare Status (FTM1_STATUS)	32	R/W	0000_0000h	40.3.9/ 1065
4003_9054	Features Mode Selection (FTM1_MODE)	32	R/W	0000_0004h	40.3.10/ 1068
4003_9058	Synchronization (FTM1_SYNC)	32	R/W	0000_0000h	40.3.11/ 1069
4003_905C	Initial State for Channels Output (FTM1_OUTINIT)	32	R/W	0000_0000h	40.3.12/ 1072
4003_9060	Output Mask (FTM1_OUTMASK)	32	R/W	0000_0000h	40.3.13/ 1073
4003_9064	Function for Linked Channels (FTM1_COMBINE)	32	R/W	0000_0000h	40.3.14/ 1075
4003_9068	Deadtime Insertion Control (FTM1_DEADTIME)	32	R/W	0000_0000h	40.3.15/ 1080
4003_906C	FTM External Trigger (FTM1_EXTTRIG)	32	R/W	0000_0000h	40.3.16/ 1081
4003_9070	Channels Polarity (FTM1_POL)	32	R/W	0000_0000h	40.3.17/ 1083
4003_9074	Fault Mode Status (FTM1_FMS)	32	R/W	0000_0000h	40.3.18/ 1085
4003_9078	Input Capture Filter Control (FTM1_FILTER)	32	R/W	0000_0000h	40.3.19/ 1087
4003_907C	Fault Control (FTM1_FLTCTRL)	32	R/W	0000_0000h	40.3.20/ 1089
4003_9080	Quadrature Decoder Control and Status (FTM1_QDCTRL)	32	R/W	0000_0000h	40.3.21/ 1091
4003_9084	Configuration (FTM1_CONF)	32	R/W	0000_0000h	40.3.22/ 1093
4003_9088	FTM Fault Input Polarity (FTM1_FLTPOL)	32	R/W	0000_0000h	40.3.23/ 1094
4003_908C	Synchronization Configuration (FTM1_SYNCONF)	32	R/W	0000_0000h	40.3.24/ 1096
4003_9090	FTM Inverting Control (FTM1_INVCTRL)	32	R/W	0000_0000h	40.3.25/ 1098

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_9094	FTM Software Output Control (FTM1_SWOCTRL)	32	R/W	0000_0000h	40.3.26/ 1099
4003_9098	FTM PWM Load (FTM1_PWMLOAD)	32	R/W	0000_0000h	40.3.27/ 1102
400B_8000	Status and Control (FTM2_SC)	32	R/W	0000_0000h	40.3.3/ 1058
400B_8004	Counter (FTM2_CNT)	32	R/W	0000_0000h	40.3.4/ 1059
400B_8008	Modulo (FTM2_MOD)	32	R/W	0000_0000h	40.3.5/ 1060
400B_800C	Channel (n) Status and Control (FTM2_C0SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_8010	Channel (n) Value (FTM2_C0V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_8014	Channel (n) Status and Control (FTM2_C1SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_8018	Channel (n) Value (FTM2_C1V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_801C	Channel (n) Status and Control (FTM2_C2SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_8020	Channel (n) Value (FTM2_C2V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_8024	Channel (n) Status and Control (FTM2_C3SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_8028	Channel (n) Value (FTM2_C3V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_802C	Channel (n) Status and Control (FTM2_C4SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_8030	Channel (n) Value (FTM2_C4V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_8034	Channel (n) Status and Control (FTM2_C5SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_8038	Channel (n) Value (FTM2_C5V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_803C	Channel (n) Status and Control (FTM2_C6SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_8040	Channel (n) Value (FTM2_C6V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_8044	Channel (n) Status and Control (FTM2_C7SC)	32	R/W	0000_0000h	40.3.6/ 1061

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400B_8048	Channel (n) Value (FTM2_C7V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_804C	Counter Initial Value (FTM2_CNTIN)	32	R/W	0000_0000h	40.3.8/ 1065
400B_8050	Capture and Compare Status (FTM2_STATUS)	32	R/W	0000_0000h	40.3.9/ 1065
400B_8054	Features Mode Selection (FTM2_MODE)	32	R/W	0000_0004h	40.3.10/ 1068
400B_8058	Synchronization (FTM2_SYNC)	32	R/W	0000_0000h	40.3.11/ 1069
400B_805C	Initial State for Channels Output (FTM2_OUTINIT)	32	R/W	0000_0000h	40.3.12/ 1072
400B_8060	Output Mask (FTM2_OUTMASK)	32	R/W	0000_0000h	40.3.13/ 1073
400B_8064	Function for Linked Channels (FTM2_COMBINE)	32	R/W	0000_0000h	40.3.14/ 1075
400B_8068	Deadtime Insertion Control (FTM2_DEADTIME)	32	R/W	0000_0000h	40.3.15/ 1080
400B_806C	FTM External Trigger (FTM2_EXTTRIG)	32	R/W	0000_0000h	40.3.16/ 1081
400B_8070	Channels Polarity (FTM2_POL)	32	R/W	0000_0000h	40.3.17/ 1083
400B_8074	Fault Mode Status (FTM2_FMS)	32	R/W	0000_0000h	40.3.18/ 1085
400B_8078	Input Capture Filter Control (FTM2_FILTER)	32	R/W	0000_0000h	40.3.19/ 1087
400B_807C	Fault Control (FTM2_FLTCTRL)	32	R/W	0000_0000h	40.3.20/ 1089
400B_8080	Quadrature Decoder Control and Status (FTM2_QDCTRL)	32	R/W	0000_0000h	40.3.21/ 1091
400B_8084	Configuration (FTM2_CONF)	32	R/W	0000_0000h	40.3.22/ 1093
400B_8088	FTM Fault Input Polarity (FTM2_FLTPOL)	32	R/W	0000_0000h	40.3.23/ 1094
400B_808C	Synchronization Configuration (FTM2_SYNCONF)	32	R/W	0000_0000h	40.3.24/ 1096
400B_8090	FTM Inverting Control (FTM2_INVCTRL)	32	R/W	0000_0000h	40.3.25/ 1098
400B_8094	FTM Software Output Control (FTM2_SWOCTRL)	32	R/W	0000_0000h	40.3.26/ 1099

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400B_8098	FTM PWM Load (FTM2_PWMLOAD)	32	R/W	0000_0000h	40.3.27/ 1102
400B_9000	Status and Control (FTM3_SC)	32	R/W	0000_0000h	40.3.3/ 1058
400B_9004	Counter (FTM3_CNT)	32	R/W	0000_0000h	40.3.4/ 1059
400B_9008	Modulo (FTM3_MOD)	32	R/W	0000_0000h	40.3.5/ 1060
400B_900C	Channel (n) Status and Control (FTM3_C0SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9010	Channel (n) Value (FTM3_C0V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_9014	Channel (n) Status and Control (FTM3_C1SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9018	Channel (n) Value (FTM3_C1V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_901C	Channel (n) Status and Control (FTM3_C2SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9020	Channel (n) Value (FTM3_C2V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_9024	Channel (n) Status and Control (FTM3_C3SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9028	Channel (n) Value (FTM3_C3V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_902C	Channel (n) Status and Control (FTM3_C4SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9030	Channel (n) Value (FTM3_C4V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_9034	Channel (n) Status and Control (FTM3_C5SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9038	Channel (n) Value (FTM3_C5V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_903C	Channel (n) Status and Control (FTM3_C6SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9040	Channel (n) Value (FTM3_C6V)	32	R/W	0000_0000h	40.3.7/ 1064
400B_9044	Channel (n) Status and Control (FTM3_C7SC)	32	R/W	0000_0000h	40.3.6/ 1061
400B_9048	Channel (n) Value (FTM3_C7V)	32	R/W	0000_0000h	40.3.7/ 1064

Table continues on the next page...

FTM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400B_904C	Counter Initial Value (FTM3_CNTIN)	32	R/W	0000_0000h	40.3.8/ 1065
400B_9050	Capture and Compare Status (FTM3_STATUS)	32	R/W	0000_0000h	40.3.9/ 1065
400B_9054	Features Mode Selection (FTM3_MODE)	32	R/W	0000_0004h	40.3.10/ 1068
400B_9058	Synchronization (FTM3_SYNC)	32	R/W	0000_0000h	40.3.11/ 1069
400B_905C	Initial State for Channels Output (FTM3_OUTINIT)	32	R/W	0000_0000h	40.3.12/ 1072
400B_9060	Output Mask (FTM3_OUTMASK)	32	R/W	0000_0000h	40.3.13/ 1073
400B_9064	Function for Linked Channels (FTM3_COMBINE)	32	R/W	0000_0000h	40.3.14/ 1075
400B_9068	Deadtime Insertion Control (FTM3_DEADTIME)	32	R/W	0000_0000h	40.3.15/ 1080
400B_906C	FTM External Trigger (FTM3_EXTTRIG)	32	R/W	0000_0000h	40.3.16/ 1081
400B_9070	Channels Polarity (FTM3_POL)	32	R/W	0000_0000h	40.3.17/ 1083
400B_9074	Fault Mode Status (FTM3_FMS)	32	R/W	0000_0000h	40.3.18/ 1085
400B_9078	Input Capture Filter Control (FTM3_FILTER)	32	R/W	0000_0000h	40.3.19/ 1087
400B_907C	Fault Control (FTM3_FLTCTRL)	32	R/W	0000_0000h	40.3.20/ 1089
400B_9080	Quadrature Decoder Control and Status (FTM3_QDCTRL)	32	R/W	0000_0000h	40.3.21/ 1091
400B_9084	Configuration (FTM3_CONF)	32	R/W	0000_0000h	40.3.22/ 1093
400B_9088	FTM Fault Input Polarity (FTM3_FLTPOL)	32	R/W	0000_0000h	40.3.23/ 1094
400B_908C	Synchronization Configuration (FTM3_SYNCONF)	32	R/W	0000_0000h	40.3.24/ 1096
400B_9090	FTM Inverting Control (FTM3_INVCTRL)	32	R/W	0000_0000h	40.3.25/ 1098
400B_9094	FTM Software Output Control (FTM3_SWOCTRL)	32	R/W	0000_0000h	40.3.26/ 1099
400B_9098	FTM PWM Load (FTM3_PWMLOAD)	32	R/W	0000_0000h	40.3.27/ 1102

40.3.3 Status and Control (FTMx_SC)

SC contains the overflow status flag and control bits used to configure the interrupt enable, FTM configuration, clock source, and prescaler factor. These controls relate to all channels within this module.

Addresses: FTM0_SC is 4003_8000h base + 0h offset = 4003_8000h
FTM1_SC is 4003_9000h base + 0h offset = 4003_9000h
FTM2_SC is 400B_8000h base + 0h offset = 400B_8000h
FTM3_SC is 400B_9000h base + 0h offset = 400B_9000h

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								TOF	TOIE	CPWMS	CLKS		PS		
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SC field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 TOF	Timer Overflow Flag Set by hardware when the FTM counter passes the value in the MOD register. The TOF bit is cleared by reading the SC register while TOF is set and then writing a 0 to TOF bit. Writing a 1 to TOF has no effect. If another FTM overflow occurs between the read and write operations, the write operation has no effect; therefore, TOF remains set indicating an overflow has occurred. In this case a TOF interrupt request is not lost due to the clearing sequence for a previous TOF. 0 FTM counter has not overflowed. 1 FTM counter has overflowed.
6 TOIE	Timer Overflow Interrupt Enable Enables FTM overflow interrupts. 0 Disable TOF interrupts. Use software polling. 1 Enable TOF interrupts. An interrupt is generated when TOF equals one.
5 CPWMS	Center-aligned PWM Select Selects CPWM mode. This mode configures the FTM to operate in up-down counting mode.

Table continues on the next page...

FTMx_SC field descriptions (continued)

Field	Description
	This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 FTM counter operates in up counting mode. 1 FTM counter operates in up-down counting mode.
4–3 CLKS	Clock Source Selection Selects one of the three FTM counter clock sources. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00 No clock selected (This in effect disables the FTM counter.) 01 System clock 10 Fixed frequency clock 11 External clock
2–0 PS	Prescale Factor Selection Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next system clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when MODE[WPDIS] = 1. 000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128

40.3.4 Counter (FTMx_CNT)

The CNT register contains the FTM counter value.

Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value (CNTIN).

When BDM is active, the FTM counter is frozen (this is the value that you may read).

Addresses: FTM0_CNT is 4003_8000h base + 4h offset = 4003_8004h
FTM1_CNT is 4003_9000h base + 4h offset = 4003_9004h
FTM2_CNT is 400B_8000h base + 4h offset = 400B_8004h
FTM3_CNT is 400B_9000h base + 4h offset = 400B_9004h

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																COUNT															
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

FTMx_CNT field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 COUNT	Counter value

40.3.5 Modulo (FTMx_MOD)

The Modulo register contains the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock, and the next value of FTM counter depends on the selected counting method (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 Registers Updated from Write Buffers.

If FTMEN = 0, this write coherency mechanism may be manually reset by writing to the SC register (whether BDM is active or not).

It is recommended to initialize the FTM counter (write to CNT) before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Addresses: FTM0_MOD is 4003_8000h base + 8h offset = 4003_8008h
FTM1_MOD is 4003_9000h base + 8h offset = 4003_9008h
FTM2_MOD is 400B_8000h base + 8h offset = 400B_8008h
FTM3_MOD is 400B_9000h base + 8h offset = 400B_9008h

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																MOD															
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

FTMx_MOD field descriptions

Field	Description
31–16 Reserved	This field is reserved.
15–0 MOD	Modulo value

40.3.6 Channel (n) Status and Control (FTMx_CSC)

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

Table 40-67. Mode, Edge, and Level Selection

DECAPEN	COMBINE	CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	X	X	XX	0	None	Pin not used for FTM

Table continues on the next page...

Table 40-67. Mode, Edge, and Level Selection (continued)

DECAPEN	COMBINE	CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
0	0	0	0	1	Input capture	Capture on Rising Edge Only
				10		Capture on Falling Edge Only
				11		Capture on Rising or Falling Edge
			1	1	Output compare	Toggle Output on match
				10		Clear Output on match
				11		Set Output on match
			1X	10	Edge-aligned PWM	High-true pulses (clear Output on match)
				X1		Low-true pulses (set Output on match)
		1	XX	10	Center-aligned PWM	High-true pulses (clear Output on match-up)
				X1		Low-true pulses (set Output on match-up)
	1	0	XX	10	Combine PWM	High-true pulses (set on channel (n) match, and clear on channel (n+1) match)
				X1		Low-true pulses (clear on channel (n) match, and set on channel (n+1) match)
1	0	0	X0	See the following table (Table 40-8).	Dual Edge Capture Mode	One-shot capture mode
			X1			Continuous capture mode

Table 40-68. Dual Edge Capture Mode — Edge Polarity Selection

ELSnB	ELSnA	Channel Port Enable	Detected Edges
0	0	Disabled	No edge
0	1	Enabled	Rising edge
1	0	Enabled	Falling edge
1	1	Enabled	Rising and falling edges

Addresses: FTM0_C0SC is 4003_8000h base + Ch offset = 4003_800Ch

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								CHF	CHIE	MSB	MSA	ELSB	ELSA	0	DMA
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_CnSC field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 CHF	Channel Flag Set by hardware when an event occurs on the channel. CHF is cleared by reading the CSC register while CHnF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect. If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF. 0 No channel event has occurred. 1 A channel event has occurred.
6 CHIE	Channel Interrupt Enable Enables channel interrupts. 0 Disable channel interrupts. Use software polling. 1 Enable channel interrupts.
5 MSB	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See Table 40-7.

Table continues on the next page...

FTMx_*Cn*SC field descriptions (continued)

Field	Description
	This field is write protected. It can be written only when MODE[WPDIS] = 1.
4 MSA	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See Table 40-7. This field is write protected. It can be written only when MODE[WPDIS] = 1.
3 ELSB	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. See Table 40-7. This field is write protected. It can be written only when MODE[WPDIS] = 1.
2 ELSA	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. See Table 40-7. This field is write protected. It can be written only when MODE[WPDIS] = 1.
1 Reserved	This read-only field is reserved and always has the value zero.
0 DMA	DMA Enable Enables DMA transfers for the channel. 0 Disable DMA transfers. 1 Enable DMA transfers.

40.3.7 Channel (n) Value (FTMx_CV)

These registers contain the captured FTM counter value for the input modes or the match value for the output modes.

In input capture, capture test, and dual edge capture modes, any write to a CnV register is ignored.

In output 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 Registers Updated from Write Buffers.

If FTMEN = 0, this write coherency mechanism may be manually reset by writing to the CnSC register (whether BDM mode is active or not).

Addresses: FTM0_C0V is 4003_8000h base + 10h offset = 4003_8010h

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																VAL															
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

FTMx_*Cn*V field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 VAL	Channel Value Captured FTM counter value of the input modes or the match value for the output modes

40.3.8 Counter Initial Value (FTMx_CNTIN)

The Counter Initial Value register contains the initial value for the FTM counter.

Writing to the CNTIN register latches the value into a buffer. The CNTIN register is updated with the value of its write buffer according to Registers Updated from Write Buffers.

The first time that the FTM clock is selected (first write to change the CLKS bits to a non-zero value), the FTM counter starts with the value 0x0000. To avoid this behavior, before the first write to select the FTM clock, write the new value to the the CNTIN register and then initialize the FTM counter (write any value to the CNT register).

Addresses: FTM0_CNTIN is 4003_8000h base + 4Ch offset = 4003_804Ch
FTM1_CNTIN is 4003_9000h base + 4Ch offset = 4003_904Ch
FTM2_CNTIN is 400B_8000h base + 4Ch offset = 400B_804Ch
FTM3_CNTIN is 400B_9000h base + 4Ch offset = 400B_904Ch

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																INIT															
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

FTMx_CNTIN field descriptions

Field	Description
31–16 Reserved	This field is reserved.
15–0 INIT	Initial Value of the FTM Counter

40.3.9 Capture and Compare Status (FTMx_STATUS)

The STATUS register contains a copy of the status flag CHnF bit (in CnSC) for each FTM channel 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 reading STATUS followed by writing 0x00 to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. CHF is cleared by reading STATUS while CHnF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect.

If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

NOTE

The STATUS register should be used only combine mode.

Addresses: FTM0_STATUS is 4003_8000h base + 50h offset = 4003_8050h
FTM1_STATUS is 4003_9000h base + 50h offset = 4003_9050h
FTM2_STATUS is 400B_8000h base + 50h offset = 400B_8050h
FTM3_STATUS is 400B_9000h base + 50h offset = 400B_9050h

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								CH7F	CH6F	CH5F	CH4F	CH3F	CH2F	CH1F	CH0F
W									0	0	0	0	0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_STATUS field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 CH7F	Channel 7 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
6 CH6F	Channel 6 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.

Table continues on the next page...

FTMx_STATUS field descriptions (continued)

Field	Description
5 CH5F	Channel 5 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
4 CH4F	Channel 4 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
3 CH3F	Channel 3 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
2 CH2F	Channel 2 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
1 CH1F	Channel 1 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.
0 CH0F	Channel 0 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred.

40.3.10 Features Mode Selection (FTMx_MODE)

This register contains the control bits used to configure the fault interrupt and fault control, capture test mode, PWM synchronization, write protection, channel output initialization, and enable the enhanced features of the FTM. These controls relate to all channels within this module.

Addresses: FTM0_MODE is 4003_8000h base + 54h offset = 4003_8054h
FTM1_MODE is 4003_9000h base + 54h offset = 4003_9054h
FTM2_MODE is 400B_8000h base + 54h offset = 400B_8054h
FTM3_MODE is 400B_9000h base + 54h offset = 400B_9054h

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								FAULTIE	FAULTM		CAPTEST	PWMSYNC	WPDIS	INIT	FTMEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

FTMx_MODE field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 FAULTIE	Fault Interrupt Enable Enables the generation of an interrupt when a fault is detected by FTM and the FTM fault control is enabled. 0 Fault control interrupt is disabled. 1 Fault control interrupt is enabled.
6–5 FAULTM	Fault Control Mode Defines the FTM fault control mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00 Fault control is disabled for all channels. 01 Fault control is enabled for even channels only (channels 0, 2, 4, and 6), and the selected mode is the manual fault clearing. 10 Fault control is enabled for all channels, and the selected mode is the manual fault clearing. 11 Fault control is enabled for all channels, and the selected mode is the automatic fault clearing.

Table continues on the next page...

FTMx_MODE field descriptions (continued)

Field	Description
4 CAPTEST	Capture Test Mode Enable Enables the capture test mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Capture test mode is disabled. 1 Capture test mode is enabled.
3 PWMSYNC	PWM Synchronization Mode Selects which triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization (PWM Synchronization). The PWMSYNC bit configures the synchronization when SYNCMODE is zero. 0 No restrictions. Software and hardware triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. 1 Software trigger can only be used by MOD and CnV synchronization, and hardware triggers can only be used by OUTMASK and FTM counter synchronization.
2 WPDIS	Write Protection Disable When write protection is enabled (WPDIS = 0), write protected bits can not be written. When write protection is disabled (WPDIS = 1), write protected bits can be written. The WPDIS bit is the negation of the WPEN bit. WPDIS is cleared when 1 is written to WPEN. WPDIS is set when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPDIS has no effect. 0 Write protection is enabled. 1 Write protection is disabled.
1 INIT	Initialize the Channels Output When a 1 is written to INIT bit the channels output is initialized according to the state of their corresponding bit in the OUTINIT register. Writing a 0 to INIT bit has no effect. The INIT bit is always read as 0.
0 FTMEN	FTM Enable This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Only the TPM-compatible registers (first set of registers) can be used without any restriction. Do not use the FTM-specific registers. 1 All registers including the FTM-specific registers (second set of registers) are available for use with no restrictions.

40.3.11 Synchronization (FTMx_SYNC)

This register configures the PWM synchronization.

A synchronization event can perform the synchronized update of MOD, CV, and OUTMASK registers with the value of their write buffer and the FTM counter initialization.

NOTE

The software trigger (SWSYNC bit) and hardware triggers (TRIG0, TRIG1, and TRIG2 bits) have a potential conflict if used together when SYNCMODE = 0. It is recommended using only hardware or software triggers but not both at the same time, otherwise unpredictable behavior is likely to happen.

The selection of the loading point (CNTMAX and CNTMIN bits) is intended to provide the update of MOD, CNTIN, and CnV registers across all enabled channels simultaneously. The use of the loading point selection together with SYNCMODE = 0 and hardware trigger selection (TRIG0, TRIG1, or TRIG2 bits) is likely to result in unpredictable behavior.

The synchronization event selection also depends on the PWMSYNC (MODE register) and SYNCMODE (SYNCONF register) bits. See PWM Synchronization.

Addresses: FTM0_SYNC is 4003_8000h base + 58h offset = 4003_8058h
FTM1_SYNC is 4003_9000h base + 58h offset = 4003_9058h
FTM2_SYNC is 400B_8000h base + 58h offset = 400B_8058h
FTM3_SYNC is 400B_9000h base + 58h offset = 400B_9058h

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								SWSYNC	TRIG2	TRIG1	TRIG0	SYNCHOM	REINIT	CNTMAX	CNTMIN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SYNC field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 SWSYNC	PWM Synchronization Software Trigger Selects the software trigger as the PWM synchronization trigger. The software trigger happens when a 1 is written to SWSYNC bit.

Table continues on the next page...

FTMx_SYNC field descriptions (continued)

Field	Description
	0 Software trigger is not selected. 1 Software trigger is selected.
6 TRIG2	PWM Synchronization Hardware Trigger 2 Enables hardware trigger 2 to the PWM synchronization. Hardware trigger 2 happens when a rising edge is detected at the trigger 2 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
5 TRIG1	PWM Synchronization Hardware Trigger 1 Enables hardware trigger 1 to the PWM synchronization. Hardware trigger 1 happens when a rising edge is detected at the trigger 1 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
4 TRIG0	PWM Synchronization Hardware Trigger 0 Enables hardware trigger 0 to the PWM synchronization. Hardware trigger 0 happens when a rising edge is detected at the trigger 0 input signal. 0 Trigger is disabled. 1 Trigger is enabled.
3 SYNCHOM	Output Mask Synchronization Selects when the OUTMASK register is updated with the value of its buffer. 0 OUTMASK register is updated with the value of its buffer in all rising edges of the system clock. 1 OUTMASK register is updated with the value of its buffer only by the PWM synchronization.
2 REINIT	FTM Counter Reinitialization by Synchronization (FTM Counter Synchronization) Determines if the FTM counter is reinitialized when the selected trigger for the synchronization is detected. The REINIT bit configures the synchronization when SYNCMODE is zero. 0 FTM counter continues to count normally. 1 FTM counter is updated with its initial value when the selected trigger is detected.
1 CNTMAX	Maximum loading point enable Selects the maximum loading point to PWM synchronization (Boundary Cycle and Loading Points). If CNTMAX is one, the selected loading point is when the FTM counter reaches its maximum value (MOD register). 0 The maximum loading point is disabled. 1 The maximum loading point is enabled.
0 CNTMIN	Minimum loading point enable Selects the minimum loading point to PWM synchronization (Boundary Cycle and Loading Points). If CNTMIN is one, the selected loading point is when the FTM counter reaches its minimum value (CNTIN register).

Table continues on the next page...

FTMx_SYNC field descriptions (continued)

Field	Description
	0 The minimum loading point is disabled. 1 The minimum loading point is enabled.

40.3.12 Initial State for Channels Output (FTMx_OUTINIT)

Addresses: FTM0_OUTINIT is 4003_8000h base + 5Ch offset = 4003_805Ch
FTM1_OUTINIT is 4003_9000h base + 5Ch offset = 4003_905Ch
FTM2_OUTINIT is 400B_8000h base + 5Ch offset = 400B_805Ch
FTM3_OUTINIT is 400B_9000h base + 5Ch offset = 400B_905Ch

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																								CH7OI	CH6OI	CH5OI	CH4OI	CH3OI	CH2OI	CH1OI	CH0OI
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

FTMx_OUTINIT field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 CH7OI	Channel 7 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
6 CH6OI	Channel 6 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
5 CH5OI	Channel 5 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
4 CH4OI	Channel 4 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.

Table continues on the next page...

FTMx_OUTINIT field descriptions (continued)

Field	Description
3 CH3OI	Channel 3 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
2 CH2OI	Channel 2 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
1 CH1OI	Channel 1 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.
0 CH0OI	Channel 0 Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0 The initialization value is 0. 1 The initialization value is 1.

40.3.13 Output Mask (FTMx_OUTMASK)

This register provides a mask for each FTM channel. The mask of a channel determines if its output responds (that is, it is masked or not) when a match occurs. This feature is used for BLDC control where the PWM signal is presented to an electric motor at specific times to provide electronic commutation.

Any write to the OUTMASK register, stores the value in its write buffer. The register is updated with the value of its write buffer according to PWM Synchronization.

Addresses: FTM0_OUTMASK is 4003_8000h base + 60h offset = 4003_8060h
FTM1_OUTMASK is 4003_9000h base + 60h offset = 4003_9060h
FTM2_OUTMASK is 400B_8000h base + 60h offset = 400B_8060h
FTM3_OUTMASK is 400B_9000h base + 60h offset = 400B_9060h

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																								CH7OM	CH6OM	CH5OM	CH4OM	CH3OM	CH2OM	CH1OM	CH0OM
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

FTMx_OUTMASK field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 CH7OM	Channel 7 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
6 CH6OM	Channel 6 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
5 CH5OM	Channel 5 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
4 CH4OM	Channel 4 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
3 CH3OM	Channel 3 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
2 CH2OM	Channel 2 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.
1 CH1OM	Channel 1 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.

Table continues on the next page...

FTMx_OUTMASK field descriptions (continued)

Field	Description
0 CH0OM	Channel 0 Output Mask Defines if the channel output is masked (forced to its inactive state) or unmasked (it continues to operate normally). 0 Channel output is not masked. It continues to operate normally. 1 Channel output is masked. It is forced to its inactive state.

40.3.14 Function for Linked Channels (FTMx_COMBINE)

This register contains the control bits used to configure the fault control, synchronization, deadtime insertion, dual edge capture mode, complementary, and combine mode for each pair of channels (n) and (n+1), where n equals 0, 2, 4, and 6.

Addresses: FTM0_COMBINE is 4003_8000h base + 64h offset = 4003_8064h
FTM1_COMBINE is 4003_9000h base + 64h offset = 4003_9064h
FTM2_COMBINE is 400B_8000h base + 64h offset = 400B_8064h
FTM3_COMBINE is 400B_9000h base + 64h offset = 400B_9064h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	FAULTEN3	SYNCEN3	DTEN3	DECAP3	DECAPEN3	COMP3	COMBINE3	0	FAULTEN2	SYNCEN2	DTEN2	DECAP2	DECAPEN2	COMP2	COMBINE2
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	FAULTEN1	SYNCEN1	DTEN1	DECAP1	DECAPEN1	COMP1	COMBINE1	0	FAULTEN0	SYNCEN0	DTEN0	DECAP0	DECAPEN0	COMP0	COMBINE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_COMBINE field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value zero.
30 FAULTEN3	Fault Control Enable for n = 6 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
29 SYNCEN3	Synchronization Enable for n = 6 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
28 DTEN3	Deadtime Enable for n = 6 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
27 DECAP3	Dual Edge Capture Mode Captures for n = 6 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when FTMEN = 1 and DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
26 DECAPEN3	Dual Edge Capture Mode Enable for n = 6 Enables the dual edge capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in dual edge capture mode according to Table 40-7. This field applies only when FTMEN = 1. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The dual edge capture mode in this pair of channels is disabled. 1 The dual edge capture mode in this pair of channels is enabled.
25 COMP3	Complement of Channel (n) for n = 6 Enables complementary mode for the combined channels. In complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
24 COMBINE3	Combine Channels for n = 6 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
23 Reserved	This read-only field is reserved and always has the value zero.
22 FAULTEN2	Fault Control Enable for n = 4 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
21 SYNCEN2	Synchronization Enable for n = 4 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
20 DTEN2	Deadtime Enable for n = 4 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
19 DECAP2	Dual Edge Capture Mode Captures for n = 4 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when FTMEN = 1 and DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
18 DECAPEN2	Dual Edge Capture Mode Enable for n = 4 Enables the dual edge capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in dual edge capture mode according to Table 40-7. This field applies only when FTMEN = 1. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The dual edge capture mode in this pair of channels is disabled. 1 The dual edge capture mode in this pair of channels is enabled.
17 COMP2	Complement of Channel (n) for n = 4 Enables complementary mode for the combined channels. In complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
	0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
16 COMBINE2	Combine Channels for n = 4 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.
15 Reserved	This read-only field is reserved and always has the value zero.
14 FAULTEN1	Fault Control Enable for n = 2 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
13 SYNCEN1	Synchronization Enable for n = 2 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
12 DTEN1	Deadtime Enable for n = 2 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
11 DECAP1	Dual Edge Capture Mode Captures for n = 2 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when FTMEN = 1 and DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.
10 DECAPEN1	Dual Edge Capture Mode Enable for n = 2 Enables the dual edge capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in dual edge capture mode according to Table 40-7. This field applies only when FTMEN = 1. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
	0 The dual edge capture mode in this pair of channels is disabled. 1 The dual edge capture mode in this pair of channels is enabled.
9 COMP1	Complement of Channel (n) for n = 2 Enables complementary mode for the combined channels. In complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
8 COMBINE1	Combine Channels for n = 2 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.
7 Reserved	This read-only field is reserved and always has the value zero.
6 FAULTEN0	Fault Control Enable for n = 0 Enables the fault control in channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault control in this pair of channels is disabled. 1 The fault control in this pair of channels is enabled.
5 SYNCEN0	Synchronization Enable for n = 0 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0 The PWM synchronization in this pair of channels is disabled. 1 The PWM synchronization in this pair of channels is enabled.
4 DTEN0	Deadtime Enable for n = 0 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The deadtime insertion in this pair of channels is disabled. 1 The deadtime insertion in this pair of channels is enabled.
3 DECAP0	Dual Edge Capture Mode Captures for n = 0 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when FTMEN = 1 and DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0 The dual edge captures are inactive. 1 The dual edge captures are active.

Table continues on the next page...

FTMx_COMBINE field descriptions (continued)

Field	Description
2 DECAPEN0	Dual Edge Capture Mode Enable for n = 0 Enables the dual edge capture mode in the channels (n) and (n+1). This bit reconfigures the function of MSnA, ELSnB:ELSnA and ELS(n+1)B:ELS(n+1)A bits in dual edge capture mode according to Table 40-7. This field applies only when FTMEN = 1. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The dual edge capture mode in this pair of channels is disabled. 1 The dual edge capture mode in this pair of channels is enabled.
1 COMP0	Complement of Channel (n) for n = 0 Enables complementary mode for the combined channels. In complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel (n+1) output is the same as the channel (n) output. 1 The channel (n+1) output is the complement of the channel (n) output.
0 COMBINE0	Combine Channels for n = 0 Enables the combine feature for channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Channels (n) and (n+1) are independent. 1 Channels (n) and (n+1) are combined.

40.3.15 Deadtime Insertion Control (FTMx_DEADTIME)

This register selects the deadtime prescaler factor and deadtime value. All FTM channels use this clock prescaler and this deadtime value for the deadtime insertion.

Addresses: FTM0_DEADTIME is 4003_8000h base + 68h offset = 4003_8068h
FTM1_DEADTIME is 4003_9000h base + 68h offset = 4003_9068h
FTM2_DEADTIME is 400B_8000h base + 68h offset = 400B_8068h
FTM3_DEADTIME is 400B_9000h base + 68h offset = 400B_9068h

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																								DTPS		DTVAL					
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

FTMx_DEADTIME field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–6 DTPS	Deadtime Prescaler Value Selects the division factor of the system clock. This prescaled clock is used by the deadtime counter. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0x Divide the system clock by 1. 10 Divide the system clock by 4. 11 Divide the system clock by 16.
5–0 DTVAL	Deadtime Value Selects the deadtime insertion value for the deadtime counter. The deadtime counter is clocked by a scaled version of the system clock. See the description of DTPS. Deadtime insert value = (DTPS × DTVAL). DTVAL selects the number of deadtime counts inserted as follows: When DTVAL is 0, no counts are inserted. When DTVAL is 1, 1 count is inserted. When DTVAL is 2, 2 counts are inserted. This pattern continues up to a possible 63 counts. This field is write protected. It can be written only when MODE[WPDIS] = 1.

40.3.16 FTM External Trigger (FTMx_EXTTRIG)

This register indicates when a channel trigger was generated, enables the generation of a trigger when the FTM counter is equal to its initial value, and selects which channels are used in the generation of the channel triggers. Several channels can be selected to generate multiple triggers in one PWM period.

Channels 6 and 7 are not used to generate channel triggers.

Addresses: FTM0_EXTTRIG is 4003_8000h base + 6Ch offset = 4003_806Ch
FTM1_EXTTRIG is 4003_9000h base + 6Ch offset = 4003_906Ch
FTM2_EXTTRIG is 400B_8000h base + 6Ch offset = 400B_806Ch
FTM3_EXTTRIG is 400B_9000h base + 6Ch offset = 400B_906Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved[bit 8]															
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[7:0]								TRIGF	INITTRIGEN	CH1TRIG	CH0TRIG	CH5TRIG	CH4TRIG	CH3TRIG	CH2TRIG
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_EXTTRIG field descriptions

Field	Description
31–8 Reserved	This field is reserved.
7 TRIGF	Channel Trigger Flag Set by hardware when a channel trigger is generated. Clear TRIGF by reading EXTTRIG while TRIGF is set and then writing a 0 to TRIGF. Writing a 1 to TRIGF has no effect. If another channel trigger is generated before the clearing sequence is completed, the sequence is reset so TRIGF remains set after the clear sequence is completed for the earlier TRIGF. 0 No channel trigger was generated. 1 A channel trigger was generated.
6 INITTRIGEN	Initialization Trigger Enable Enables the generation of the trigger when the FTM counter is equal to the CNTIN register. 0 The generation of initialization trigger is disabled. 1 The generation of initialization trigger is enabled.
5 CH1TRIG	Channel 1 Trigger Enable Enable the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
4 CH0TRIG	Channel 0 Trigger Enable Enable the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.

Table continues on the next page...

FTMx_EXTTRIG field descriptions (continued)

Field	Description
3 CH5TRIG	Channel 5 Trigger Enable Enable the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
2 CH4TRIG	Channel 4 Trigger Enable Enable the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
1 CH3TRIG	Channel 3 Trigger Enable Enable the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.
0 CH2TRIG	Channel 2 Trigger Enable Enable the generation of the channel trigger when the FTM counter is equal to the CnV register. 0 The generation of the channel trigger is disabled. 1 The generation of the channel trigger is enabled.

40.3.17 Channels Polarity (FTMx_POL)

This register defines the output polarity of the FTM channels.

NOTE

> The safe value that is driven in a channel output when the fault control is enabled and a fault condition is detected is the inactive state of the channel. That is, the safe value of a channel is the value of its POL bit.

Addresses: FTM0_POL is 4003_8000h base + 70h offset = 4003_8070h
FTM1_POL is 4003_9000h base + 70h offset = 4003_9070h
FTM2_POL is 400B_8000h base + 70h offset = 400B_8070h
FTM3_POL is 400B_9000h base + 70h offset = 400B_9070h

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																								POL7	POL6	POL5	POL4	POL3	POL2	POL1	POL0
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

FTMx_POL field descriptions

Field	Description
31–8 Reserved	This field is reserved.
7 POL7	Channel 7 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
6 POL6	Channel 6 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
5 POL5	Channel 5 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
4 POL4	Channel 4 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
3 POL3	Channel 3 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
2 POL2	Channel 2 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.
1 POL1	Channel 1 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_POL field descriptions (continued)

Field	Description
	0 The channel polarity is active high. 1 The channel polarity is active low.
0 POL0	Channel 0 Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The channel polarity is active high. 1 The channel polarity is active low.

40.3.18 Fault Mode Status (FTMx_FMS)

This register contains the fault detection flags, write protection enable bit, and the logic OR of the enabled fault inputs.

Addresses: FTM0_FMS is 4003_8000h base + 74h offset = 4003_8074h
FTM1_FMS is 4003_9000h base + 74h offset = 4003_9074h
FTM2_FMS is 400B_8000h base + 74h offset = 400B_8074h
FTM3_FMS is 400B_9000h base + 74h offset = 400B_9074h

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								FAULTF	WPEN	FAULTIN	0	FAULTF3	FAULTF2	FAULTF1	FAULTF0
W									0				0	0	0	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_FMS field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

FTMx_FMS field descriptions (continued)

Field	Description
7 FAULTF	Fault Detection Flag Represents the logic OR of the individual FAULTFj bits (where j = 3, 2, 1, 0). Clear FAULTF by reading the FMS register while FAULTF is set and then writing a 0 to FAULTF while there is no existing fault condition at the enabled fault inputs. Writing a 1 to FAULTF has no effect. If another fault condition is detected in an enabled fault input before the clearing sequence is completed, the sequence is reset so FAULTF remains set after the clearing sequence is completed for the earlier fault condition. FAULTF is also cleared when FAULTFj bits are cleared individually. 0 No fault condition was detected. 1 A fault condition was detected.
6 WPEN	Write Protection Enable The WPEN bit is the negation of the WPDIS bit. WPEN is set when 1 is written to it. WPEN is cleared when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPEN has no effect. 0 Write protection is disabled. Write protected bits can be written. 1 Write protection is enabled. Write protected bits cannot be written.
5 FAULTIN	Fault Inputs Represents the logic OR of the enabled fault inputs after their filter (if their filter is enabled) when fault control is enabled. 0 The logic OR of the enabled fault inputs is 0. 1 The logic OR of the enabled fault inputs is 1.
4 Reserved	This read-only field is reserved and always has the value zero.
3 FAULTF3	Fault Detection Flag 3 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF3 by reading the FMS register while FAULTF3 is set and then writing a 0 to FAULTF3 while there is no existing fault condition at the the corresponding fault input. Writing a 1 to FAULTF3 has no effect. FAULTF3 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF3 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
2 FAULTF2	Fault Detection Flag 2 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF2 by reading the FMS register while FAULTF2 is set and then writing a 0 to FAULTF2 while there is no existing fault condition at the the corresponding fault input. Writing a 1 to FAULTF2 has no effect. FAULTF2 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF2 remains set after the clearing sequence is completed for the earlier fault condition.

Table continues on the next page...

FTMx_FMS field descriptions (continued)

Field	Description
	0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
1 FAULTF1	Fault Detection Flag 1 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF1 by reading the FMS register while FAULTF1 is set and then writing a 0 to FAULTF1 while there is no existing fault condition at the the corresponding fault input. Writing a 1 to FAULTF1 has no effect. FAULTF1 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF1 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.
0 FAULTF0	Fault Detection Flag 0 Set by hardware when fault control is enabled, the corresponding fault input is enabled and a fault condition is detected at the fault input. Clear FAULTF0 by reading the FMS register while FAULTF0 is set and then writing a 0 to FAULTF0 while there is no existing fault condition at the the corresponding fault input. Writing a 1 to FAULTF0 has no effect. FAULTF0 bit is also cleared when FAULTF bit is cleared. If another fault condition is detected at the corresponding fault input before the clearing sequence is completed, the sequence is reset so FAULTF0 remains set after the clearing sequence is completed for the earlier fault condition. 0 No fault condition was detected at the fault input. 1 A fault condition was detected at the fault input.

40.3.19 Input Capture Filter Control (FTMx_FILTER)

This register selects the filter value for the inputs of channels.

Channels 4, 5, 6 and 7 do not have an input filter.

NOTE

Writing to the FILTER register has immediate effect and must be done only when the channels 0, 1, 2, and 3 are not in input modes. Failure to do this could result in a missing valid signal.

Addresses: FTM0_FILTER is 4003_8000h base + 78h offset = 4003_8078h
FTM1_FILTER is 4003_9000h base + 78h offset = 4003_9078h
FTM2_FILTER is 400B_8000h base + 78h offset = 400B_8078h
FTM3_FILTER is 400B_9000h base + 78h offset = 400B_9078h

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																CH3FVAL				CH2FVAL				CH1FVAL				CH0FVAL			
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

FTMx_FILTER field descriptions

Field	Description
31–16 Reserved	This field is reserved.
15–12 CH3FVAL	Channel 3 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
11–8 CH2FVAL	Channel 2 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
7–4 CH1FVAL	Channel 1 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
3–0 CH0FVAL	Channel 0 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.

40.3.20 Fault Control (FTMx_FLTCTRL)

This register selects the filter value for the fault inputs, enables the fault inputs and the fault inputs filter.

Addresses: FTM0_FLTCTRL is 4003_8000h base + 7Ch offset = 4003_807Ch
FTM1_FLTCTRL is 4003_9000h base + 7Ch offset = 4003_907Ch
FTM2_FLTCTRL is 400B_8000h base + 7Ch offset = 400B_807Ch
FTM3_FLTCTRL is 400B_9000h base + 7Ch offset = 400B_907Ch

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				FFVAL				FFLTR3EN	FFLTR2EN	FFLTR1EN	FFLTR0EN	FAULT3EN	FAULT2EN	FAULT1EN	FAULT0EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_FLTCTRL field descriptions

Field	Description
31–12 Reserved	This read-only field is reserved and always has the value zero.
11–8 FFVAL	Fault Input Filter Selects the filter value for the fault inputs. The fault filter is disabled when the value is zero. **NOTE:** Writing to this field has immediate effect and must be done only when the fault control or all fault inputs are disabled. Failure to do this could result in a missing fault detection.
7 FFLTR3EN	Fault Input 3 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
6 FFLTR2EN	Fault Input 2 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_FLTCTRL field descriptions (continued)

Field	Description
	0 Fault input filter is disabled. 1 Fault input filter is enabled.
5 FFLTR1EN	Fault Input 1 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
4 FFLTR0EN	Fault Input 0 Filter Enable Enables the filter for the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input filter is disabled. 1 Fault input filter is enabled.
3 FAULT3EN	Fault Input 3 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.
2 FAULT2EN	Fault Input 2 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.
1 FAULT1EN	Fault Input 1 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.
0 FAULT0EN	Fault Input 0 Enable Enables the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Fault input is disabled. 1 Fault input is enabled.

40.3.21 Quadrature Decoder Control and Status (FTMx_QDCTRL)

This register has the control and status bits for the quadrature decoder mode.

Addresses: FTM0_QDCTRL is 4003_8000h base + 80h offset = 4003_8080h
FTM1_QDCTRL is 4003_9000h base + 80h offset = 4003_9080h
FTM2_QDCTRL is 400B_8000h base + 80h offset = 400B_8080h
FTM3_QDCTRL is 400B_9000h base + 80h offset = 400B_9080h

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								PHAFLTREN	PHBFLTREN	PHAPOL	PHBPOL	QUADMODE	QUADIR	TOFDIR	QUADEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_QDCTRL field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 PHAFLTREN	Phase A Input Filter Enable Enables the filter for the quadrature decoder phase A input. The filter value for the phase A input is defined by the CH0FVAL field of FILTER. The phase A filter is also disabled when CH0FVAL is zero. 0 Phase A input filter is disabled. 1 Phase A input filter is enabled.
6 PHBFLTREN	Phase B Input Filter Enable Enables the filter for the quadrature decoder phase B input. The filter value for the phase B input is defined by the CH1FVAL field of FILTER. The phase B filter is also disabled when CH1FVAL is zero. 0 Phase B input filter is disabled. 1 Phase B input filter is enabled.

Table continues on the next page...

FTMx_QDCTRL field descriptions (continued)

Field	Description
5 PHAPOL	Phase A Input Polarity Selects the polarity for the quadrature decoder phase A input. 0 Normal polarity. Phase A input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase A input signal is inverted before identifying the rising and falling edges of this signal.
4 PHBPOL	Phase B Input Polarity Selects the polarity for the quadrature decoder phase B input. 0 Normal polarity. Phase B input signal is not inverted before identifying the rising and falling edges of this signal. 1 Inverted polarity. Phase B input signal is inverted before identifying the rising and falling edges of this signal.
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the quadrature decoder mode. 0 Phase A and phase B encoding mode. 1 Count and direction encoding mode.
2 QUADIR	FTM Counter Direction in Quadrature Decoder Mode Indicates the counting direction. 0 Counting direction is decreasing (FTM counter decrement). 1 Counting direction is increasing (FTM counter increment).
1 TOFDIR	Timer Overflow Direction in Quadrature Decoder Mode Indicates if the TOF bit was set on the top or the bottom of counting. 0 TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (CNTIN register) to its maximum value (MOD register). 1 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 (CNTIN register).
0 QUADEN	Quadrature Decoder Mode Enable Enables the quadrature decoder mode. In this mode, the phase A and B input signals control the FTM counter direction. The quadrature decoder mode has precedence over the other modes. (See Table 40-7.) This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 Quadrature decoder mode is disabled. 1 Quadrature decoder mode is enabled.

40.3.22 Configuration (FTMx_CONF)

This register selects the number of times that the FTM counter overflow should occur before the TOF bit to be set, the FTM behavior in BDM modes, the use of an external global time base, and the global time base signal generation.

Addresses: FTM0_CONF is 4003_8000h base + 84h offset = 4003_8084h
FTM1_CONF is 4003_9000h base + 84h offset = 4003_9084h
FTM2_CONF is 400B_8000h base + 84h offset = 400B_8084h
FTM3_CONF is 400B_9000h base + 84h offset = 400B_9084h

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					GTBEOUT	GTBEEN	0	BDMMODE		0	NUMTOF				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_CONF field descriptions

Field	Description
31–11 Reserved	This read-only field is reserved and always has the value zero.
10 GTBEOUT	Global time base output Enables the global time base signal generation to other FTMs. 0 A global time base signal generation is disabled. 1 A global time base signal generation is enabled.
9 GTBEEN	Global time base enable Configures the FTM to use an external global time base signal that is generated by another FTM. 0 Use of an external global time base is disabled. 1 Use of an external global time base is enabled.
8 Reserved	This read-only field is reserved and always has the value zero.
7–6 BDMMODE	BDM Mode Selects the FTM behavior in BDM mode. See BDM Mode.

Table continues on the next page...

FTMx_CONF field descriptions (continued)

Field	Description
5 Reserved	This read-only field is reserved and always has the value zero.
4–0 NUMTOF	TOF Frequency Selects the ratio between the number of counter overflows to the number of times the TOF bit is set. NUMTOF = 0: The TOF bit is set for each counter overflow. NUMTOF = 1: The TOF bit is set for the first counter overflow but not for the next overflow. NUMTOF = 2: The TOF bit is set for the first counter overflow but not for the next 2 overflows. NUMTOF = 3: The TOF bit is set for the first counter overflow but not for the next 3 overflows. This pattern continues up to a maximum of 31.

40.3.23 FTM Fault Input Polarity (FTMx_FLTPOL)

This register defines the fault inputs polarity.

Addresses: FTM0_FLTPOL is 4003_8000h base + 88h offset = 4003_8088h
FTM1_FLTPOL is 4003_9000h base + 88h offset = 4003_9088h
FTM2_FLTPOL is 400B_8000h base + 88h offset = 400B_8088h
FTM3_FLTPOL is 400B_9000h base + 88h offset = 400B_9088h

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												FLT3POL	FLT2POL	FLT1POL	FLT0POL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_FLTPOL field descriptions

Field	Description
31–4 Reserved	This read-only field is reserved and always has the value zero.
3 FLT3POL	Fault Input 3 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Table continues on the next page...

FTMx_FLTPOL field descriptions (continued)

Field	Description
	0 The fault input polarity is active high. A one at the fault input indicates a fault. 1 The fault input polarity is active low. A zero at the fault input indicates a fault.
2 FLT2POL	Fault Input 2 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A one at the fault input indicates a fault. 1 The fault input polarity is active low. A zero at the fault input indicates a fault.
1 FLT1POL	Fault Input 1 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A one at the fault input indicates a fault. 1 The fault input polarity is active low. A zero at the fault input indicates a fault.
0 FLT0POL	Fault Input 0 Polarity Defines the polarity of the fault input. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0 The fault input polarity is active high. A one at the fault input indicates a fault. 1 The fault input polarity is active low. A zero at the fault input indicates a fault.

40.3.24 Synchronization Configuration (FTMx_SYNCONF)

This register selects the PWM synchronization configuration, SWOCTRL, INVCTRL and CNTIN registers synchronization, if FTM clears the TRIGj bit (where j = 0, 1, 2) when the hardware trigger j is detected.

Addresses: FTM0_SYNCONF is 4003_8000h base + 8Ch offset = 4003_808Ch
FTM1_SYNCONF is 4003_9000h base + 8Ch offset = 4003_908Ch
FTM2_SYNCONF is 400B_8000h base + 8Ch offset = 400B_808Ch
FTM3_SYNCONF is 400B_9000h base + 8Ch offset = 400B_908Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											HWSOC	HWINVC	HWOM	HWWRBUF	HWRSTCNT
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			SWSOC	SWINVC	SWOM	SWWRBUF	SWRSTCNT	SYNCMODE	0	SWOC	INVC	0	CNTINC	0	HWTRIGMODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SYNCONF field descriptions

Field	Description
31–21 Reserved	This read-only field is reserved and always has the value zero.
20 HWSOC	Software output control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the SWOCTRL register synchronization. 1 A hardware trigger activates the SWOCTRL register synchronization.
19 HWINVC	Inverting control synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the INVCTRL register synchronization. 1 A hardware trigger activates the INVCTRL register synchronization.
18 HWOM	Output mask synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the OUTMASK register synchronization. 1 A hardware trigger activates the OUTMASK register synchronization.
17 HWWRBUF	MOD, CNTIN, and CV registers synchronization is activated by a hardware trigger.

Table continues on the next page...

FTMx_SYNCONF field descriptions (continued)

Field	Description
	0 A hardware trigger does not activate MOD, CNTIN, and CV registers synchronization. 1 A hardware trigger activates MOD, CNTIN, and CV registers synchronization.
16 HWRSTCNT	FTM counter synchronization is activated by a hardware trigger. 0 A hardware trigger does not activate the FTM counter synchronization. 1 A hardware trigger activates the FTM counter synchronization.
15–13 Reserved	This read-only field is reserved and always has the value zero.
12 SWSOC	Software output control synchronization is activated by the software trigger. 0 The software trigger does not activate the SWOCTRL register synchronization. 1 The software trigger activates the SWOCTRL register synchronization.
11 SWINVC	Inverting control synchronization is activated by the software trigger. 0 The software trigger does not activate the INVCTRL register synchronization. 1 The software trigger activates the INVCTRL register synchronization.
10 SWOM	Output mask synchronization is activated by the software trigger. 0 The software trigger does not activate the OUTMASK register synchronization. 1 The software trigger activates the OUTMASK register synchronization.
9 SWWRBUF	MOD, CNTIN, and CV registers synchronization is activated by the software trigger. 0 The software trigger does not activate MOD, CNTIN, and CV registers synchronization. 1 The software trigger activates MOD, CNTIN, and CV registers synchronization.
8 SWRSTCNT	FTM counter synchronization is activated by the software trigger. 0 The software trigger does not activate the FTM counter synchronization. 1 The software trigger activates the FTM counter synchronization.
7 SYNCMODE	Synchronization Mode Selects the PWM synchronization mode. 0 Legacy PWM synchronization is selected. 1 Enhanced PWM synchronization is selected.
6 Reserved	This read-only field is reserved and always has the value zero.
5 SWOC	SWOCTRL register synchronization 0 SWOCTRL register is updated with its buffer value at all rising edges of system clock. 1 SWOCTRL register is updated with its buffer value by the PWM synchronization.
4 INVC	INVCTRL register synchronization 0 INVCTRL register is updated with its buffer value at all rising edges of system clock. 1 INVCTRL register is updated with its buffer value by the PWM synchronization.
3 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

FTMx_SYNCONF field descriptions (continued)

Field	Description
2 CNTINC	CNTIN register synchronization 0 CNTIN register is updated with its buffer value at all rising edges of system clock. 1 CNTIN register is updated with its buffer value by the PWM synchronization.
1 Reserved	This read-only field is reserved and always has the value zero.
0 HWTRIGMODE	Hardware Trigger Mode 0 FTM clears the TRIGj bit when the hardware trigger j is detected. 1 FTM does not clear the TRIGj bit when the hardware trigger j is detected.

40.3.25 FTM Inverting Control (FTMx_INVCTRL)

This register controls controls when the channel (n) output becomes the channel (n+1) output, and channel (n+1) output becomes the channel (n) output. Each INVmEN bit enables the inverting operation for the corresponding pair channels m.

This register has a write buffer. The INVmEN bit is updated by the INVCTRL register synchronization.

Addresses: FTM0_INVCTRL is 4003_8000h base + 90h offset = 4003_8090h
FTM1_INVCTRL is 4003_9000h base + 90h offset = 4003_9090h
FTM2_INVCTRL is 400B_8000h base + 90h offset = 400B_8090h
FTM3_INVCTRL is 400B_9000h base + 90h offset = 400B_9090h

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																												INV3EN	INV2EN	INV1EN	INV0EN
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

FTMx_INVCTRL field descriptions

Field	Description
31–4 Reserved	This read-only field is reserved and always has the value zero.
3 INV3EN	Pair Channels 3 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
2 INV2EN	Pair Channels 2 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.

Table continues on the next page...

FTMx_INVCTRL field descriptions (continued)

Field	Description
1 INV1EN	Pair Channels 1 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.
0 INV0EN	Pair Channels 0 Inverting Enable 0 Inverting is disabled. 1 Inverting is enabled.

40.3.26 FTM Software Output Control (FTMx_SWOCTRL)

This register enables software control of channel (n) output and defines the value forced to the channel (n) output:

- The CHnOC bits enable the control of the corresponding channel (n) output by software.
- The CHnOCV bits select the value that is forced at the corresponding channel (n) output.

This register has a write buffer. The fields are updated by the SWOCTRL register synchronization.

Addresses: FTM0_SWOCTRL is 4003_8000h base + 94h offset = 4003_8094h
FTM1_SWOCTRL is 4003_9000h base + 94h offset = 4003_9094h
FTM2_SWOCTRL is 400B_8000h base + 94h offset = 400B_8094h
FTM3_SWOCTRL is 400B_9000h base + 94h offset = 400B_9094h

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 W	CH7OCV	CH6OCV	CH5OCV	CH4OCV	CH3OCV	CH2OCV	CH1OCV	CH0OCV	CH7OC	CH6OC	CH5OC	CH4OC	CH3OC	CH2OC	CH1OC	CH0OC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_SWOCTRL field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15 CH7OCV	Channel 7 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
14 CH6OCV	Channel 6 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
13 CH5OCV	Channel 5 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
12 CH4OCV	Channel 4 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
11 CH3OCV	Channel 3 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
10 CH2OCV	Channel 2 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
9 CH1OCV	Channel 1 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
8 CH0OCV	Channel 0 Software Output Control Value 0 The software output control forces 0 to the channel output. 1 The software output control forces 1 to the channel output.
7 CH7OC	Channel 7 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
6 CH6OC	Channel 6 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
5 CH5OC	Channel 5 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.

Table continues on the next page...

FTMx_SWOCTRL field descriptions (continued)

Field	Description
4 CH4OC	Channel 4 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
3 CH3OC	Channel 3 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
2 CH2OC	Channel 2 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
1 CH1OC	Channel 1 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.
0 CH0OC	Channel 0 Software Output Control Enable 0 The channel output is not affected by software output control. 1 The channel output is affected by software output control.

40.3.27 FTM PWM Load (FTMx_PWMLOAD)

Enables the loading of the MOD, CNTIN, C(n)V, and C(n+1)V registers with the values of their write buffers when the FTM counter changes from the MOD register value to its next value or when a channel (j) match occurs. A match occurs for the channel (j) when FTM counter = C(j)V.

Addresses: FTM0_PWMLOAD is 4003_8000h base + 98h offset = 4003_8098h
FTM1_PWMLOAD is 4003_9000h base + 98h offset = 4003_9098h
FTM2_PWMLOAD is 400B_8000h base + 98h offset = 400B_8098h
FTM3_PWMLOAD is 400B_9000h base + 98h offset = 400B_9098h

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						LDOK	0	CH7SEL	CH6SEL	CH5SEL	CH4SEL	CH3SEL	CH2SEL	CH1SEL	CH0SEL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FTMx_PWMLOAD field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value zero.
9 LDOK	Load Enable Enables the loading of the MOD, CNTIN, and CV registers with the values of their write buffers. 0 Loading updated values is disabled. 1 Loading updated values is enabled.
8 Reserved	This read-only field is reserved and always has the value zero.
7 CH7SEL	Channel 7 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
6 CH6SEL	Channel 6 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.

Table continues on the next page...

FTMx_PWMLOAD field descriptions (continued)

Field	Description
5 CH5SEL	Channel 5 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
4 CH4SEL	Channel 4 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
3 CH3SEL	Channel 3 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
2 CH2SEL	Channel 2 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
1 CH1SEL	Channel 1 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.
0 CH0SEL	Channel 0 Select 0 Do not include the channel in the matching process. 1 Include the channel in the matching process.

40.4 Functional Description

The following sections describe the FTM features.

The notation used in this document to represent the counters and the generation of the signals is shown in the following figure.

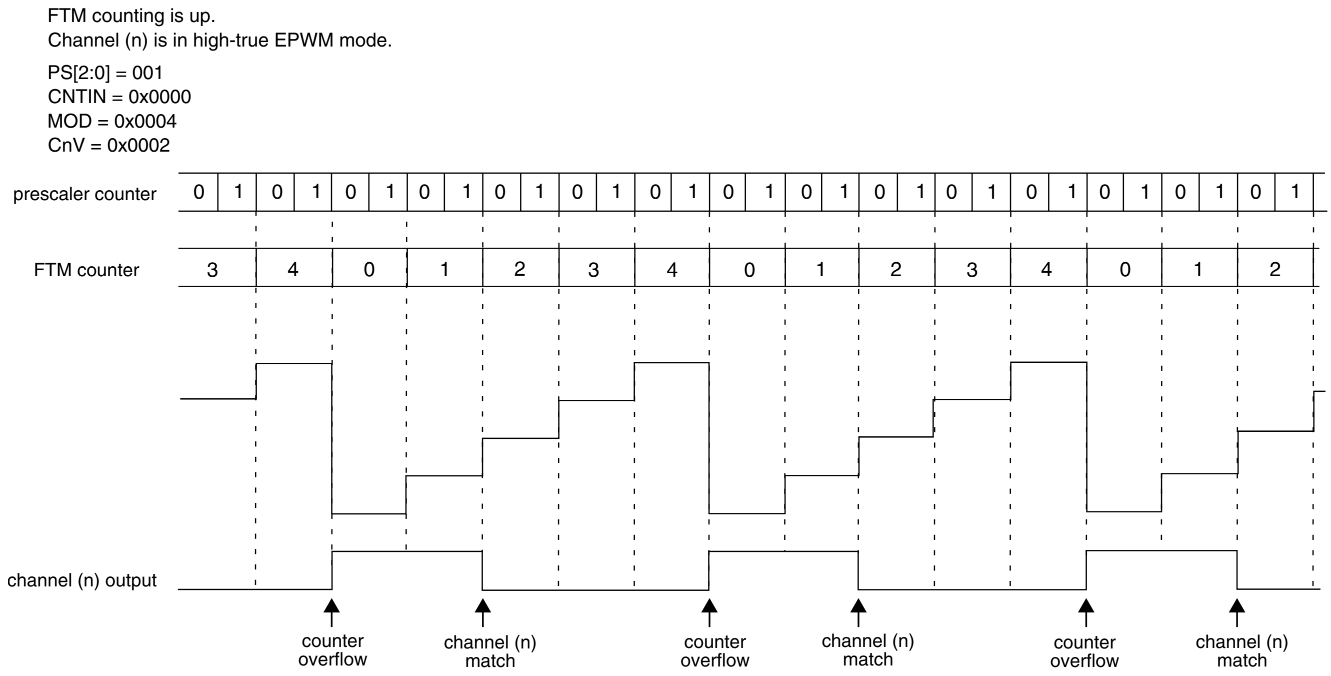


Figure 40-207. Notation Used

40.4.1 Clock Source

FTM module has only one clock domain that is the system clock.

40.4.1.1 Counter Clock Source

The CLKS[1:0] bits in the SC register select one of three possible clock sources for the FTM counter or disable the FTM counter. After any MCU reset, CLKS[1:0] = 0:0 so no clock source is selected.

The CLKS[1:0] bits may be read or written at any time. Disabling the FTM counter by writing 0:0 to the CLKS[1:0] bits does not affect the FTM counter value or other registers.

The fixed frequency clock is an alternative clock source for the FTM counter that allows the selection of a clock other than the system clock or an external clock. This clock input is defined by chip integration. Refer the chip specific documentation for further information. Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the system clock frequency.

The external clock passes through a synchronizer clocked by the system clock to assure that counter transitions are properly aligned to system clock transitions.Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

40.4.2 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 FTM counter.

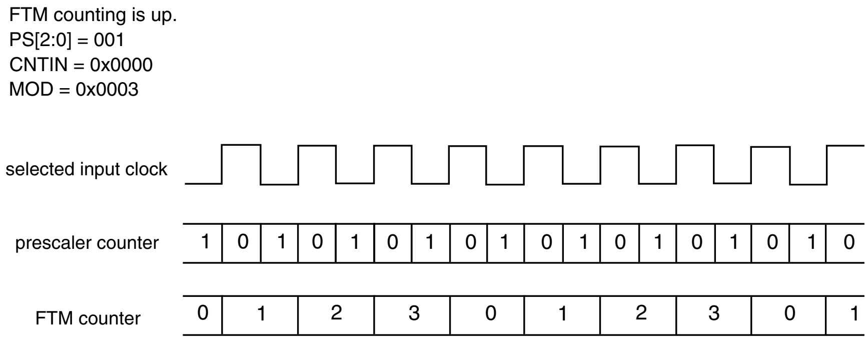


Figure 40-208. Example of the Prescaler Counter

40.4.3 Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler.

The FTM counter has these modes of operation:

- up counting (see Up Counting)
- up-down counting (see Up-Down Counting)
- quadrature mode (see Quadrature Decoder Mode)

40.4.3.1 Up Counting

Up counting is selected when (QUADEN = 0) and (CPWMS = 0).

CNTIN defines the starting value of the count and MOD defines the final value of the count (see the following figure). The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with the value of CNTIN.

The FTM period when using up counting is (MOD – CNTIN + 0x0001) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to CNTIN.

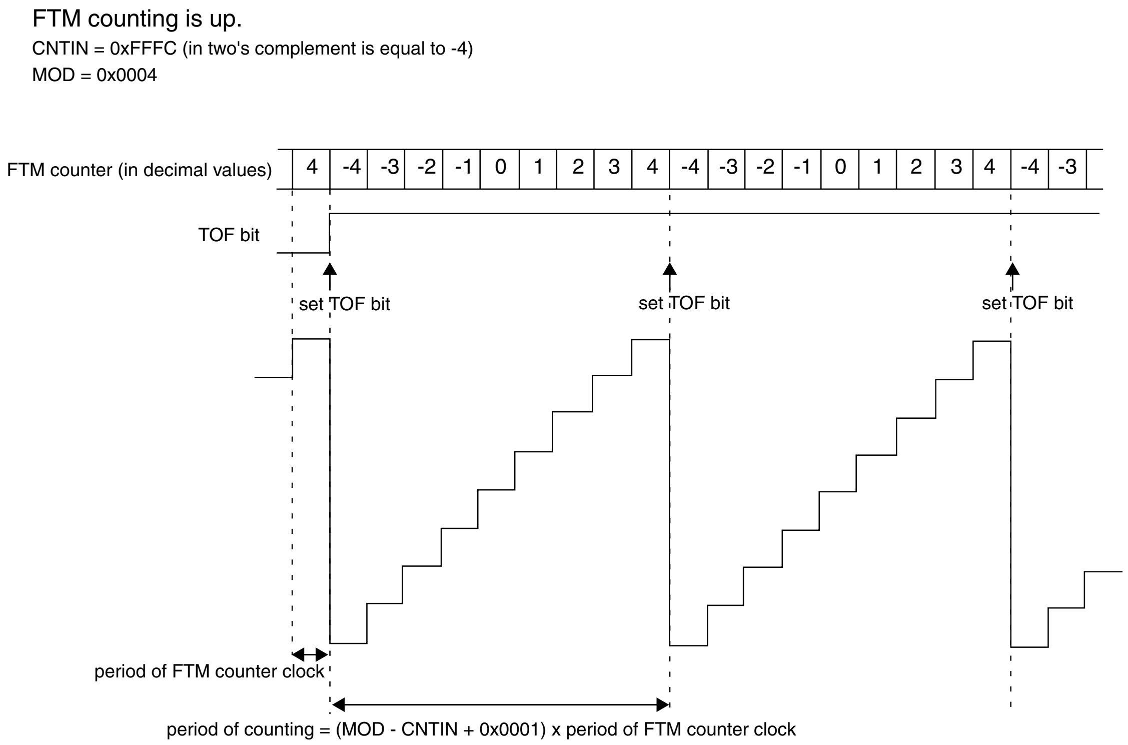

FTM counting is up.
CNTIN = 0xFFFC (in two's complement is equal to -4)
MOD = 0x0004
FTM counter (in decimal values)
4 -4 -3 -2 -1 0 1 2 3 4 -4 -3 -2 -1 0 1 2 3 4 -4 -3
TOF bit
set TOF bit
set TOF bit
set TOF bit
period of FTM counter clock
period of counting = (MOD - CNTIN + 0x0001) x period of FTM counter clock

Figure 40-209. Example of FTM Up and Signed Counting

If (CNTIN = 0x0000), the FTM counting is equivalent to TPM up counting (that is, up and unsigned counting) (see the following figure). If (CNTIN[15] = 1), then the initial value of the FTM counter is a negative number in two's complement, so the FTM counting is up and signed. Conversely if (CNTIN[15] = 0 and CNTIN ≠ 0x0000), then the initial value of the FTM counter is a positive number, so the FTM counting is up and unsigned.

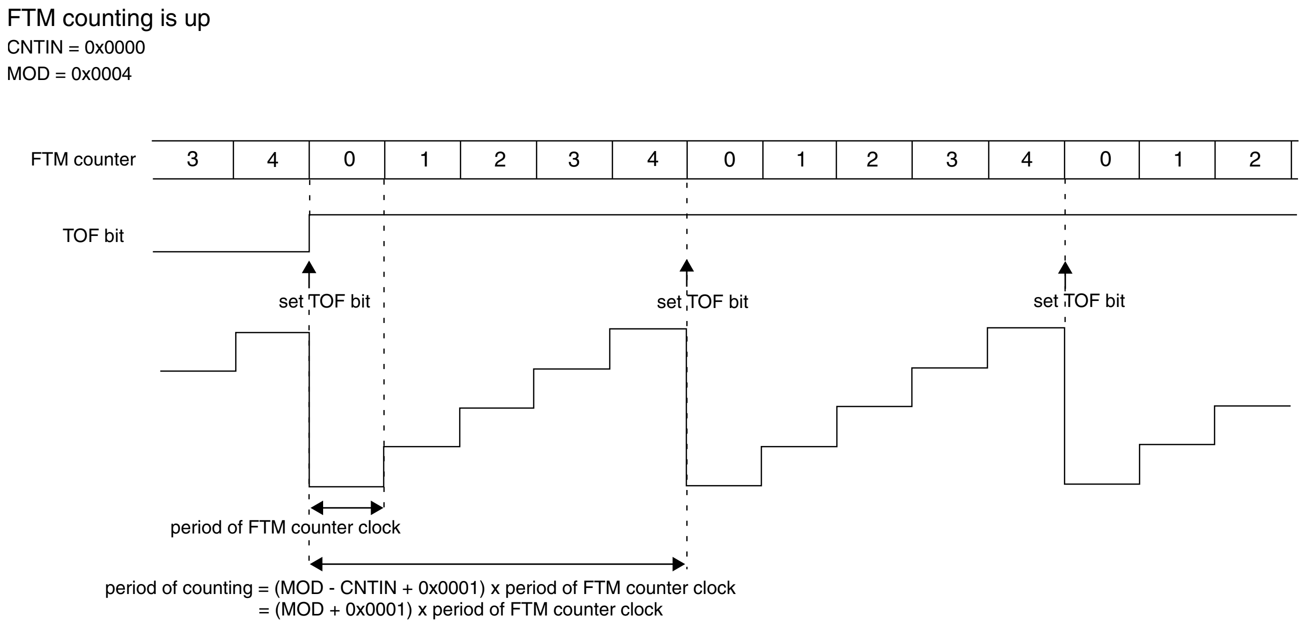

FTM counting is up
CNTIN = 0x0000
MOD = 0x0004
FTM counter
3
4
0
1
2
3
4
0
1
2
3
4
0
1
2
TOF bit
set TOF bit
set TOF bit
set TOF bit
period of FTM counter clock
period of counting = (MOD - CNTIN + 0x0001) x period of FTM counter clock
= (MOD + 0x0001) x period of FTM counter clock

Figure 40-210. Example of FTM Up Counting with CNTIN = 0x0000

Note

- FTM operation is only valid when the value of the CNTIN register is less than the value of the MOD register (either in the unsigned counting or signed counting). It is the responsibility of the software to ensure that the values in the CNTIN and MOD registers meet this requirement. Any values of CNTIN and MOD that do not satisfy this criteria can result in unpredictable behavior.
- MOD = CNTIN is a redundant condition. In this case, the FTM counter is always equal to MOD and the TOF bit is set in each rising edge of the FTM counter clock.
- When MOD = 0x0000, CNTIN = 0x0000 (for example after reset), and FTMEN = 1, the FTM counter remains stopped at 0x0000 until a non-zero value is written into the MOD or CNTIN registers.
- Setting CNTIN to be greater than the value of MOD is not recommended as this unusual setting may make the FTM operation difficult to comprehend. However, there is no restriction on this configuration, and an example is shown in the following figure.

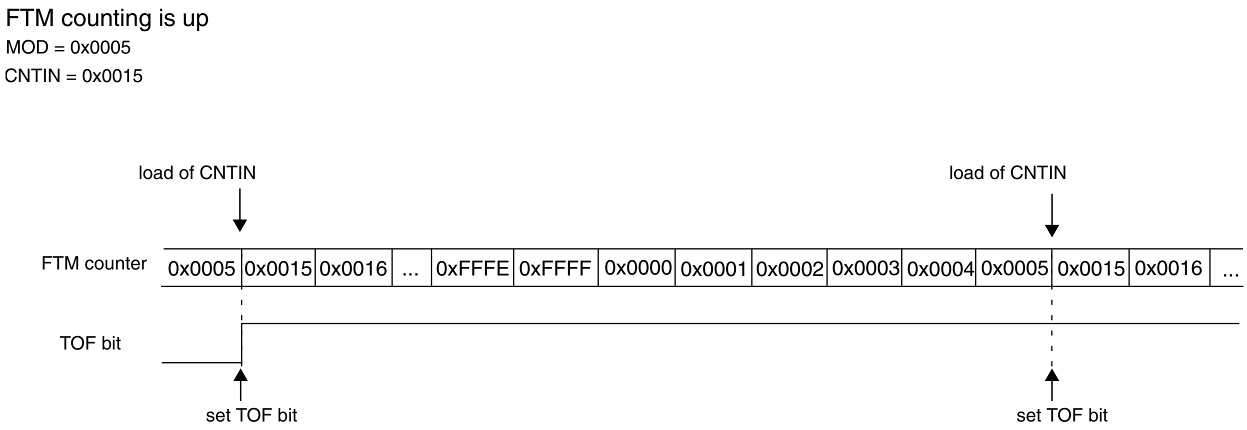


Figure 40-211. Example of Up Counting When the Value of CNTIN Is Greater Than the Value of MOD

40.4.3.2 Up-Down Counting

Up-down counting is selected when (QUADEN= 0) and (CPWMS = 1).

CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

The FTM period when using up-down counting is 2 × (MOD – CNTIN) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to (MOD – 1).

If (CNTIN = 0x0000), the FTM counting is equivalent to TPM up-down counting (that is, up-down and unsigned counting) (see the following figure).

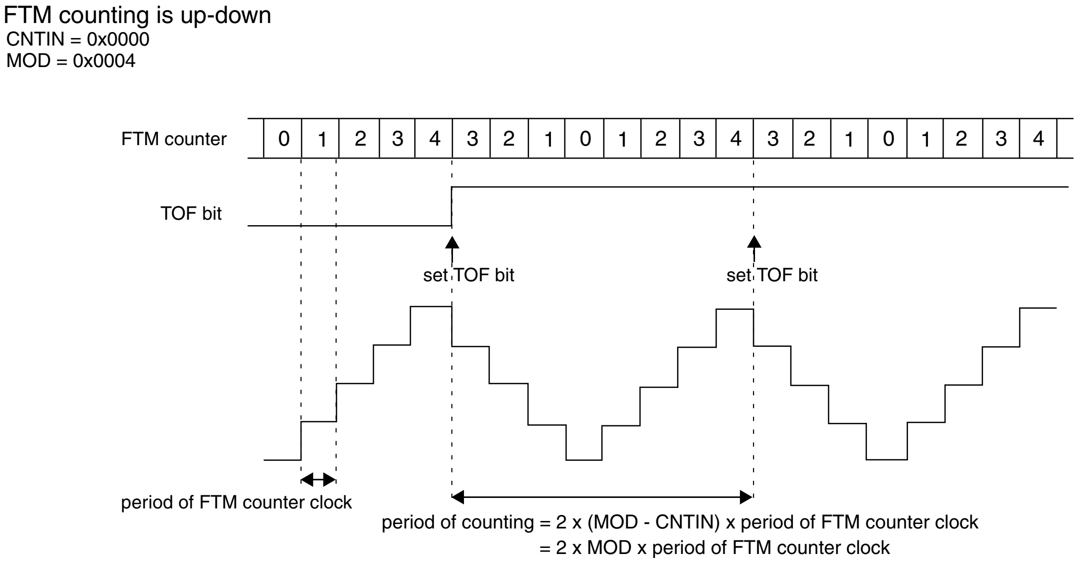

FTM counting is up-down
CNTIN = 0x0000
MOD = 0x0004
FTM counter
0 1 2 3 4 3 2 1 0 1 2 3 4 3 2 1 0 1 2 3 4
TOF bit
set TOF bit
set TOF bit
period of FTM counter clock
period of counting = 2 x (MOD - CNTIN) x period of FTM counter clock
= 2 x MOD x period of FTM counter clock

Figure 40-212. Example of Up-Down Counting When CNTIN = 0x0000

Note

It is expected that the up-down counting be used only with CNTIN = 0x0000.

40.4.3.3 Free Running Counter

If (FTMEN = 0) and (MOD = 0x0000 or MOD = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000 (see the following figure).

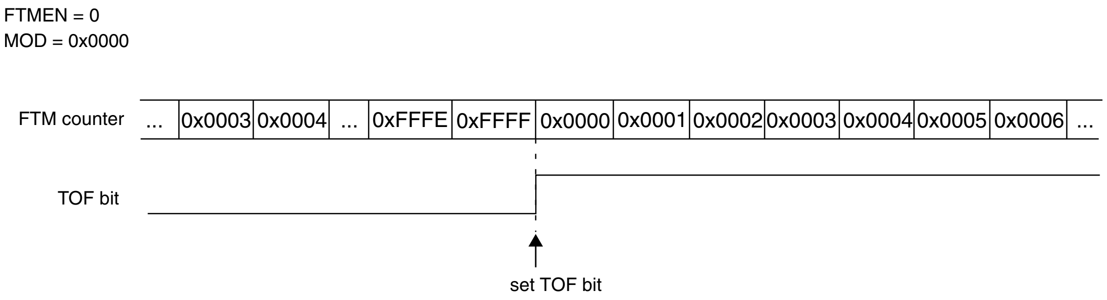

FTMEN = 0
MOD = 0x0000
FTM counter
... 0x0003 0x0004 ... 0xFFFE 0xFFFF 0x0000 0x0001 0x0002 0x0003 0x0004 0x0005 0x0006 ...
TOF bit
set TOF bit

Figure 40-213. Example When the FTM Counter Is a Free Running

If (FTMEN = 1), (QUADEN = 0), (CPWMS = 0), (CNTIN = 0x0000), and (MOD = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000.

40.4.3.4 Counter Reset

Any write to CNT resets the FTM counter to the value in the CNTIN register and the channels output to its initial value (except for channels in output compare mode).

The FTM counter synchronization (see FTM Counter Synchronization) can also be used to force the value of CNTIN into the FTM counter and the channels output to its initial value (except for channels in output compare mode).

40.4.3.5 When the TOF Bit is Set

The NUMTOF[4:0] bits define the number of times that the FTM counter overflow should occur before the TOF bit to be set. If NUMTOF[4:0] = 0x00, then the TOF bit is set at each FTM counter overflow.

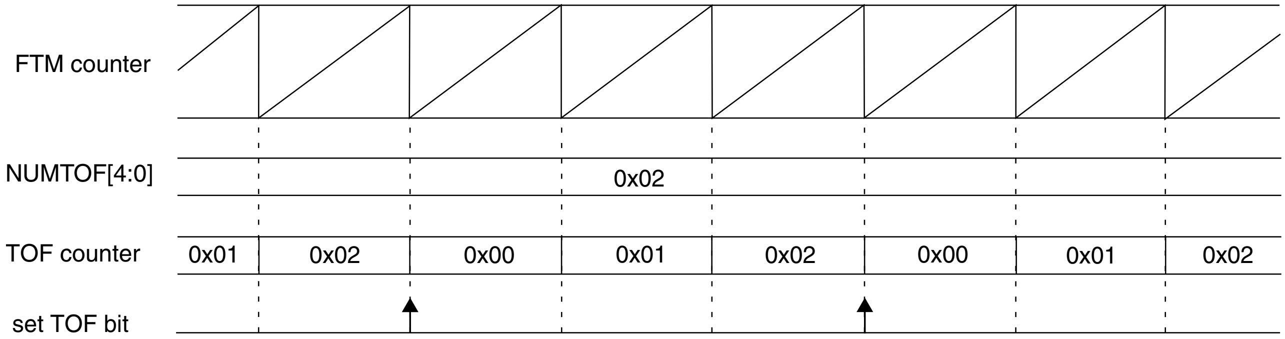


Figure 40-214. Periodic TOF When NUMTOF = 0x02

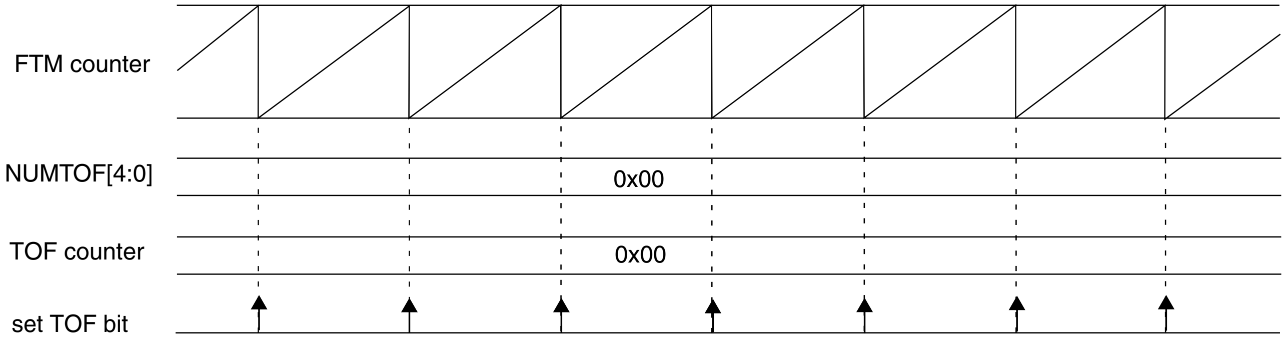


Figure 40-215. Periodic TOF When NUMTOF = 0x00

40.4.4 Input Capture Mode

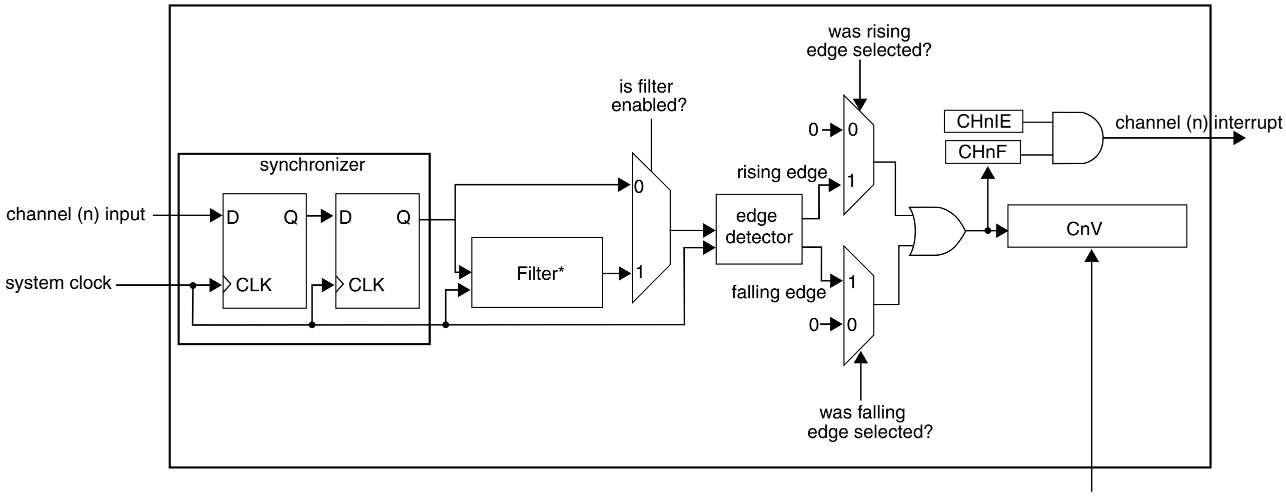
The input capture mode is selected when (DECAPEN = 0), (COMBINE = 0), (CPWMS = 0), (MSnB:MSnA = 0:0), and (ELSnB:ELSnA ≠ 0:0).

When a selected edge occurs on the channel input, the current value of the FTM 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 FTMxCHn 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 system clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register is ignored in input capture mode.

While in BDM, the input capture function works as configured. When a selected edge event occurs, the FTM counter value (which is frozen because of BDM) is captured into the CnV register and the CHnF bit is set.



* Filtering function is only available in the inputs of channel 0, 1, 2, and 3

Figure 40-216. Input Capture Mode

If the channel input does not have a filter enabled, then the input signal is always delayed 3 rising edges of the system clock (two rising edges to the synchronizer plus one more rising edge to the edge detector). In other words, the CHnF bit is set on the third rising edge of the system clock after a valid edge occurs on the channel input.

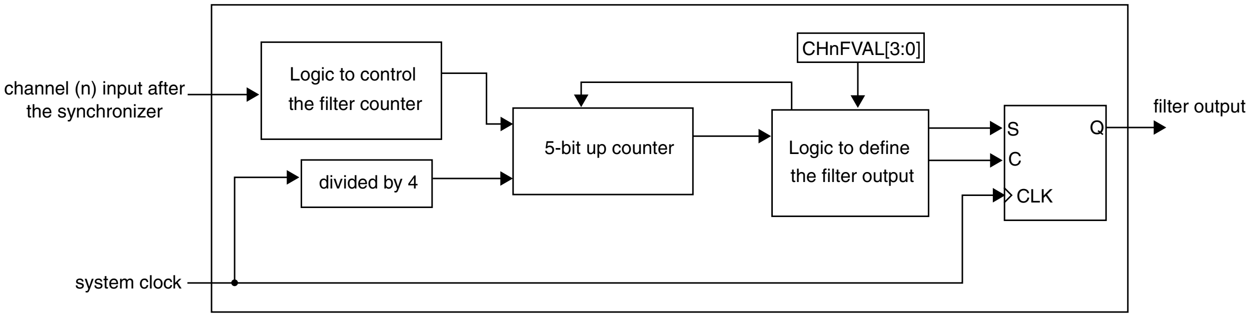
Note

It is expected that the input capture mode be used only with CNTIN = 0x0000.

40.4.4.1 Filter for Input Capture Mode

The filter function is only available on channels 0, 1, 2, and 3.

Firstly the input signal is synchronized by the system clock. Following synchronization, the input signal enters the filter block (see the following figure). When there is a state change in the input signal, the 5-bit counter is reset and starts counting up. As long as the new state is stable on the input, the counter continues to increment. If the 5-bit counter overflows (the counter exceeds the value of the CHnFVAL[3:0] bits), the state change of the input signal is validated. It is then transmitted as a pulse edge to the edge detector.



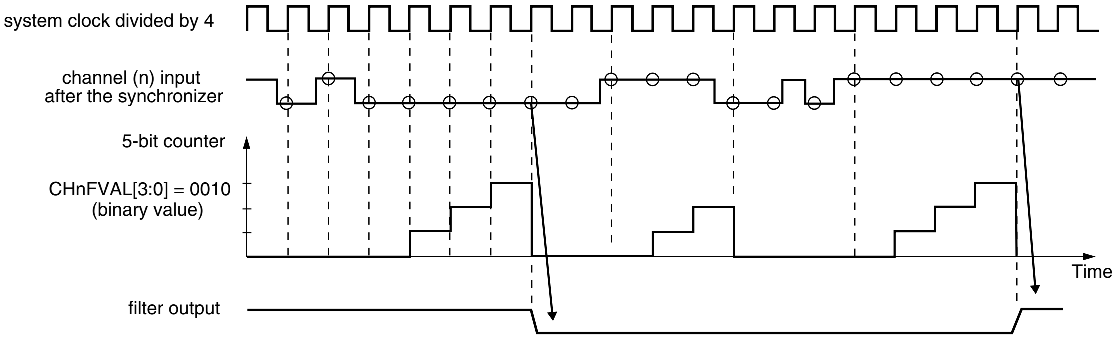
channel (n) input after the synchronizer
Logic to control the filter counter
divided by 4
5-bit up counter
CHnFVAL[3:0]
Logic to define the filter output
S
C
CLK
Q
filter output
system clock

Figure 40-217. Channel Input Filter

If the opposite edge appears on the input signal before validation (counter overflow), 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] bits (× 4 system clocks) is regarded as a glitch and is not passed on to the edge detector. 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 3 rising edges of the system 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 4 rising edges of the system clock (two rising edges to the synchronizer, one rising edge to the filter output plus one more to the edge detector). In other words, CHnF is set (4 + 4 × CHnFVAL[3:0]) system clock periods after a valid edge occurs on the channel input.

The clock for the 5-bit counter in the channel input filter is the system clock divided by 4.



system clock divided by 4
channel (n) input after the synchronizer
5-bit counter
CHnFVAL[3:0] = 0010 (binary value)
Time
filter output

Figure 40-218. Channel Input Filter Example

40.4.5 Output Compare Mode

The output compare mode is selected when (DECAPEN = 0), (COMBINE = 0), (CPWMS = 0), and (MSnB:MSnA = 0:1).

In output compare mode, the FTM 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.

When a channel is initially configured to toggle mode, the previous value of the channel output is held until the first output compare event occurs.

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnV).

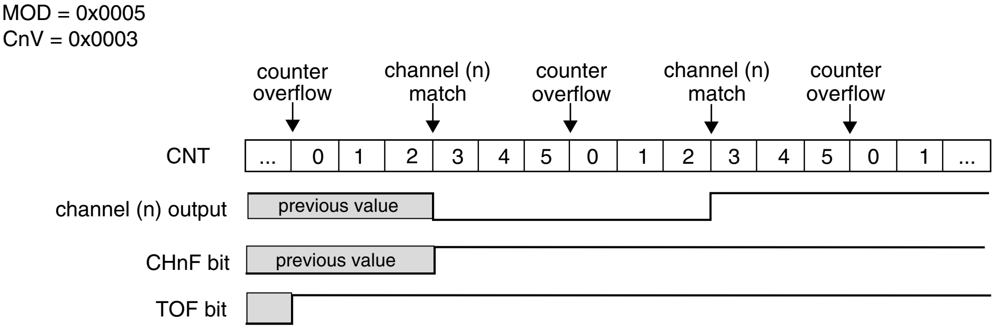


Figure 40-219. Example of the Output Compare Mode when the Match Toggles the Channel Output

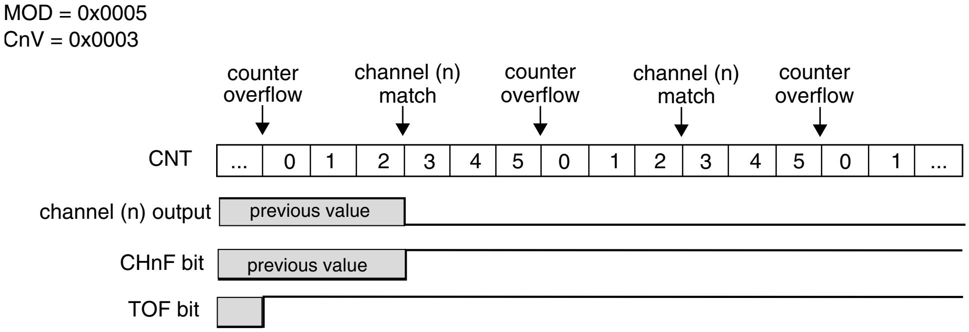


Figure 40-220. Example of the Output Compare Mode when the Match Clears the Channel Output

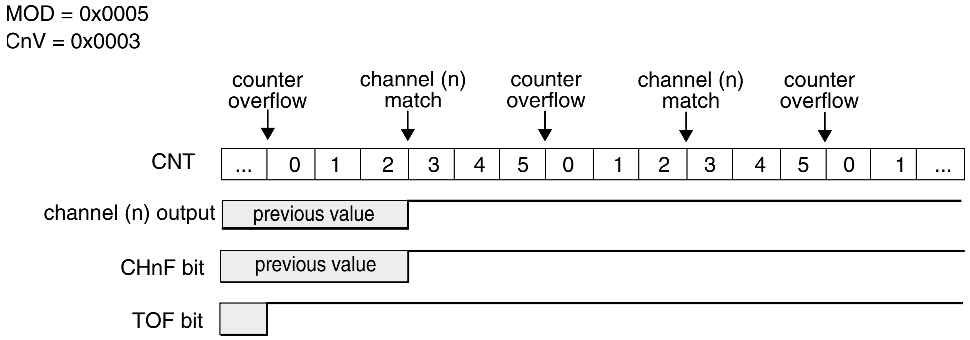


Figure 40-221. Example of the Output Compare Mode when the Match Sets the Channel Output

It is possible to use the output compare mode with (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 and controlled by FTM.

Note

It is expected that the output compare mode be used only with CNTIN = 0x0000.

40.4.6 Edge-Aligned PWM (EPWM) Mode

The edge-aligned mode is selected when (QUADEN = 0), (DECAPEN = 0), (COMBINE = 0), (CPWMS = 0), and (MSnB = 1).

The EPWM period is determined by (MOD – CNTIN + 0x0001) and the pulse width (duty cycle) is determined by (CnV – CNTIN).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM 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 FTM.

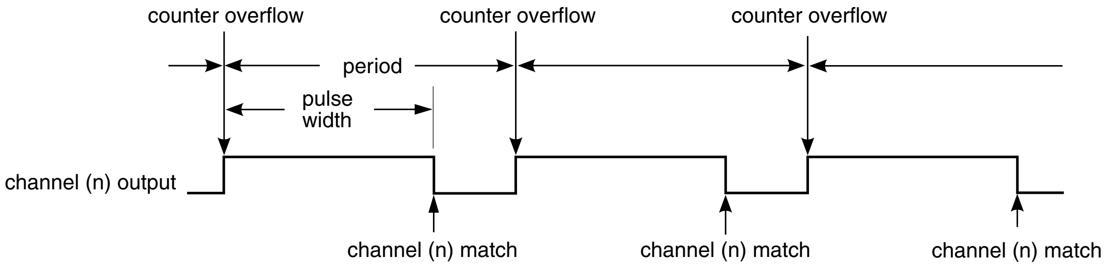


Figure 40-222. EPWM Period and Pulse Width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) 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 controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow (when the CNTIN register value is are loaded into the FTM counter), and it is forced low at the channel (n) match (FTM counter = CnV) (see the following figure).

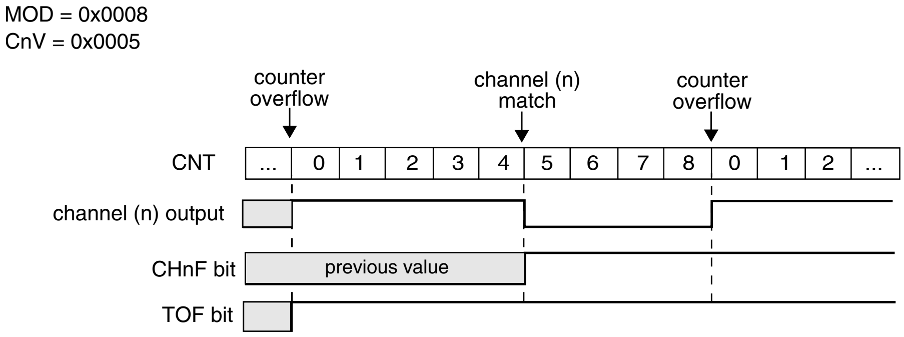


Figure 40-223. EPWM Signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the counter overflow (when the CNTIN register value is loaded into the FTM counter), and it is forced high at the channel (n) match (FTM counter = CnV) (see the following figure).

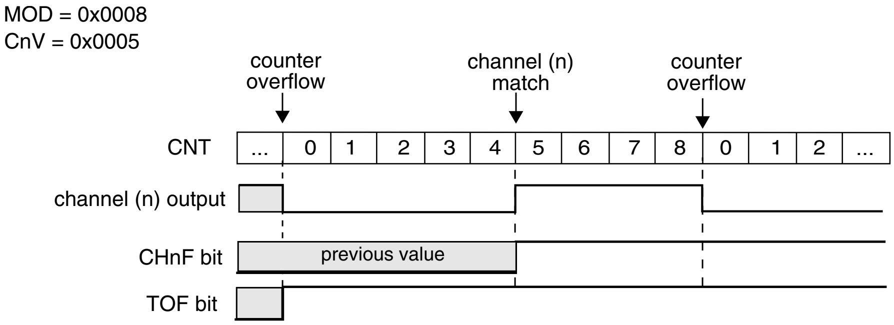


Figure 40-224. EPWM Signal with ELSnB:ELSnA = X:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHnF bit is not set even when there is the channel (n) match. If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set even when there is the channel (n) match. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

Note

It is expected that the EPWM mode be used only with CNTIN = 0x0000.

40.4.7 Center-Aligned PWM (CPWM) Mode

The center-aligned mode is selected when (QUADEN = 0), (DECAPEN = 0), (COMBINE = 0), and (CPWMS = 1).

The CPWM pulse width (duty cycle) is determined by 2 × (CnV – CNTIN) and the period is determined by 2 × (MOD – CNTIN)(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 FTM counter counts up until it reaches MOD and then counts down until it reaches CNTIN.

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnV) when the FTM counting is down (at the begin of the pulse width) and when the FTM 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 aligned with the value of CNTIN.

The other channel modes are not compatible with the up-down counter (CPWMS = 1). Therefore, all FTM channels must be used in CPWM mode when (CPWMS = 1).

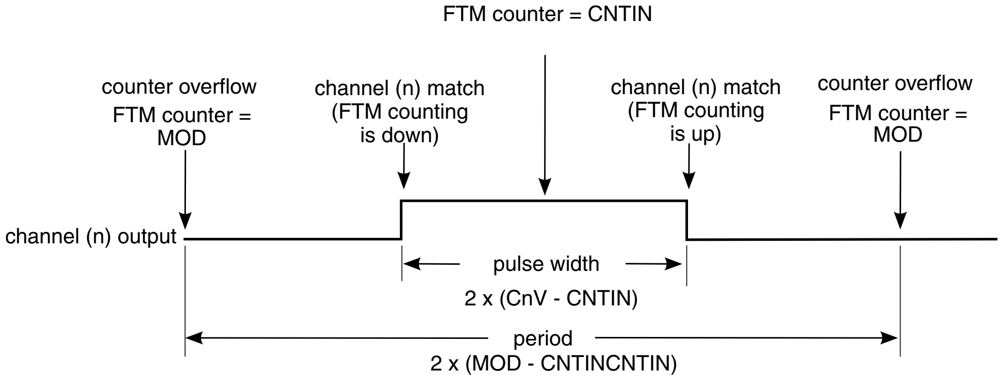


Figure 40-225. CPWM Period and Pulse Width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the FTM 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 controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (FTM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up (see the following figure).

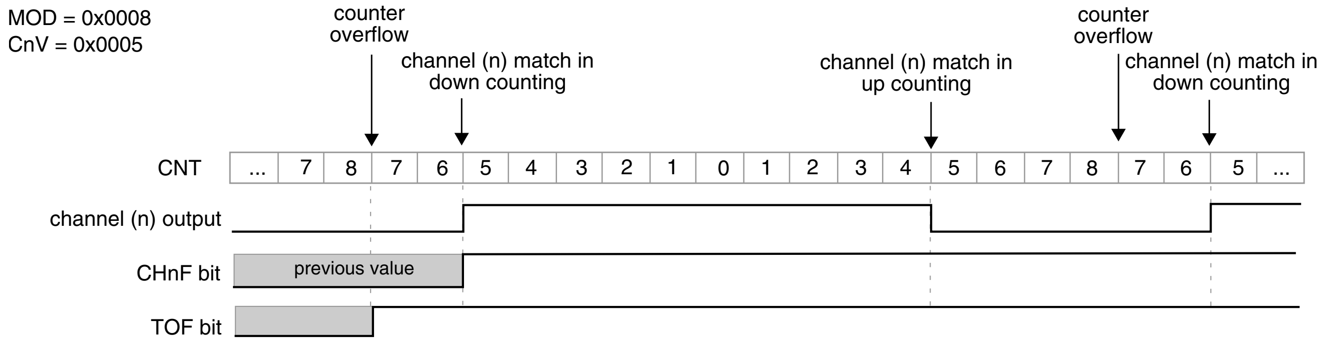


Figure 40-226. CPWM Signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the channel (n) match (FTM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up (see the following figure).

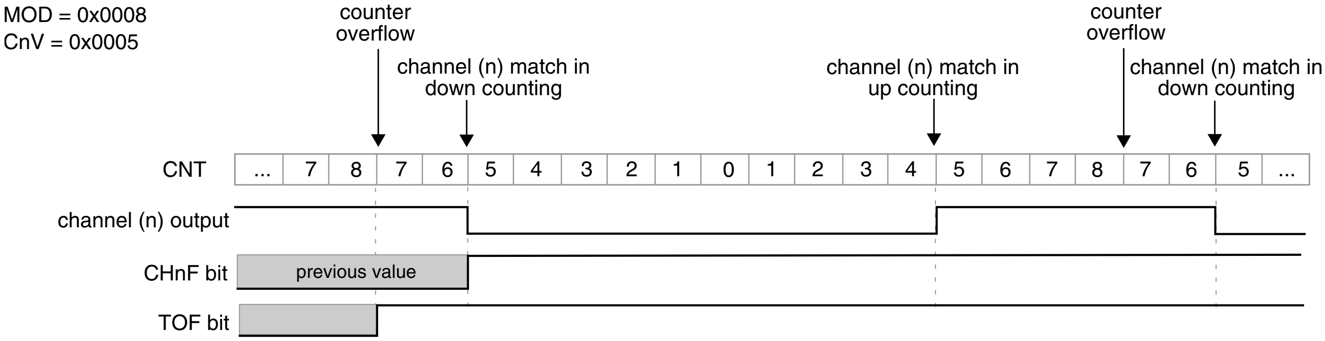


Figure 40-227. CPWM Signal with ELSnB:ELSnA = X:1

If (CnV = 0x0000) or (CnV is a negative value, that is, CnV[15] = 1) then the channel (n) output is a 0% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match.

If (CnV is a positive value, that is, CnV[15] = 0), (CnV ≥ MOD), and (MOD ≠ 0x0000), then the channel (n) output is a 100% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match. This implies that the usable range of periods set by MOD is 0x0001 through 0x7FFE (0x7FFF if you do not need to generate a 100% duty cycle CPWM signal). This is not a significant limitation because the resulting period is much longer than required for normal applications.

The CPWM mode must not be used when the FTM counter is a free running counter.

Note

It is expected that the CPWM mode be used only with CNTIN = 0x0000.

40.4.8 Combine Mode

The combine mode is selected when (FTMEN = 1), (QUADEN = 0), (DECAPEN = 0), (COMBINE = 1), and (CPWMS = 0).

In combine mode, the channel (n) (an even channel) and channel (n+1) (the adjacent odd channel) are combined to generate a PWM signal in the channel (n) output.

In the combine mode, the PWM period is determined by (MOD − CNTIN + 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 (FTM 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 (FTM counter = C(n+1)V).

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n +1)V). It is forced high at the channel (n) match (FTM counter = C(n)V)(see the following figure).

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced high at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n +1)V). It is forced low at the channel (n) match (FTM counter = C(n)V)(see the following figure).

In combine mode, the ELS(n+1)B and ELS(n+1)A bits are not used in the generation of the channels (n) and (n+1) output. However, if (ELSnB:ELSnA = 0:0) then the channel (n) output is not controlled by FTM, and if (ELS(n+1)B:ELS(n+1)A = 0:0) then the channel (n+1) output is not controlled by FTM.

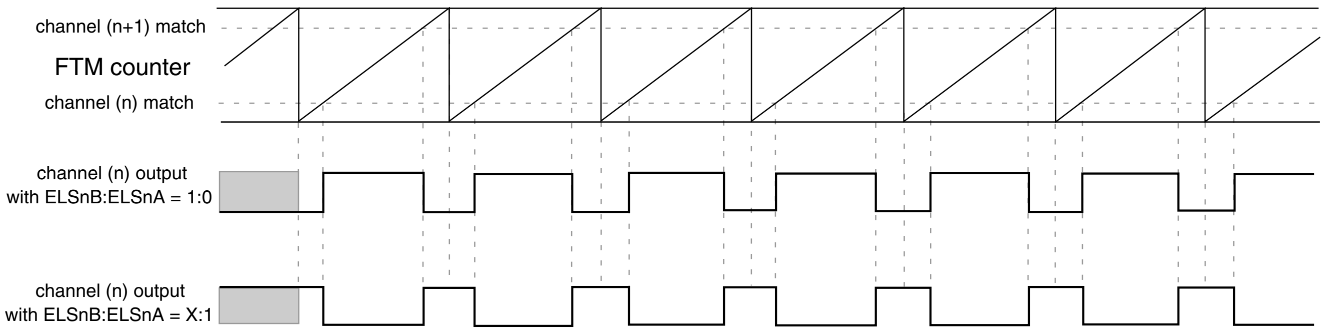


Figure 40-228. Combine Mode

The following figures illustrate the PWM signals generation using combine mode.

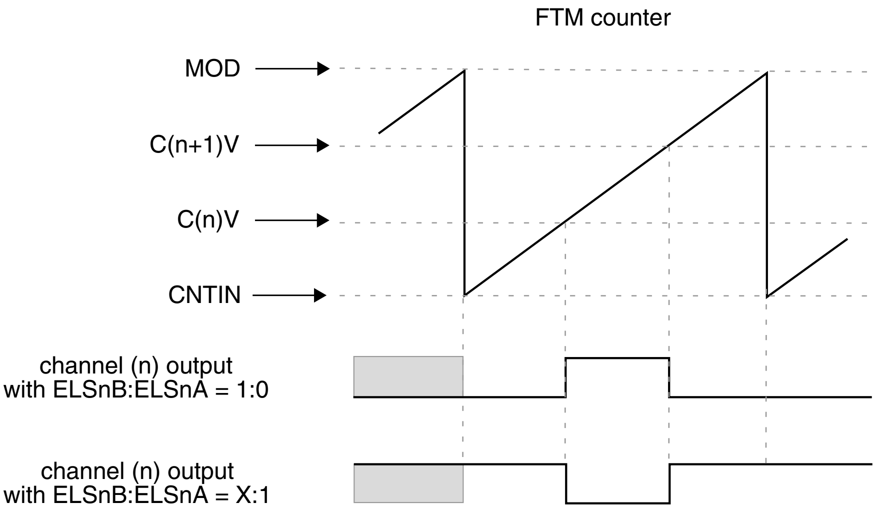


Figure 40-229. Channel (n) Output If (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V < C(n+1)V)

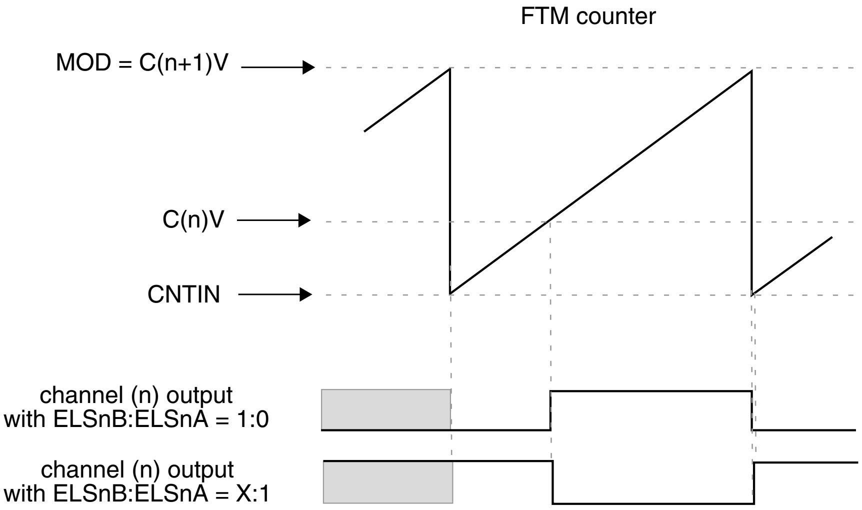


Figure 40-230. Channel (n) Output If (CNTIN < C(n)V < MOD) and (C(n+1)V = MOD)

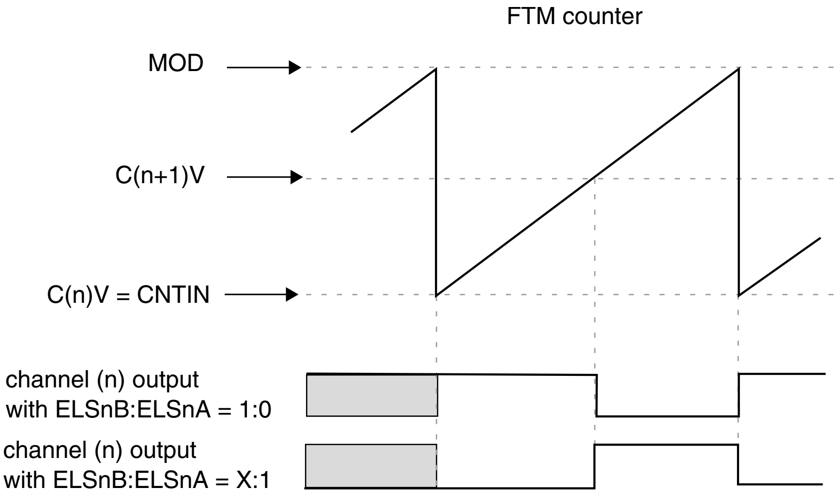


Figure 40-231. Channel (n) Output If (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD)

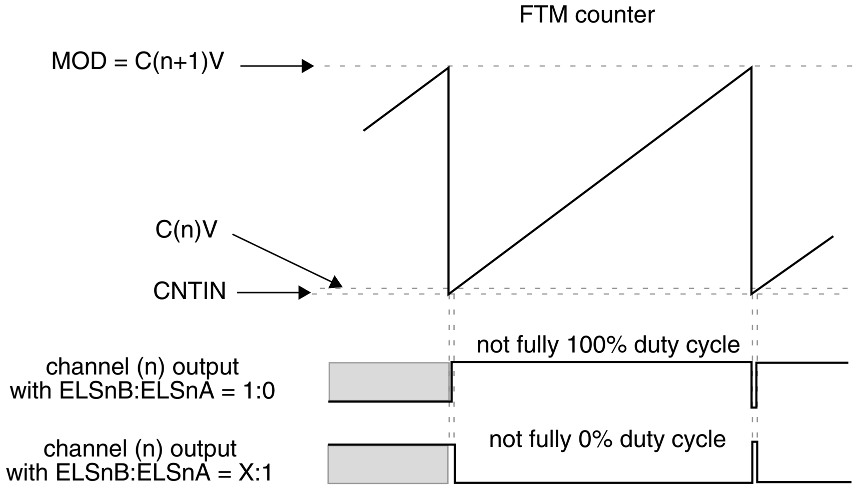


Figure 40-232. Channel (n) Output If (CNTIN < C(n)V < MOD) and (C(n)V is Almost Equal to CNTIN) and (C(n+1)V = MOD)

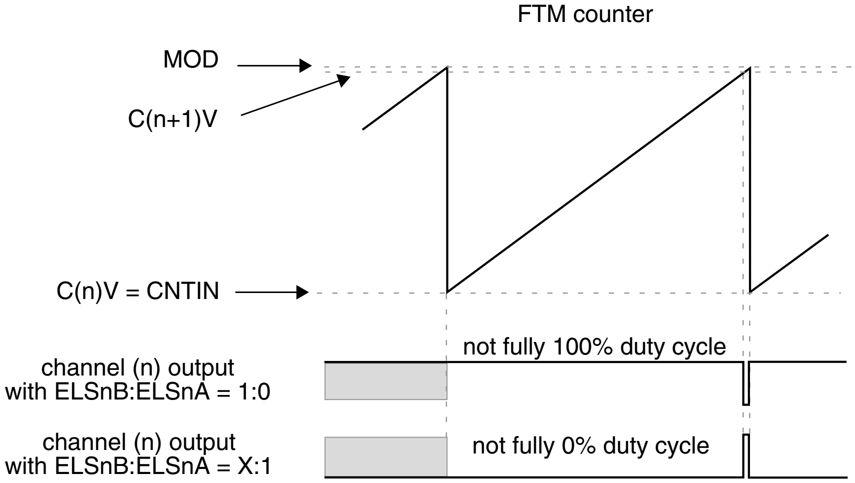


Figure 40-233. Channel (n) Output If (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD) and (C(n+1)V is Almost Equal to MOD)

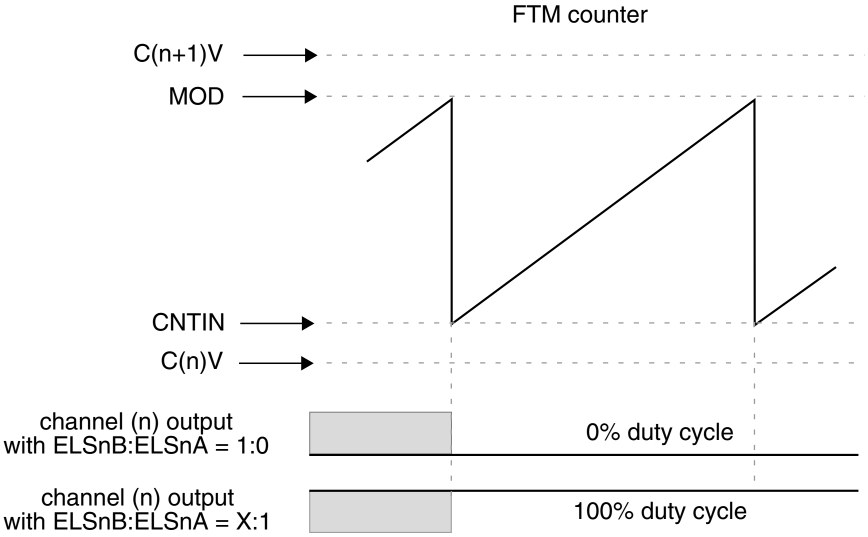


Figure 40-234. Channel (n) Output If C(n)V and C(n+1)V Are Not Between CNTIN and MOD

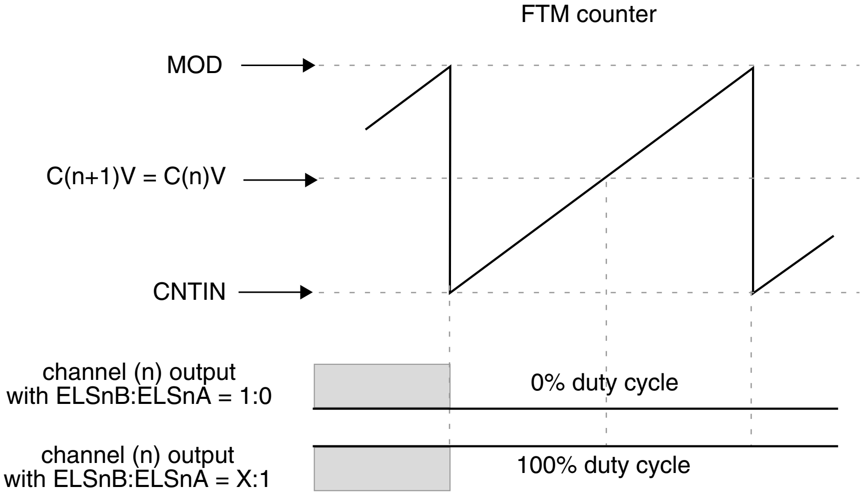


Figure 40-235. Channel (n) Output If (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V = C(n+1)V)

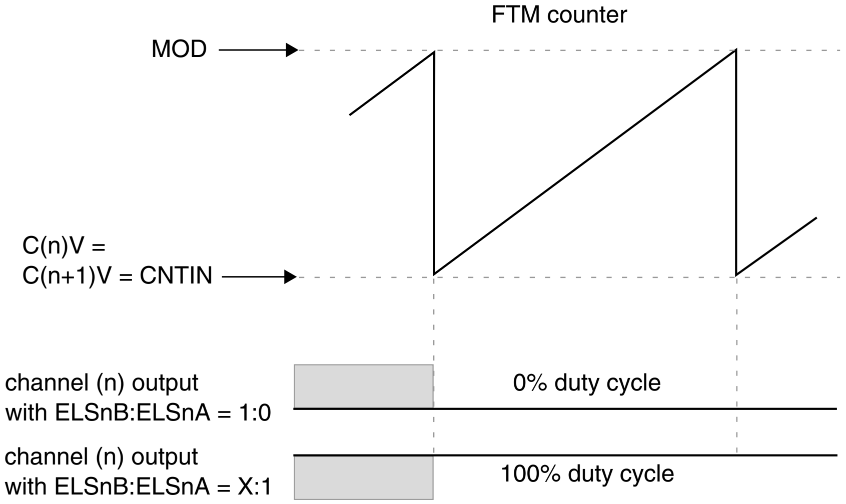


Figure 40-236. Channel (n) Output If (C(n)V = C(n+1)V = CNTIN)

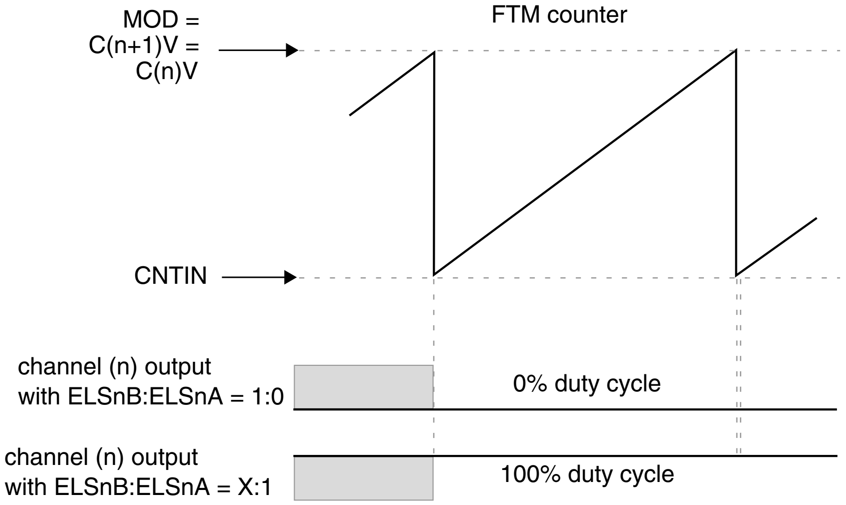


Figure 40-237. Channel (n) Output If (C(n)V = C(n+1)V = MOD)

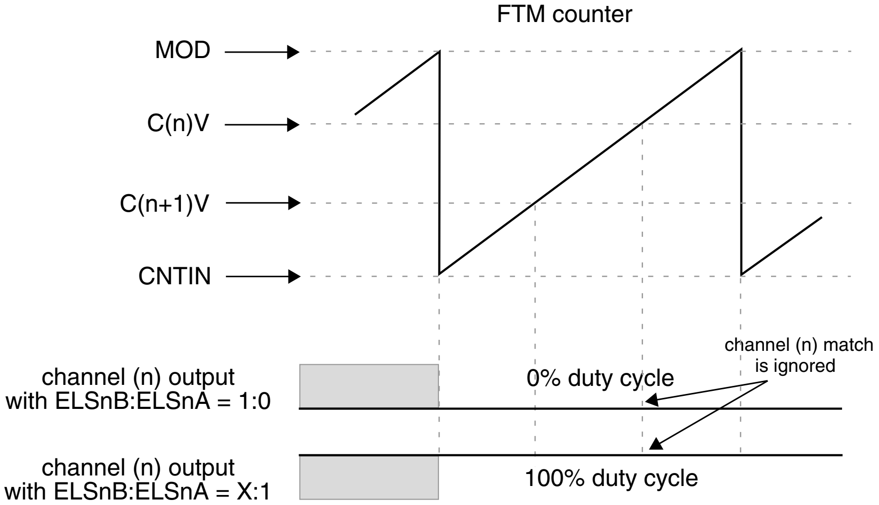


Figure 40-238. Channel (n) Output If (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V > C(n+1)V)

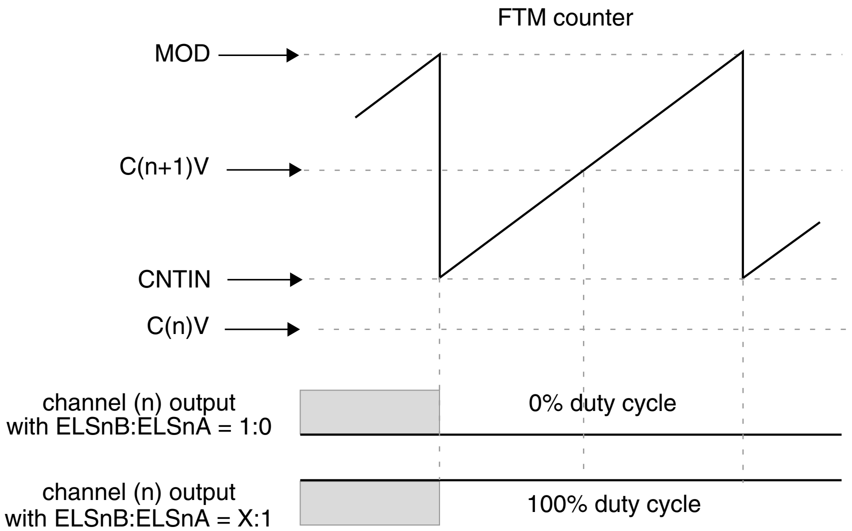


Figure 40-239. Channel (n) Output If (C(n)V < CNTIN) and (CNTIN < C(n+1)V < MOD)

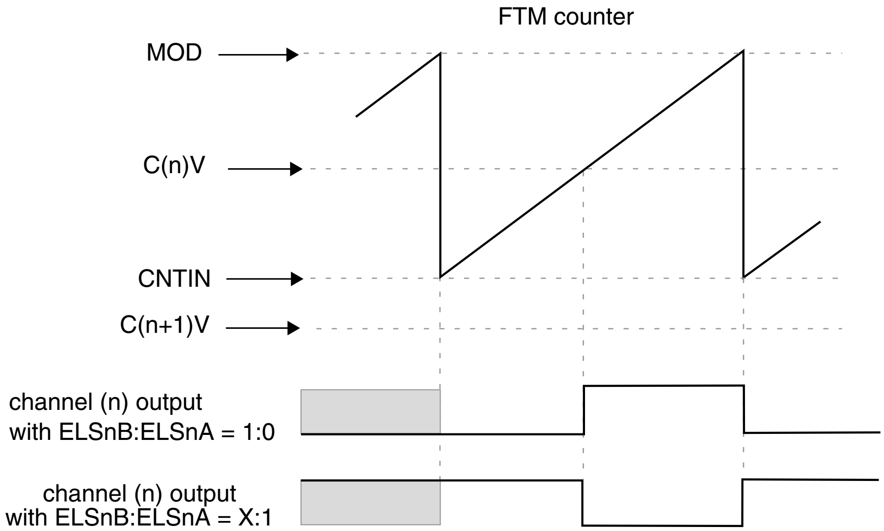


Figure 40-240. Channel (n) Output If (C(n+1)V < CNTIN) and (CNTIN < C(n)V < MOD)

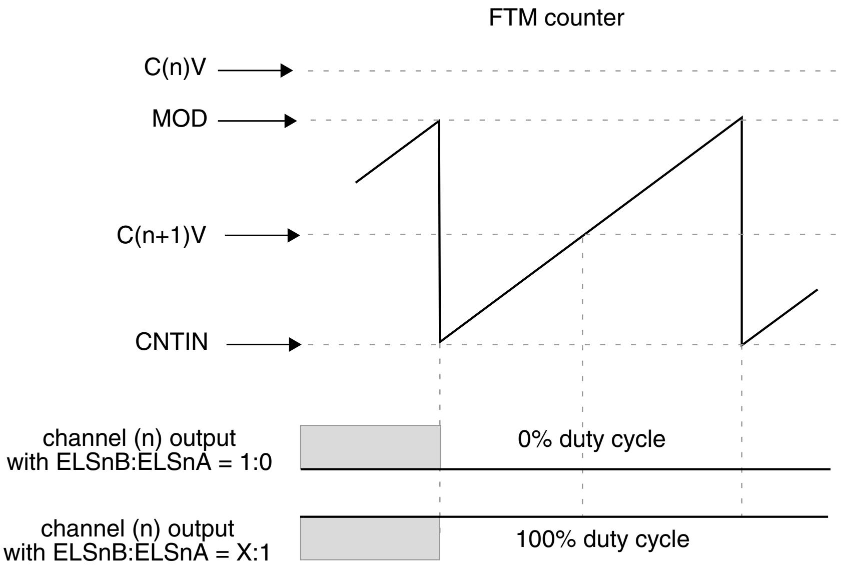


Figure 40-241. Channel (n) Output If (C(n)V > MOD) and (CNTIN < C(n+1)V < MOD)

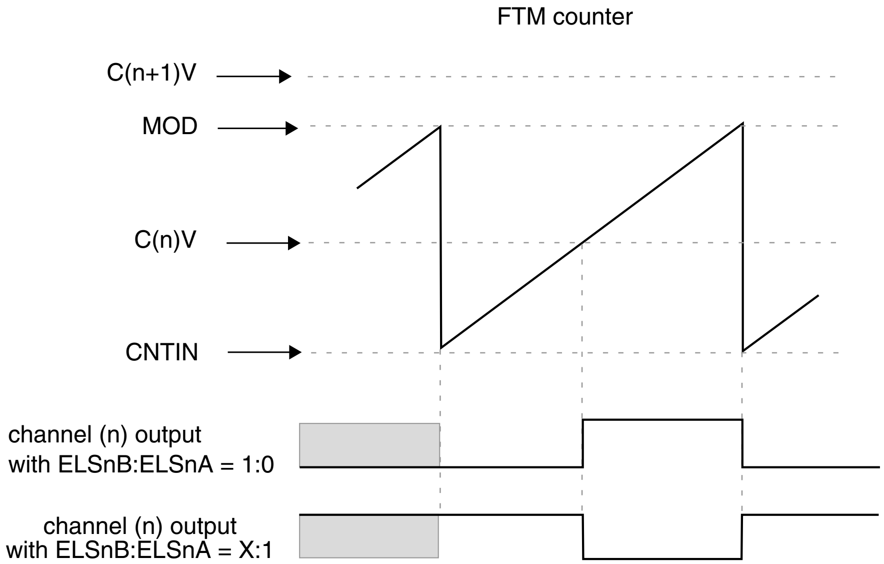


Figure 40-242. Channel (n) Output If (C(n+1)V > MOD) and (CNTIN < C(n)V < MOD)

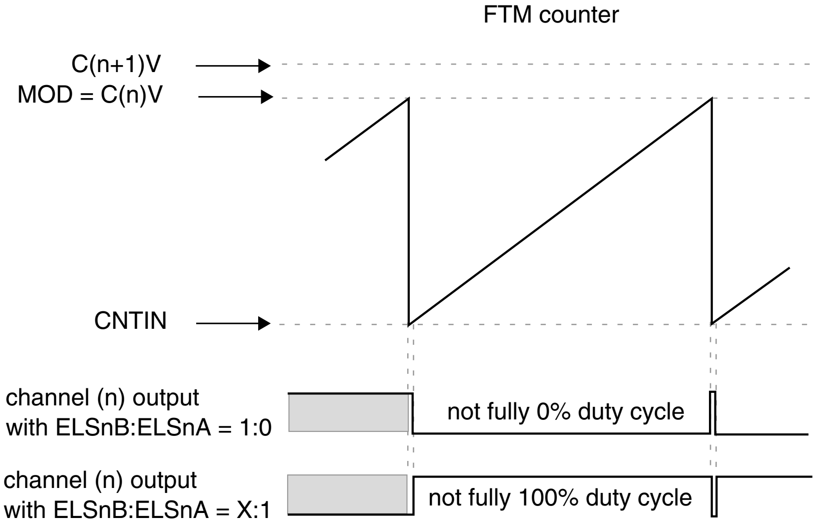


Figure 40-243. Channel (n) Output If (C(n+1)V > MOD) and (CNTIN < C(n)V = MOD)

40.4.8.1 Asymmetrical PWM

In combine mode, the control of the PWM signal first edge (when the channel (n) match occurs, that is, FTM counter = C(n)V) is independent of the control of the PWM signal second edge (when the channel (n+1) match occurs, that is, FTM counter = C(n+1)V). So, combine mode allows the generation of asymmetrical PWM signals.

40.4.9 Complementary Mode

The complementary mode is selected when (FTMEN = 1), (QUADEN = 0), (DECAPEN = 0), (COMBINE = 1), (CPWMS = 0), and (COMP = 1).

In complementary mode the channel (n+1) output is the inverse of the channel (n) output.

If (FTMEN = 1), (QUADEN = 0), (DECAPEN = 0), (COMBINE = 1), (CPWMS = 0), and (COMP = 0), then the channel (n+1) output is the same as the channel (n) output.

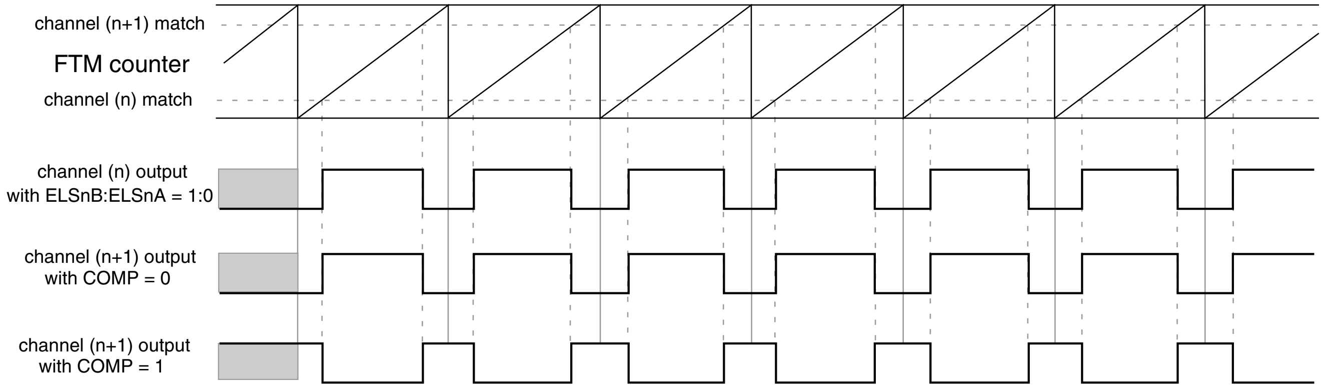


Figure 40-244. Channel (n+1) Output in Complementary Mode with (ELSnB:ELSnA = 1:0)

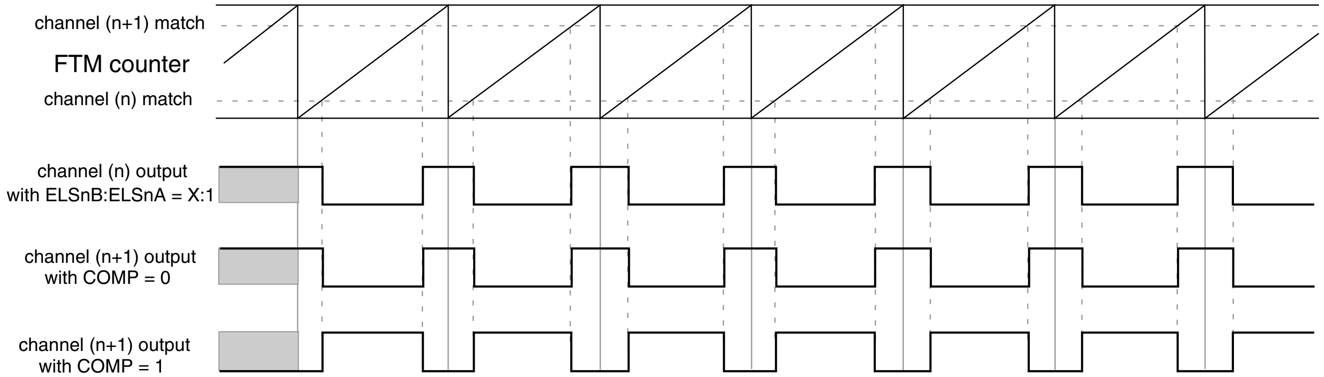


Figure 40-245. Channel (n+1) Output in Complementary Mode with (ELSnB:ELSnA = X:1)

40.4.10 Registers Updated from Write Buffers

40.4.10.1 CNTIN Register Update

If (CLKS[1:0] = 0:0) then CNTIN register is updated when CNTIN register is written (independent of FTMEN bit).

If (FTMEN = 0) or (CNTINC = 0) then CNTIN register is updated at the next system clock after CNTIN was written.

If (FTMEN = 1), (SYNCMODE = 1) and (CNTINC = 1) then CNTIN register is updated by the CNTIN register synchronization (CNTIN Register Synchronization).

40.4.10.2 MOD Register Update

If (CLKS[1:0] = 0:0) then MOD register is updated when MOD register is written (independent of FTMEN bit).

If (CLKS[1:0] ≠ 0:0 and FTMEN = 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 FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000.
- If the selected mode is CPWM then MOD register is updated after MOD register was written and the FTM counter changes from MOD to (MOD – 0x0001).

If (CLKS[1:0] ≠ 0:0 and FTMEN = 1) then MOD register is updated by the MOD register synchronization (MOD Register Synchronization).

40.4.10.3 CnV Register Update

If (CLKS[1:0] = 0:0) then CnV register is updated when CnV register is written (independent of FTMEN bit).

If (CLKS[1:0] ≠ 0:0 and FTMEN = 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 FTM counter change (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 FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000.
- If the selected mode is CPWM then CnV register is updated after CnV register was written and the FTM counter changes from MOD to (MOD – 0x0001).

If (CLKS[1:0] ≠ 0:0 and FTMEN = 1) then CnV register is updated according to the selected mode, that is:.

- If the selected mode is output compare then CnV register is updated according to the SYNCEN bit. If (SYNCEN = 0) then CnV register is updated after CnV register was written at the next change of the FTM counter (end of the prescaler counting). If (SYNCEN = 1) then CnV register is updated by the CnV register synchronization (C(n)V and C(n+1)V Register Synchronization).
- If the selected mode is not output compare and (SYNCEN = 1) then CnV register is updated by the CnV register synchronization (C(n)V and C(n+1)V Register Synchronization).

40.4.11 PWM Synchronization

The PWM synchronization provides an opportunity to update the MOD, CNTIN, CnV, OUTMASK, INVCTRL and SWOCTRL registers with their buffered value and force the FTM counter to the CNTIN register value.

Note

- It is expected that the PWM synchronization be used only in combine mode.
- The legacy PWM synchronization (SYNCMODE = 0) is a subset of the enhanced PWM synchronization (SYNCMODE = 1). Thus, it is expected that only the enhanced PWM synchronization be used.

40.4.11.1 Hardware Trigger

Three hardware trigger signal inputs of the FTM module are enabled when TRIGn = 1 (where n = 0, 1 or 2 corresponding to each one of the input signals, respectively). The hardware trigger input n is synchronized by the system clock. The PWM synchronization with hardware trigger is initiated when a rising edge is detected at the enabled hardware trigger inputs.

If (HWTRIGMODE = 0) then the TRIGn bit is cleared when 0 is written to it or when the trigger n event is detected.

In this case if two or more hardware triggers are enabled (for example, TRIG0 and TRIG1 = 1) and only trigger 1 event occurs then only TRIG1 bit is cleared. If a trigger n event occurs together with a write setting TRIGn bit then the synchronization is initiated, but TRIGn bit remains set due to the write operation.

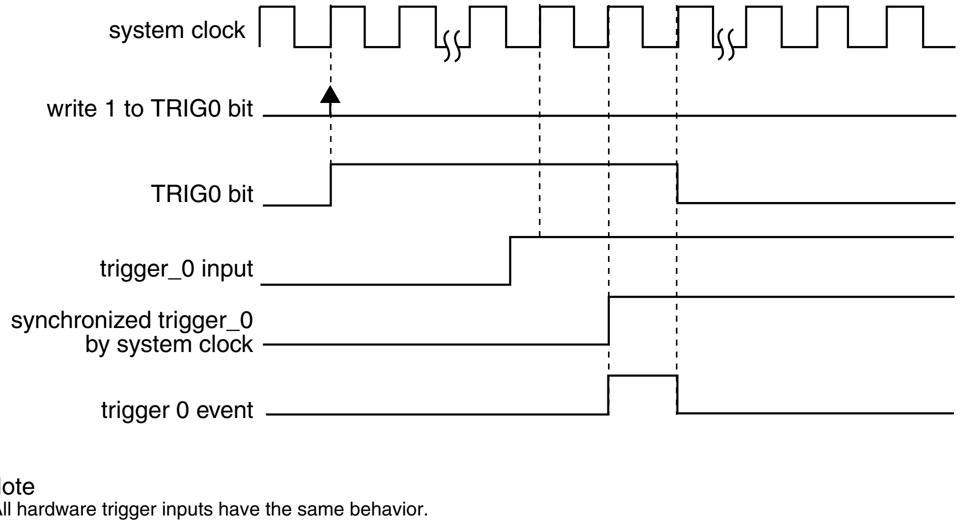

system clock
write 1 to TRIG0 bit
TRIG0 bit
trigger_0 input
synchronized trigger_0 by system clock
trigger 0 event
Note
All hardware trigger inputs have the same behavior.

Figure 40-246. Hardware Trigger Event with HWTRIGMODE = 0

If HWTRIGMODE = 1 then the TRIGn bit is only cleared when 0 is written to it.

NOTE

It is expected that the HWTRIGMODE bit be 1 only with enhanced PWM synchronization (SYNCMODE = 1).

40.4.11.2 Software Trigger

A software trigger event occurs when 1 is written to the SYNC[SWSYNC] bit. The SWSYNC bit is cleared when 0 is written to it or when the PWM synchronization (initiated by the software event) is completed.

If a new software trigger event occurs (write 1 to SWSYNC bit) together with the end of the previous synchronization (also initiated by the software trigger event) then this new synchronization is started and SWSYNC bit remains equal to 1.

If SYNCMODE = 0 then the SWSYNC bit is also cleared by FTM according to PWMSYNC and REINIT bits. In this case if (PWMSYNC = 1) or (PWMSYNC = 0 and REINIT = 0) then SWSYNC bit is cleared at the next selected loading point (Boundary Cycle and Loading Points) after that the software trigger event occurred (see the following figure). If (PWMSYNC = 0) and (REINIT = 1) then SWSYNC bit is cleared when the software trigger event occurs.

If SYNCMODE = 1 then the SWSYNC bit is also cleared by FTM according to the SWRSTCNT bit. If SWRSTCNT = 0 then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred (see the following figure). If SWRSTCNT = 1 then SWSYNC bit is cleared when the software trigger event occurs.

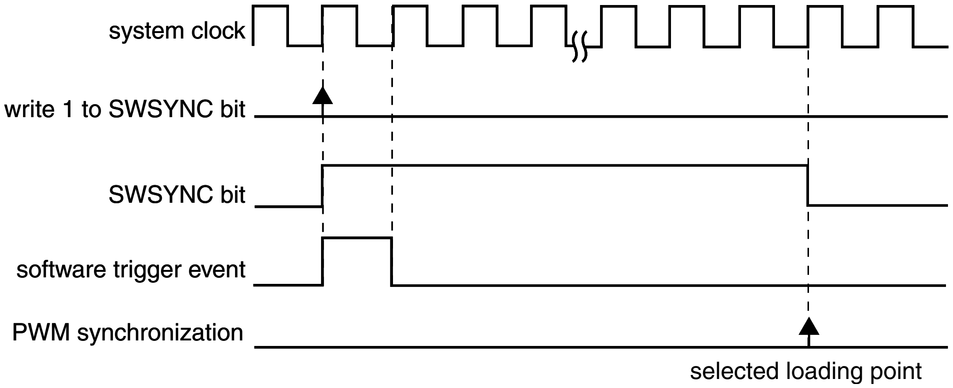


Figure 40-247. Software Trigger Event

40.4.11.3 Boundary Cycle and Loading Points

The boundary cycle definition is important for the loading points for the registers MOD, CNTIN and C(n)V.

In up counting mode (Up Counting) the boundary cycle is defined as when the counter wraps to its initial value (CNTIN). If in up-down counting mode (Up-Down Counting) then the boundary cycle is defined as when the counter turns from down to up counting and when from up to down counting.

The following figure shows the boundary cycles and the loading points. In the up counting mode, the loading points are enabled if one of CNTMIN or CTMAX bits are 1. In the up-down counting mode, the loading points are selected by CNTMIN and CNTMAX bits, as indicated in the figure. These loading points are safe places for register updates thus allowing a smooth transitions in PWM waveform generation.

For both counting modes if neither CNTMIN nor CNTMAX are 1 then the boundary cycles are not used as loading points for registers updates. See the register synchronization descriptions in the following sections for details.

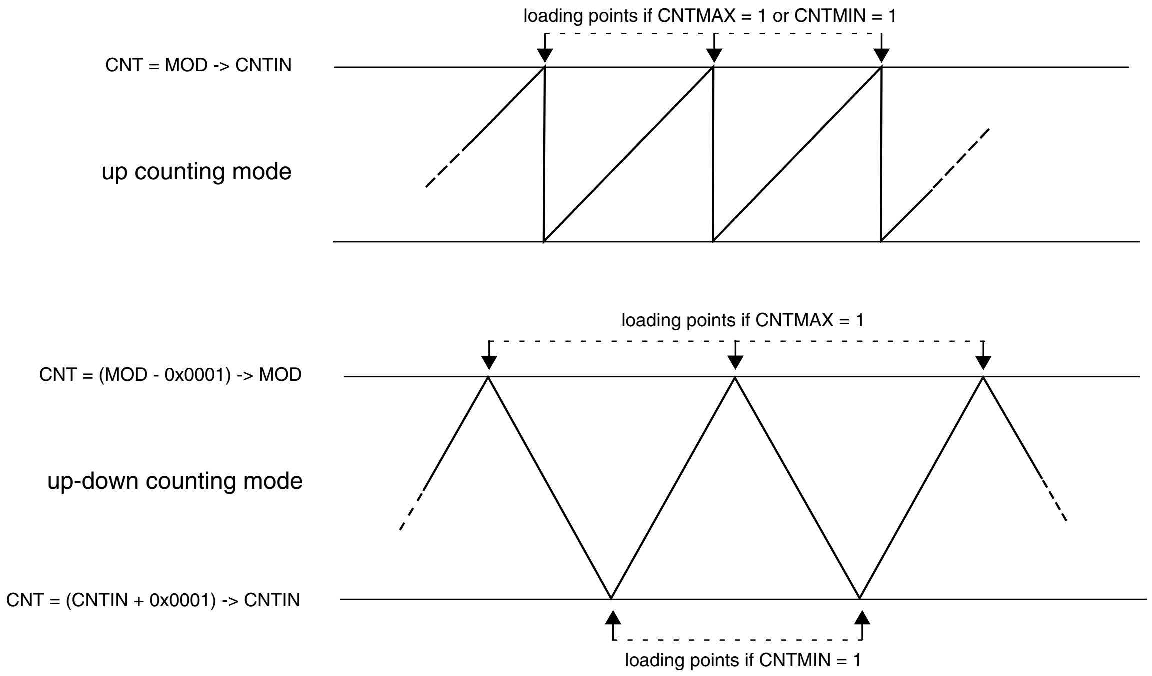

loading points if CNTMAX = 1 or CNTMIN = 1
CNT = MOD -> CNTIN
up counting mode
loading points if CNTMAX = 1
CNT = (MOD - 0x0001) -> MOD
up-down counting mode
CNT = (CNTIN + 0x0001) -> CNTIN
loading points if CNTMIN = 1

Figure 40-248. Boundary Cycles and Loading Points

40.4.11.4 MOD Register Synchronization

The MOD register synchronization updates the MOD register with its buffer value. This synchronization is enabled if (FTMEN = 1).

The MOD register synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, it is expected that the MOD register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the MOD register synchronization depends on SWWRBUF, SWRSTCNT, HWWRBUF and HWRSTCNT bits according to this flowchart:

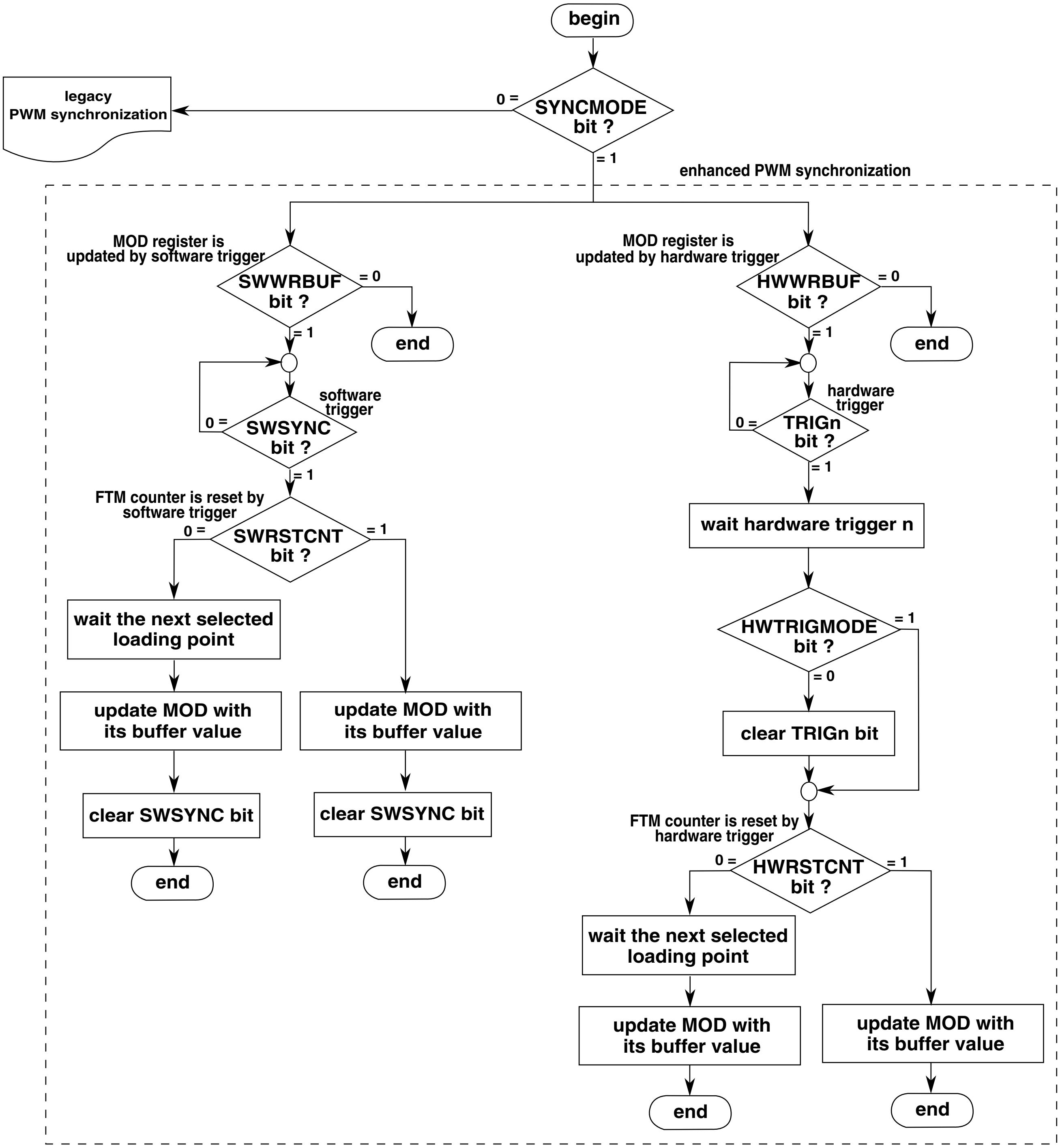

begin
SYNCMODE bit ?
0 =
legacy PWM synchronization
= 1
enhanced PWM synchronization
MOD register is updated by software trigger
SWWRBUF bit ?
= 0
end
= 1
software trigger
0 =
SWSYNC bit ?
= 1
FTM counter is reset by software trigger
0 =
SWRSTCNT bit ?
= 1
wait the next selected loading point
update MOD with its buffer value
clear SWSYNC bit
end
update MOD with its buffer value
clear SWSYNC bit
end
MOD register is updated by hardware trigger
HWWRBUF bit ?
= 0
end
= 1
hardware trigger
0 =
TRIGn bit ?
= 1
wait hardware trigger n
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
FTM counter is reset by hardware trigger
0 =
HWRSTCNT bit ?
= 1
wait the next selected loading point
update MOD with its buffer value
end
update MOD with its buffer value
end

Figure 40-249. MOD Register Synchronization Flowchart

In the case of legacy PWM synchronization, the MOD register synchronization depends on PWMSYNC and REINIT bits according to the following description.

If (SYNCMODE = 0), (PWMSYNC = 0) and (REINIT = 0) then this synchronization is made on the next selected loading point after an enabled trigger event takes place. If the trigger event was a software trigger then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger then the trigger enable bit (TRIGn) is cleared according to Hardware Trigger. Examples with software and hardware triggers follow.

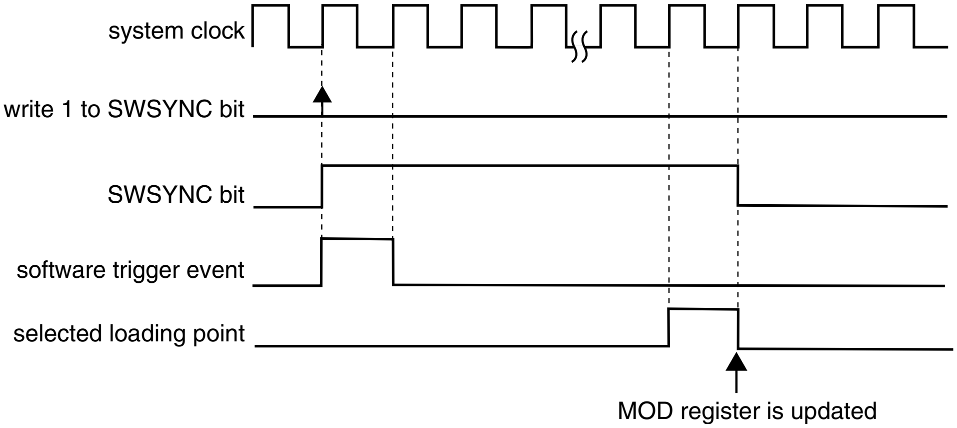


Figure 40-250. MOD Synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 0), and (Software Trigger Was Used)

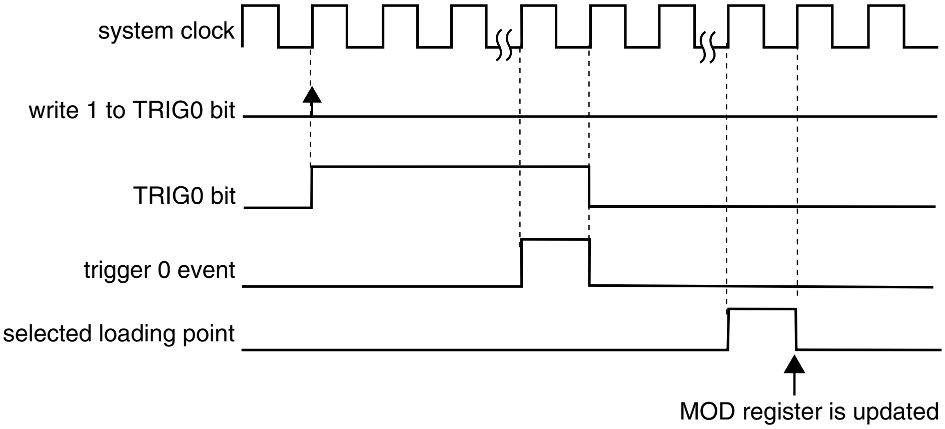


Figure 40-251. MOD Synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 0), and (a Hardware Trigger Was Used)

If (SYNCMODE = 0), (PWMSYNC = 0) and (REINIT = 1) then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger then the TRIGn bit is cleared according to Hardware Trigger. Examples with software and hardware triggers follow.

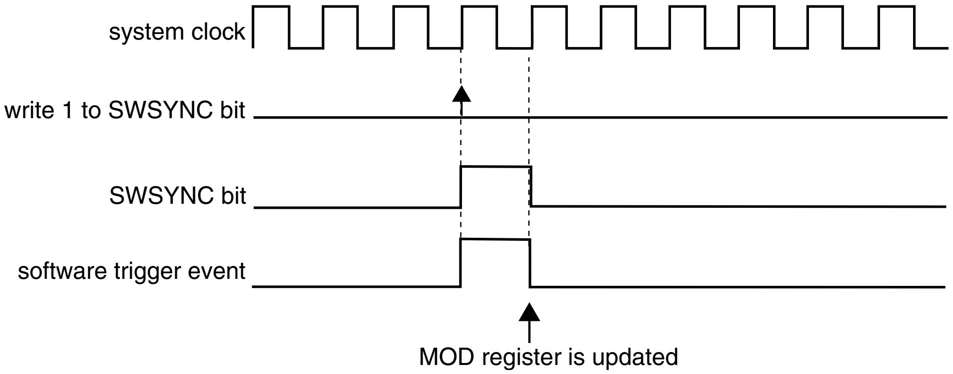


Figure 40-252. MOD Synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 1), and (Software Trigger Was Used)

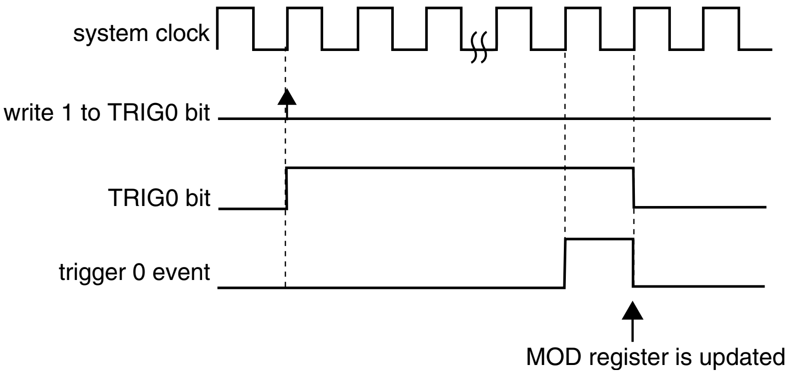


Figure 40-253. MOD Synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 1), and (a Hardware Trigger Was Used)

If (SYNCMODE = 0) and (PWMSYNC = 1) then this synchronization is made on the next selected loading point after the software trigger event takes place. The SWSYNC bit is cleared on the next selected loading point:

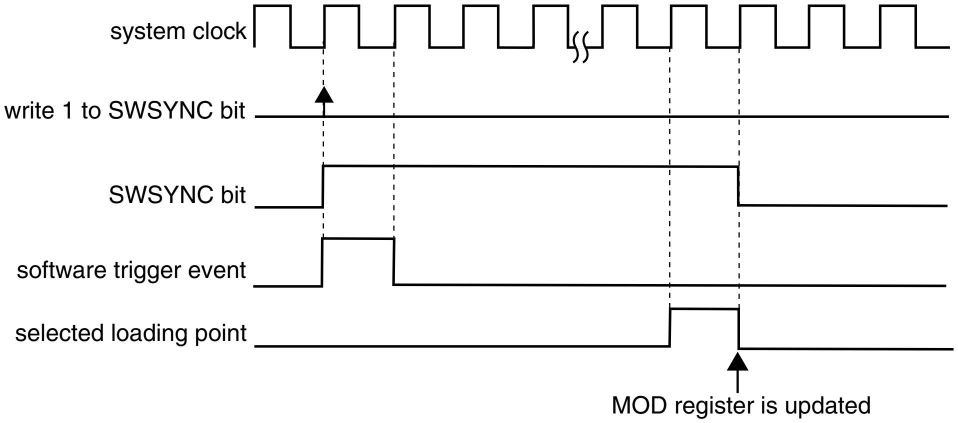


Figure 40-254. MOD Synchronization with (SYNCMODE = 0) and (PWMSYNC = 1)

40.4.11.5 CNTIN Register Synchronization

The CNTIN register synchronization updates the CNTIN register with its buffer value.

This synchronization is enabled if (FTMEN = 1), (SYNCMODE = 1) and (CNTINC = 1). The CNTIN register synchronization can be done only by the enhanced PWM synchronization (SYNCMODE = 1). The synchronization mechanism is the same as the MOD register synchronization done by the enhanced PWM synchronization (MOD Register Synchronization).

40.4.11.6 C(n)V and C(n+1)V Register Synchronization

The C(n)V and C(n+1)V registers synchronization updates the C(n)V and C(n+1)V registers with their buffer values.

This synchronization is enabled if (FTMEN = 1) and (SYNCEN = 1). The synchronization mechanism is the same as the MOD register synchronization (MOD Register Synchronization). However, it is expected that the C(n)V and C(n+1)V registers be synchronized only by the enhanced PWM synchronization (SYNCMODE = 1).

40.4.11.7 OUTMASK Register Synchronization

The OUTMASK register synchronization updates the OUTMASK register with its buffer value.

The OUTMASK register can be updated at each rising edge of system clock (SYNCHOM = 0), by the enhanced PWM synchronization (SYNCHOM = 1 and SYNCMODE = 1) or by the legacy PWM synchronization (SYNCHOM = 1 and SYNCMODE = 0). However, it is expected that the OUTMASK register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the OUTMASK register synchronization depends on SWOM and HWOM bits. See the following flowchart:

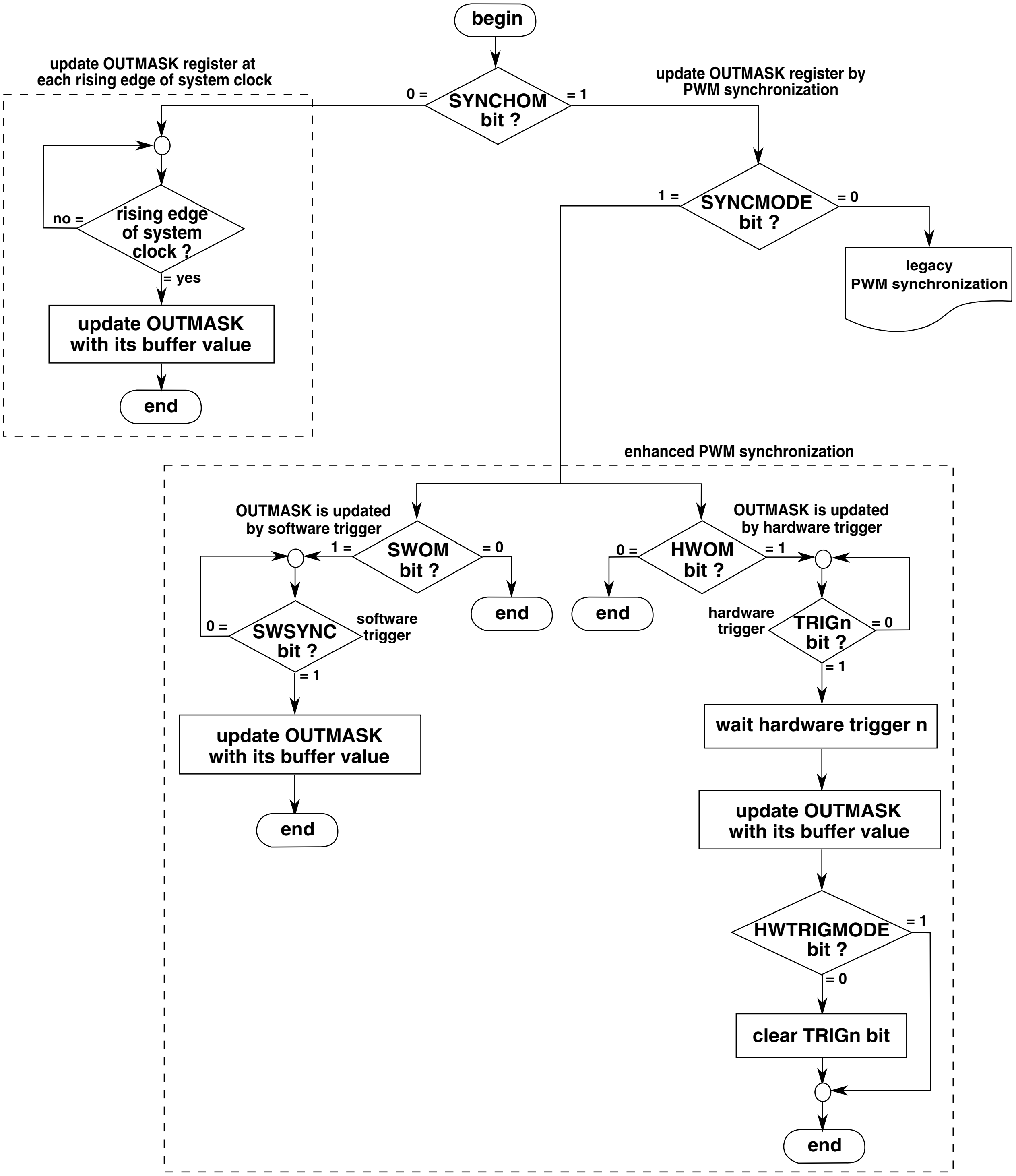

begin
update OUTMASK register at each rising edge of system clock
0 =
SYNCHOM bit ?
= 1
update OUTMASK register by PWM synchronization
no =
rising edge of system clock ?
= yes
update OUTMASK with its buffer value
end
1 =
SYNCMODE bit ?
= 0
legacy PWM synchronization
enhanced PWM synchronization
OUTMASK is updated by software trigger
1 =
SWOM bit ?
= 0
end
0 =
SWSYNC bit ?
software trigger
= 1
update OUTMASK with its buffer value
end
0 =
HWOM bit ?
= 1
end
OUTMASK is updated by hardware trigger
hardware trigger
TRIGn bit ?
= 0
= 1
wait hardware trigger n
update OUTMASK with its buffer value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 40-255. OUTMASK Register Synchronization Flowchart

In the case of legacy PWM synchronization, the OUTMASK register synchronization depends on PWMSYNC bit according to the following description.

If (SYNCMODE = 0), (SYNCHOM = 1) and (PWMSYNC = 0) then this synchronization is done on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger then the TRIGn bit is cleared according to Hardware Trigger. Examples with software and hardware triggers follow.

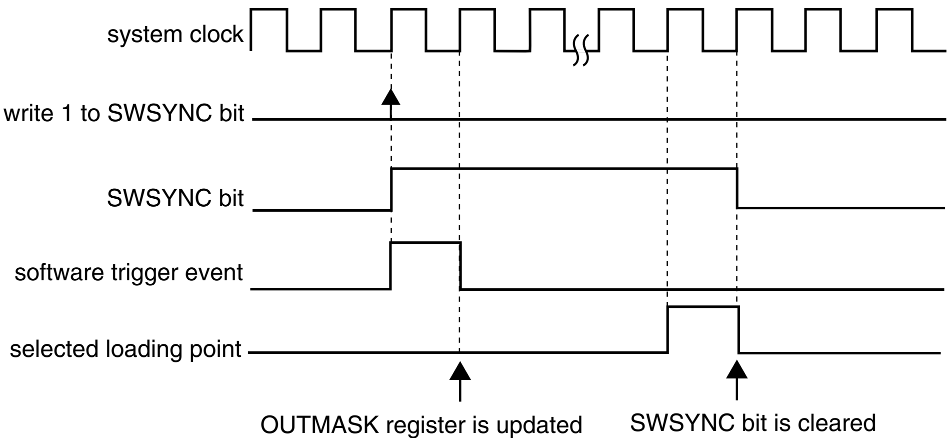


Figure 40-256. OUTMASK Synchronization with (SYNCMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0) and (Software Trigger Was Used)

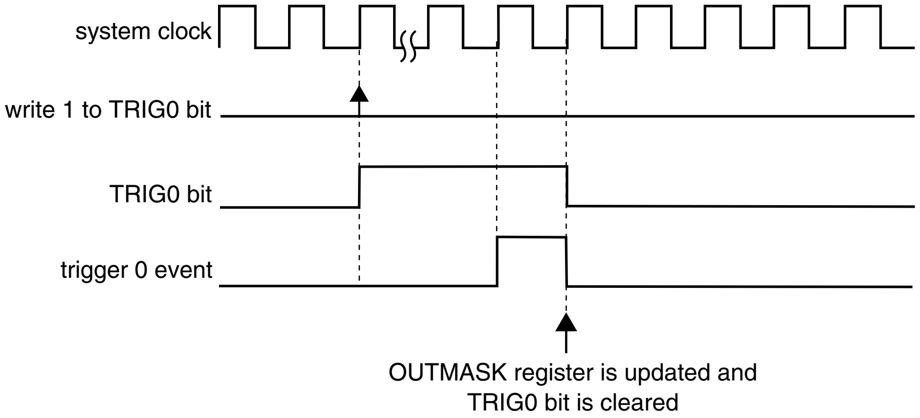


Figure 40-257. OUTMASK Synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0), and (a Hardware Trigger Was Used)

If (SYNCMODE = 0), (SYNCHOM = 1) and (PWMSYNC = 1) then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware Trigger. An example with a hardware trigger follows.

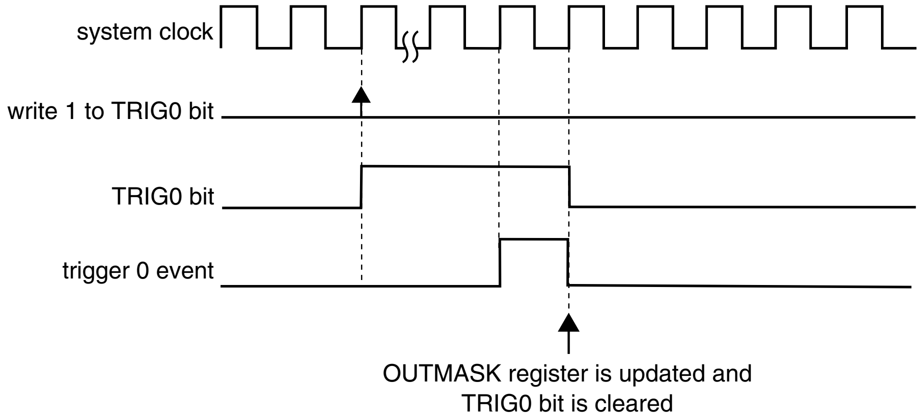


Figure 40-258. OUTMASK Synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 1), and (a Hardware Trigger Was Used)

40.4.11.8 INVCTRL Register Synchronization

The INVCTRL register synchronization updates the INVCTRL register with its buffer value.

The INVCTRL register can be updated at each rising edge of system clock (INVC = 0) or by the enhanced PWM synchronization (INVC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the INVCTRL register synchronization depends on SWINVC and HWINVC bits.

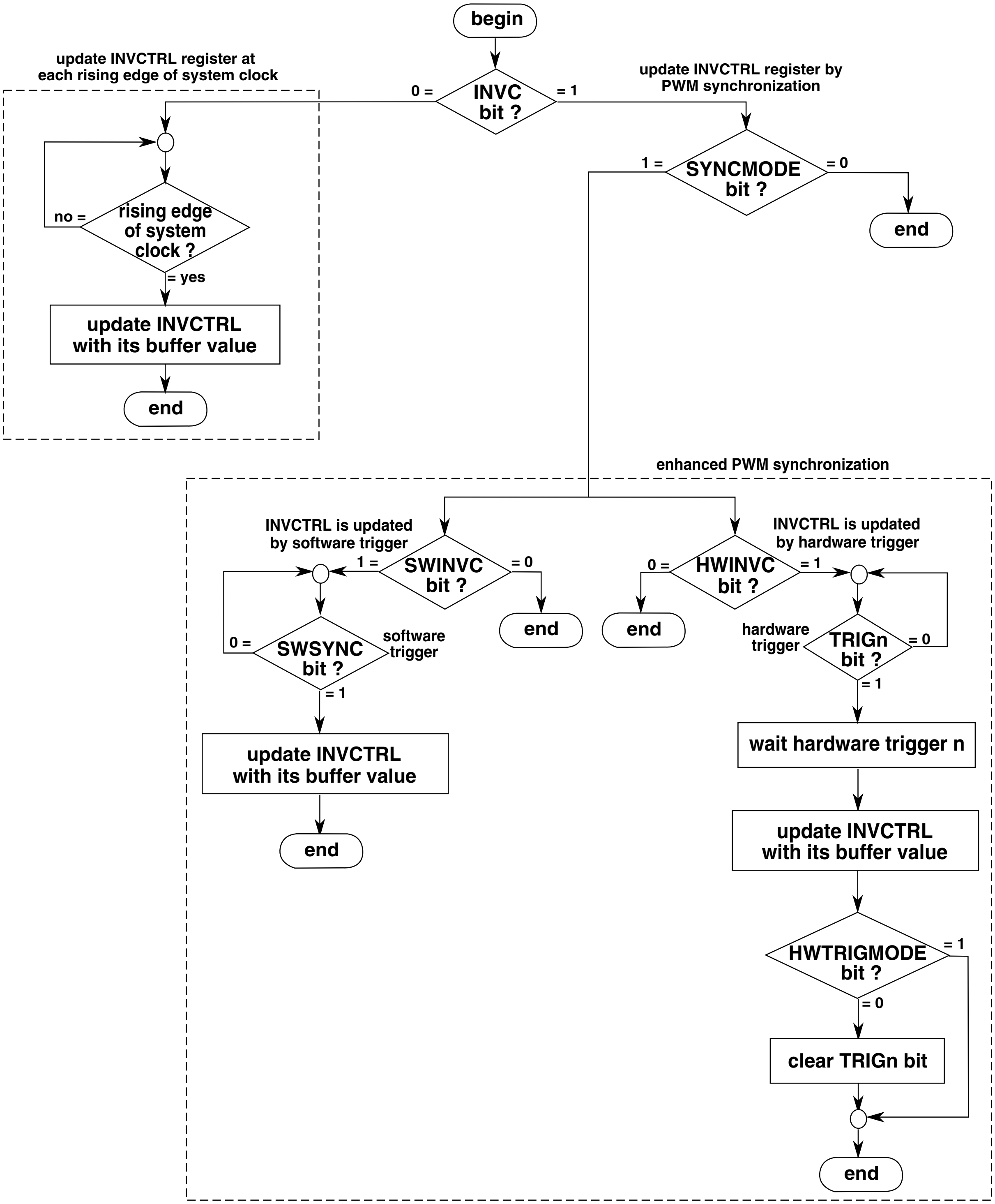

begin
update INVCTRL register at each rising edge of system clock
0 =
INVC bit ?
= 1
update INVCTRL register by PWM synchronization
1 =
SYNCMODE bit ?
= 0
end
no =
rising edge of system clock ?
= yes
update INVCTRL with its buffer value
end
enhanced PWM synchronization
INVCTRL is updated by software trigger
1 =
SWINVC bit ?
= 0
end
0 =
SWSYNC bit ?
software trigger
= 1
update INVCTRL with its buffer value
end
INVCTRL is updated by hardware trigger
0 =
HWINVC bit ?
= 1
end
hardware trigger
TRIGn bit ?
= 0
= 1
wait hardware trigger n
update INVCTRL with its buffer value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 40-259. INVCTRL Register Synchronization Flowchart

40.4.11.9 SWOCTRL Register Synchronization

The SWOCTRL register synchronization updates the SWOCTRL register with its buffer value.

The SWOCTRL register can be updated at each rising edge of system clock (SWOC = 0) or by the enhanced PWM synchronization (SWOC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the SWOCTRL register synchronization depends on SWSOC and HWSOC bits.

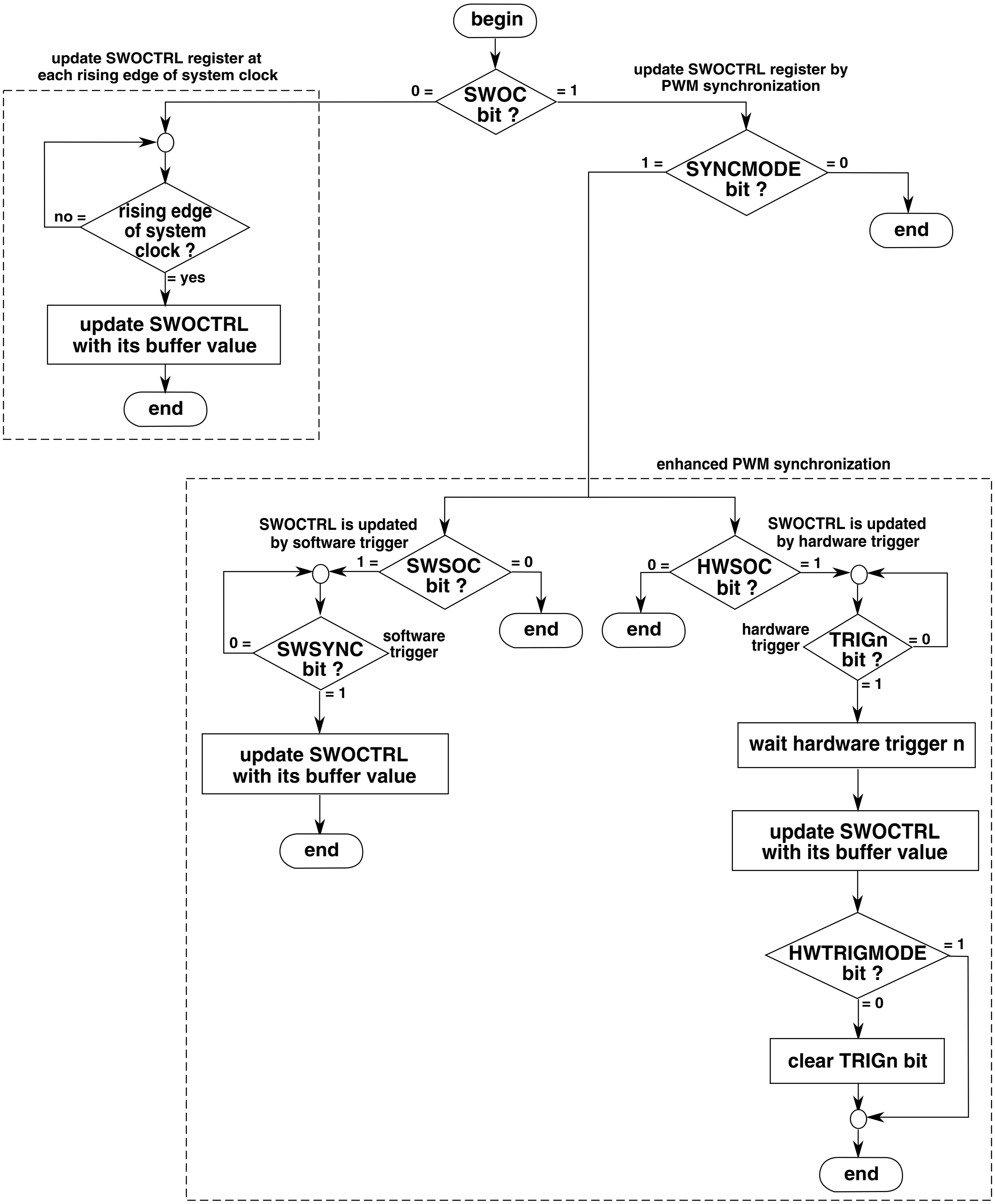

begin
update SWOCTRL register at each rising edge of system clock
0 =
SWOC bit ?
= 1
update SWOCTRL register by PWM synchronization
no =
rising edge of system clock ?
= yes
update SWOCTRL with its buffer value
end
1 =
SYNCMODE bit ?
= 0
end
enhanced PWM synchronization
SWOCTRL is updated by software trigger
1 =
SWSOC bit ?
= 0
end
0 =
SWSYNC bit ?
software trigger
= 1
update SWOCTRL with its buffer value
end
0 =
HWSOC bit ?
= 1
SWOCTRL is updated by hardware trigger
end
hardware trigger
TRIGn bit ?
= 0
= 1
wait hardware trigger n
update SWOCTRL with its buffer value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 40-260. SWOCTRL Register Synchronization Flowchart

40.4.11.10 FTM Counter Synchronization

The FTM counter synchronization is a mechanism that allows the FTM to re-start the PWM generation at a certain point in the PWM period. The channels outputs are forced to their initial value (except for channels in output compare mode) and the FTM counter is forced to its initial counting value defined by CNTIN register.

The following figure shows the FTM counter synchronization. Note that after the synchronization event had occurred the channel (n) is set to its initial value and the channel (n+1) is not set to its initial value due to a specific timing of this figure in which the deadtime insertion prevents this channel output from transitioning to 1. If no deadtime insertion is selected then the channel (n+1) transitions to logical value 1 immediately after the synchronization event had occurred.

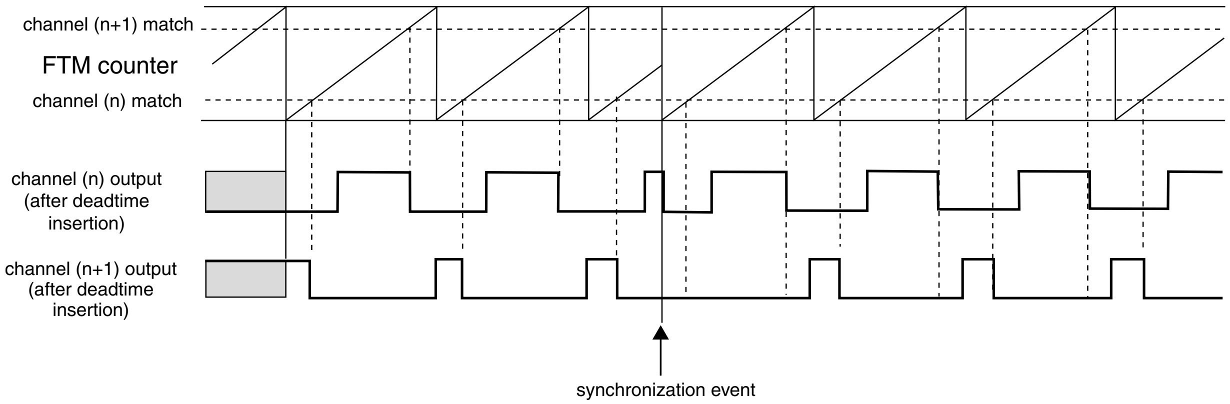

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)
synchronization event

Figure 40-261. FTM Counter Synchronization

The FTM counter synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, it is expected that the FTM counter be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the FTM counter synchronization depends on SWRSTCNT and HWRSTCNT bits according to the following flowchart.

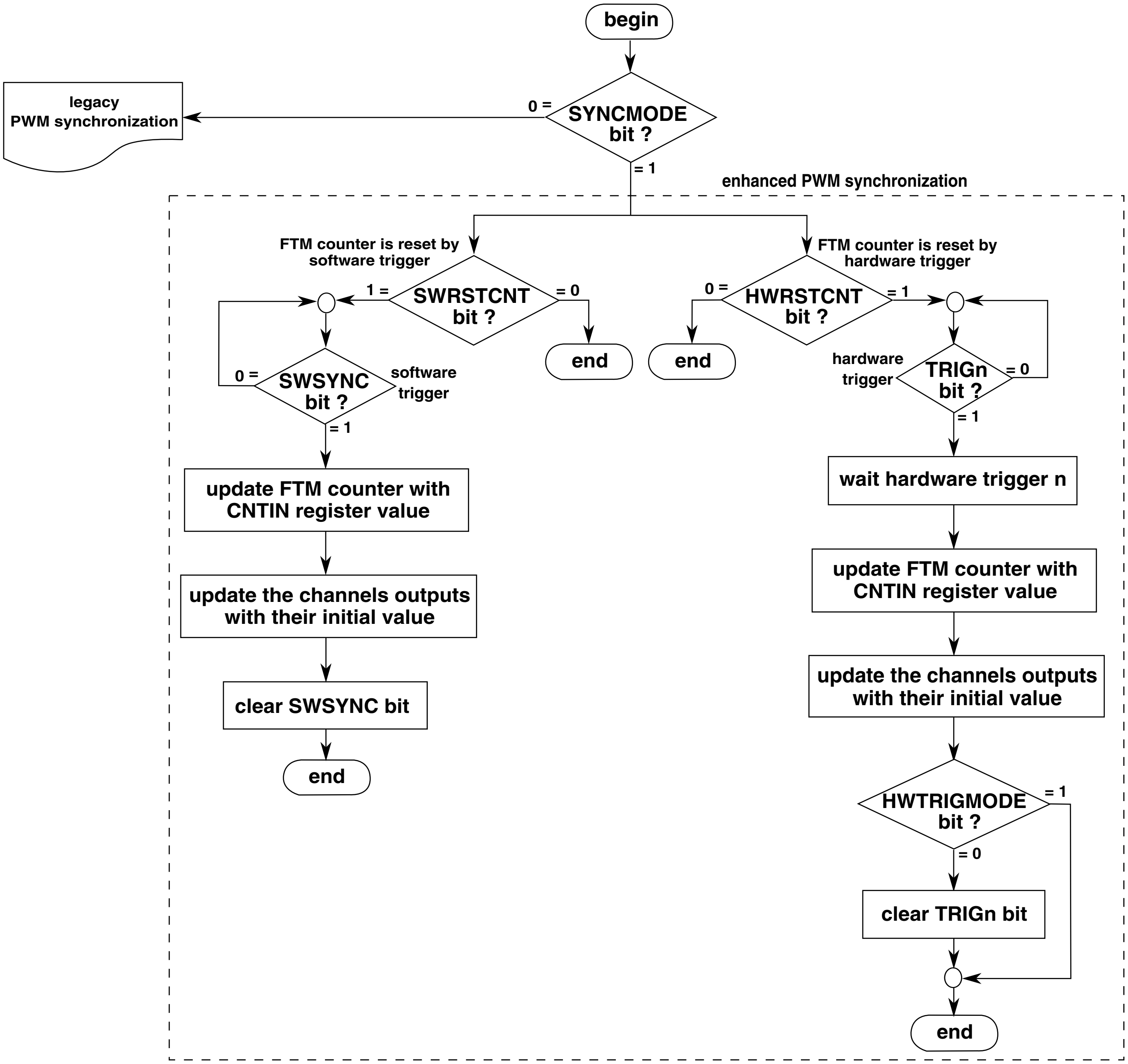

begin
SYNCMODE bit ?
0 =
legacy PWM synchronization
= 1
enhanced PWM synchronization
FTM counter is reset by software trigger
SWRSTCNT bit ?
1 =
= 0
end
SWSYNC bit ?
0 =
software trigger
= 1
update FTM counter with CNTIN register value
update the channels outputs with their initial value
clear SWSYNC bit
end
FTM counter is reset by hardware trigger
HWRSTCNT bit ?
0 =
= 1
end
TRIGn bit ?
hardware trigger
= 0
= 1
wait hardware trigger n
update FTM counter with CNTIN register value
update the channels outputs with their initial value
HWTRIGMODE bit ?
= 1
= 0
clear TRIGn bit
end

Figure 40-262. FTM Counter Synchronization Flowchart

In the case of legacy PWM synchronization, the FTM counter synchronization depends on REINIT and PWMSYNC bits according to the following description.

If (SYNCMODE = 0), (REINIT = 1) and (PWMSYNC = 0) then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger then the TRIGn bit is cleared according to Hardware Trigger. Examples with software and hardware triggers follow.

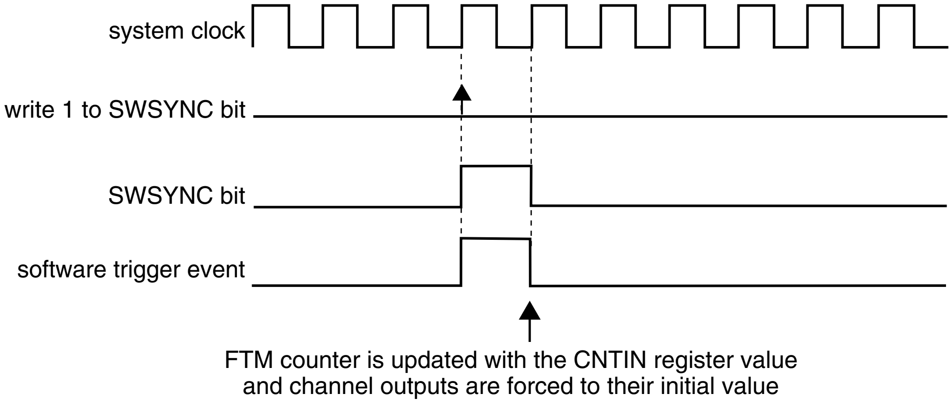

system clock
write 1 to SWSYNC bit
SWSYNC bit
software trigger event
FTM counter is updated with the CNTIN register value
and channel outputs are forced to their initial value

Figure 40-263. FTM Counter Synchronization with (SYNCMODE = 0), (REINIT = 1), (PWMSYNC = 0), and (Software Trigger Was Used)

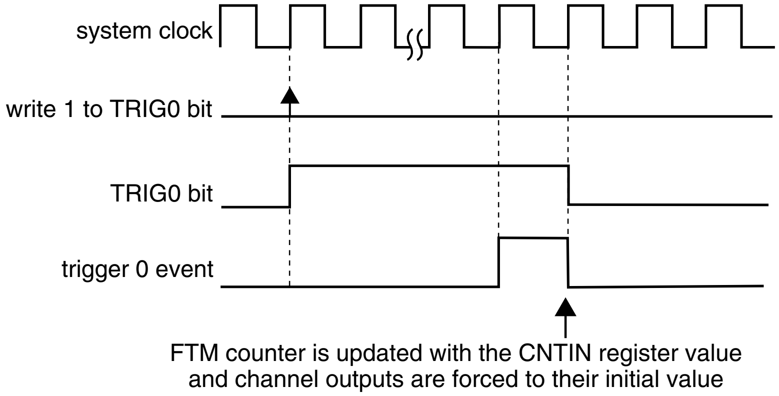

system clock
write 1 to TRIG0 bit
TRIG0 bit
trigger 0 event
FTM counter is updated with the CNTIN register value
and channel outputs are forced to their initial value

Figure 40-264. FTM Counter Synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 0), and (a Hardware Trigger Was Used)

If (SYNCMODE = 0), (REINIT = 1) and (PWMSYNC = 1) then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware Trigger.

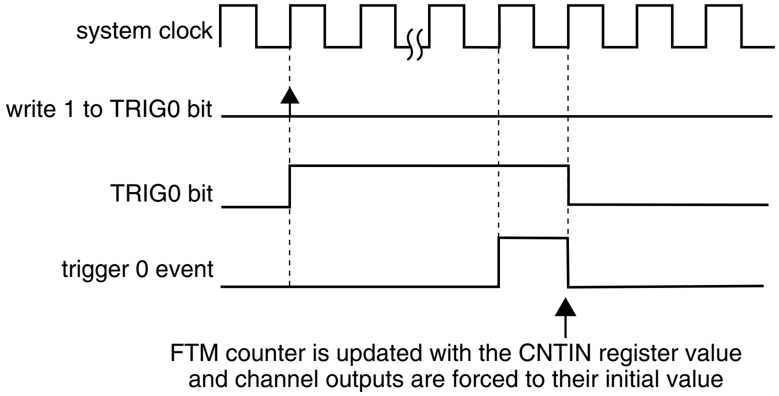
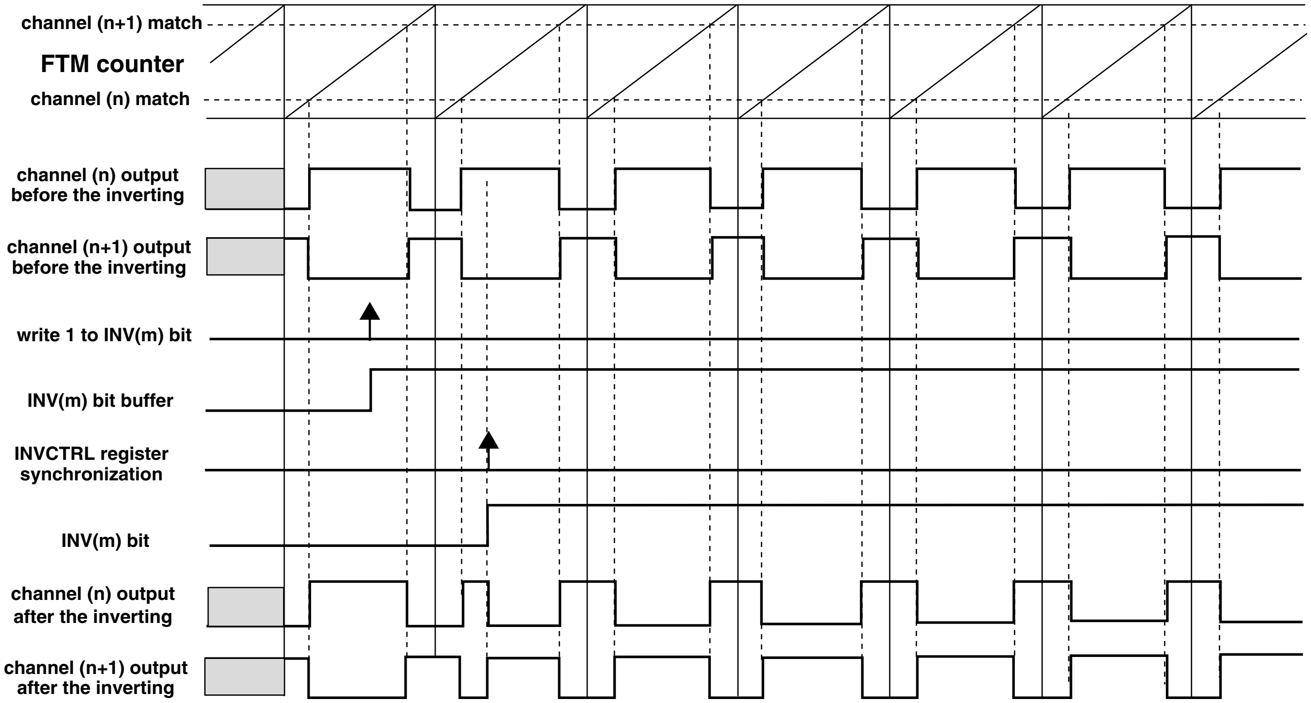

system clock
write 1 to TRIG0 bit
TRIG0 bit
trigger 0 event
FTM counter is updated with the CNTIN register value
and channel outputs are forced to their initial value

Figure 40-265. FTM Counter Synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 1), and (a Hardware Trigger Was Used)

40.4.12 Inverting

The invert functionality swaps the signals between channel (n) and channel (n+1) outputs. The inverting operation is selected when (FTMEN = 1), (QUADEN = 0), (DECAPEN = 0), (COMBINE = 1), (COMP = 1), (CPWMS = 0), and (INVm = 1), where m represents a channel pair. The INVm bit in INVCTRL register is updated with its buffer value according to INVCTRL Register Synchronization

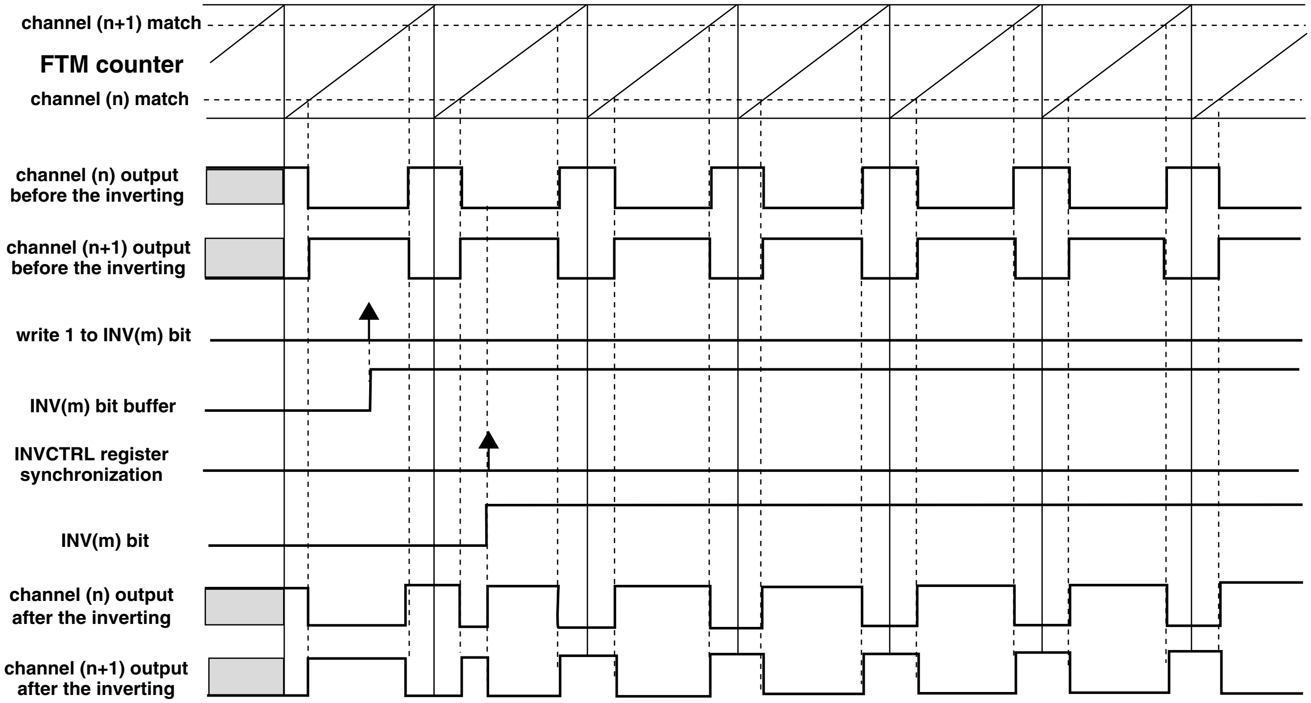
In high-true (ELSnB:ELSnA = 1:0) combine mode, the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN), forced high at the channel (n) match and forced low at the channel (n+1) match. If the inverting is selected, the channel (n) output behavior is changed to force high at the beginning of the PWM period, force low at the channel (n) match and force high at the channel (n+1) match. See the following figure.


channel (n+1) match
FTM counter
channel (n) match
channel (n) output before the inverting
channel (n+1) output before the inverting
write 1 to INV(m) bit
INV(m) bit buffer
INVCTRL register synchronization
INV(m) bit
channel (n) output after the inverting
channel (n+1) output after the inverting

NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 40-266. Channels (n) and (n+1) Outputs After the Inverting in High-True (ELSnB:ELSnA = 1:0) Combine Mode

Note that the ELSnB:ELSnA bits value should be consider since that they define the active state of the channels outputs. In low-true (ELSnB:ELSnA = X:1) combine mode, the channel (n) output is forced high at the beginning of the period, forced low at the channel (n) match and forced high at the channel (n+1) match. In the case the inverting is selected the channels (n) and (n+1) present waveforms as shown in the following figure.


channel (n+1) match
FTM counter
channel (n) match
channel (n) output before the inverting
channel (n+1) output before the inverting
write 1 to INV(m) bit
INV(m) bit buffer
INVCTRL register synchronization
INV(m) bit
channel (n) output after the inverting
channel (n+1) output after the inverting

NOTE
INV(m) bit selects the inverting to the pair channels (n) and (n+1).

Figure 40-267. Channels (n) and (n+1) Outputs After the Inverting in Low-True (ELSnB:ELSnA = X:1) Combine Mode

Note

It is expected that the inverting feature be used only in combine mode.

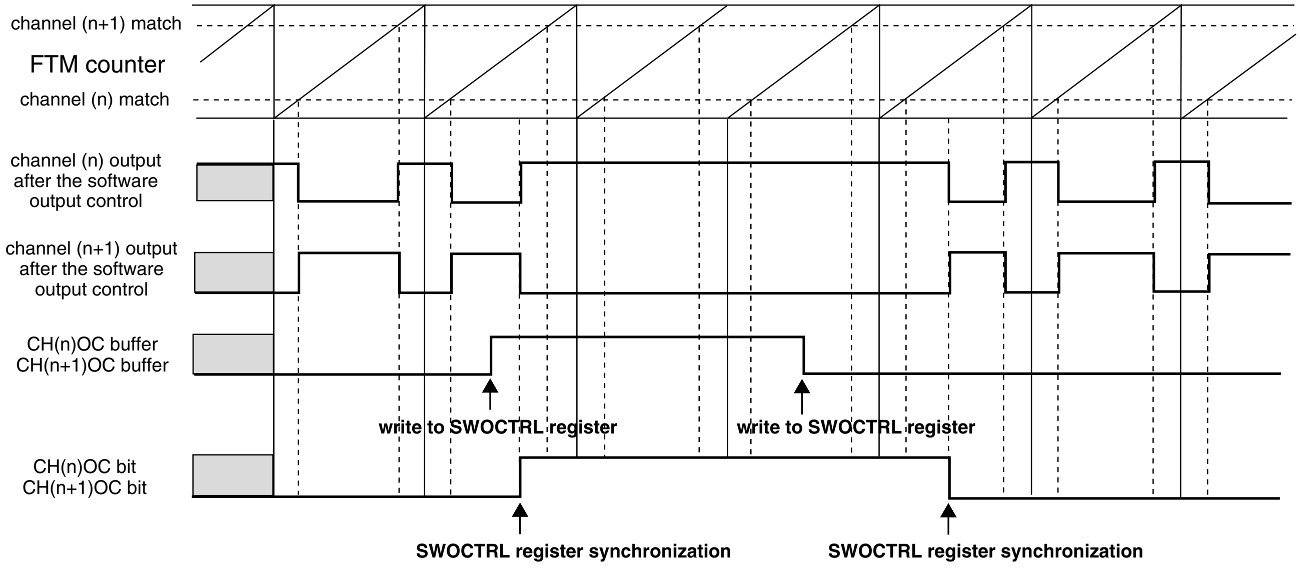
40.4.13 Software Output Control

The software output control forces the channel output according to software defined values at a specific time in the PWM generation.

The software output control is selected when (FTMEN = 1), (QUADEN = 0), (DECAPEN = 0), (COMBINE = 1), (CPWMS = 0), and (CHnOC = 1). The CHnOC bit enables the software output control for a specific channel output and the CHnOCV selects the value that is forced to this channel output.

Both CHnOC and CHnOCV bits in SWOCTRL register are buffered and updated with their buffer value according to SWOCTRL Register Synchronization.

The following figure shows the channels (n) and (n+1) outputs signals when the software output control is used. In this case the channels (n) and (n+1) are set to combine and complementary mode.


channel (n+1) match
FTM counter
channel (n) match
channel (n) output after the software output control
channel (n+1) output after the software output control
CH(n)OC buffer
CH(n+1)OC buffer
write to SWOCTRL register
write to SWOCTRL register
CH(n)OC bit
CH(n+1)OC bit
SWOCTRL register synchronization
SWOCTRL register synchronization

NOTE
CH(n)OCV = 1 and CH(n+1)OCV = 0.

Figure 40-268. Example of Software Output Control in Combine and Complementary Mode

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is zero.

Table 40-302. Software Ouput Control Behavior when (COMP = 0)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to one

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is one.

Table 40-303. Software Ouput Control Behavior when (COMP = 1)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to zero

Note

- It is expected that the software output control feature be used only in combine mode.
- The CH(n)OC and CH(n+1)OC bits should be equal.
- The COMP bit should not be modified when software output control is enabled, that is, CH(n)OC = 1 and/or CH(n+1)OC = 1.
- Software output control has the same behavior with disabled or enabled FTM counter (see the CLKS bitfield description in the Status and Control register).

40.4.14 Deadtime Insertion

The deadtime insertion is enabled when (DTEN = 1) and (DTVAL[5:0] is non- zero).

DEADTIME register defines the deadtime delay that can be used for all FTM channels. The DTPS[1:0] bits define the prescaler for the system clock and the DTVAL[5:0] bits define the deadtime modulo (number of the deadtime prescaler clocks).

The deadtime delay insertion ensures that no two complementary signals (channels (n) and (n+1)) drive the active state at the same time.

If POL(n) = 0, POL(n+1) = 0, and the deadtime is enabled, then when the channel (n) match (FTM 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 (FTM 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 POL(n) = 1, POL(n+1) = 1, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the high value until the end of the deadtime delay when the channel (n) output is cleared. Similarly, when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the high value until the end of the deadtime delay when the channel (n +1) output is cleared.

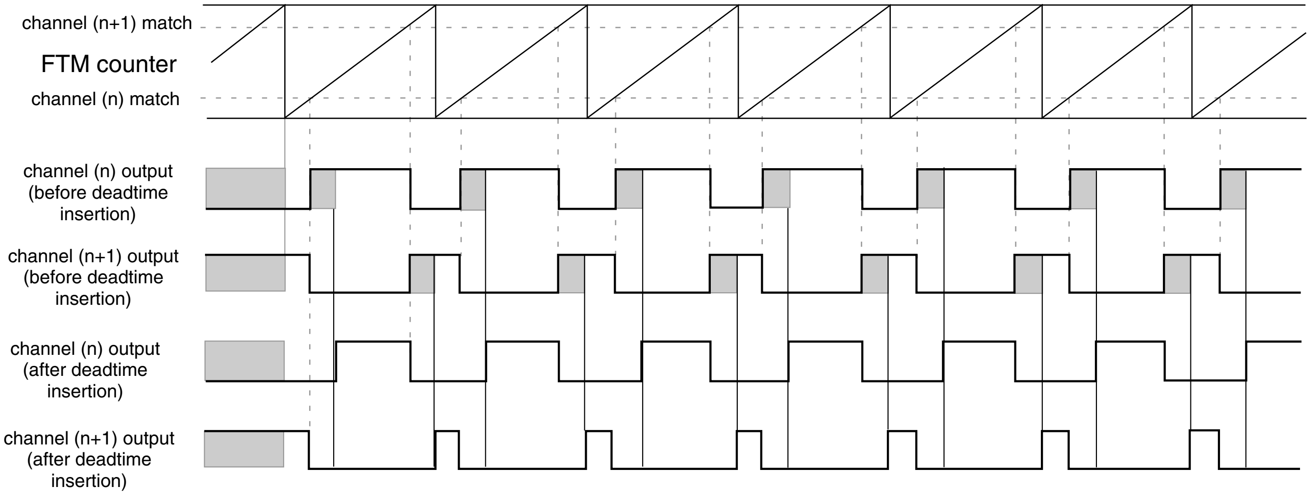

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 40-269. Deadtime Insertion with ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0

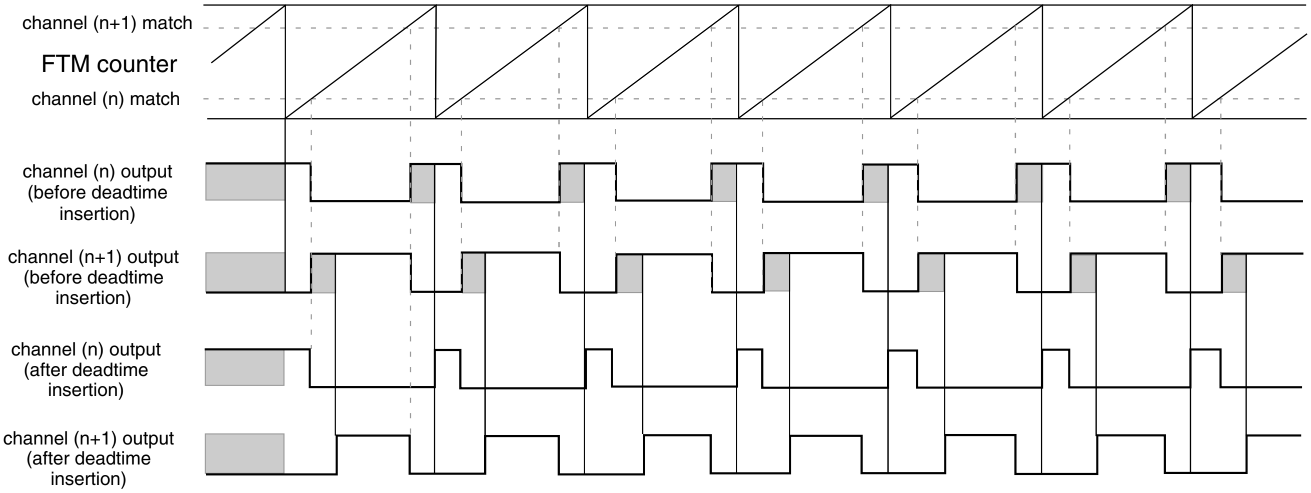

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 40-270. Deadtime Insertion with ELSnB:ELSnA = X:1, POL(n) = 0, and POL(n+1) = 0

NOTE

It is expected that the deadtime feature be used only in combine and complementary modes.

40.4.14.1 Deadtime Insertion Corner Cases

If (PS[2:0] is cleared), (DTPS[1:0] = 0:0 or DTPS[1:0] = 0:1):

- and the deadtime delay is greater than or equal to the channel (n) duty cycle ((C(n+1)V – C(n)V) × system clock), then the channel (n) output is always the inactive value (POL(n) bit value).
- and the deadtime delay is greater than or equal to the channel (n+1) duty cycle ((MOD – CNTIN + 1 – (C(n+1)V – C(n)V)) × system clock), then the channel (n+1) output is always the inactive value (POL(n+1) bit value).

Although, in most cases the deadtime delay is not comparable to channels (n) and (n+1) duty cycle, the following figures show examples where the deadtime delay is comparable to the duty cycle.

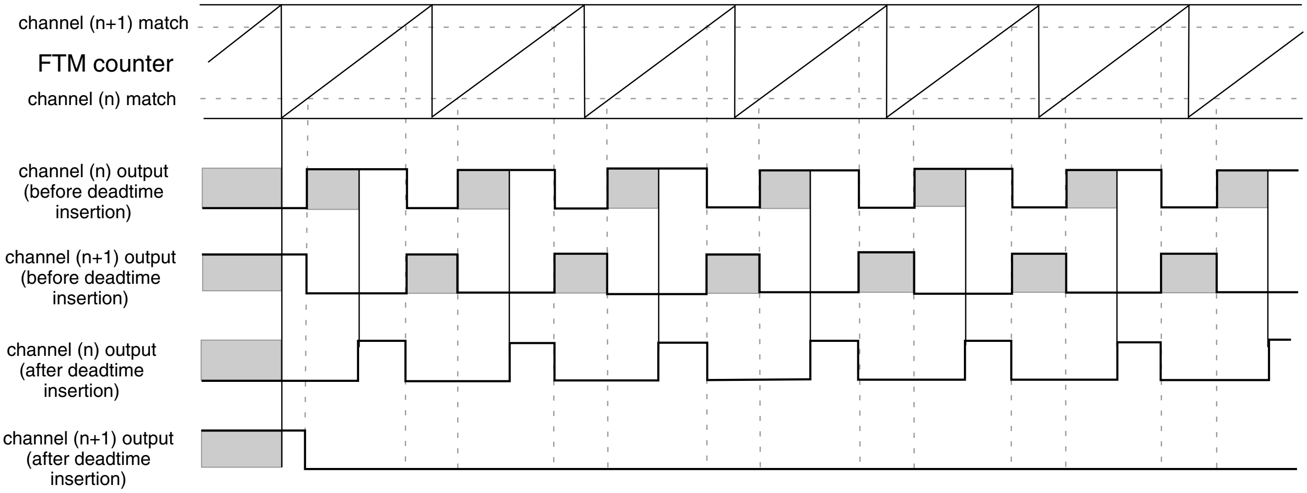

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 40-271. Example of the Deadtime Insertion (ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the Deadtime Delay Is Comparable To Channel (n+1) Duty Cycle

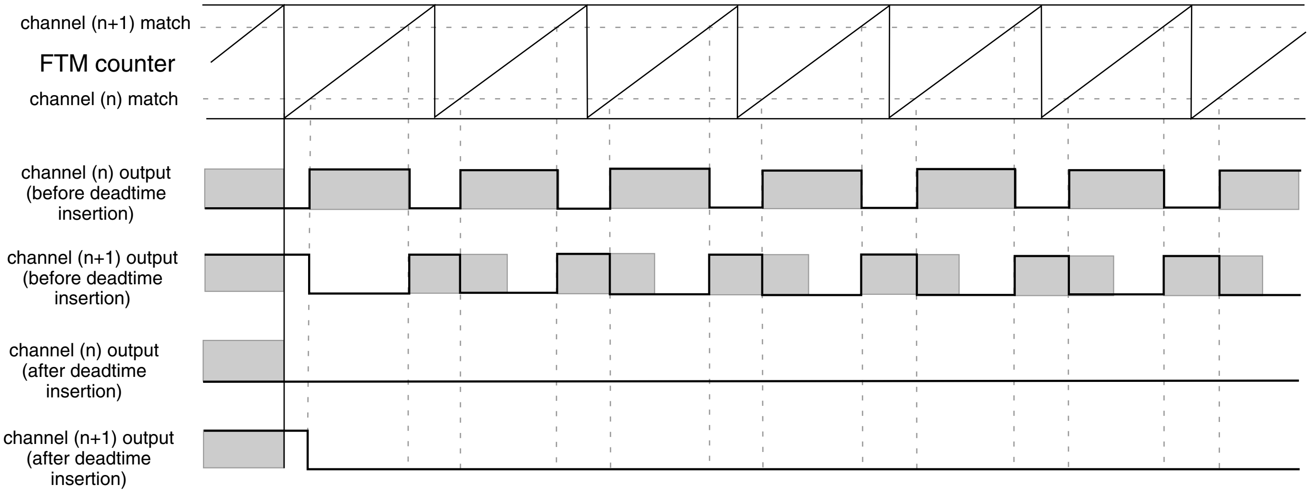

channel (n+1) match
FTM counter
channel (n) match
channel (n) output (before deadtime insertion)
channel (n+1) output (before deadtime insertion)
channel (n) output (after deadtime insertion)
channel (n+1) output (after deadtime insertion)

Figure 40-272. Example of the Deadtime Insertion (ELSnB:ELSnA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the Deadtime Delay Is Comparable To Channels (n) and (n+1) Duty Cycle

40.4.15 Output Mask

The output mask can be used to force channels output to their inactive state through software (for example: to control a BLDC motor).

Any write to the OUTMASK register updates its write buffer. The OUTMASK register is updated with its buffer value by PWM synchronization (OUTMASK Register Synchronization).

If CHnOM = 1, then the channel (n) output is forced to its inactive state (POLn bit value). If CHnOM = 0, then the channel (n) output is unaffected by the output mask (see the following figure).

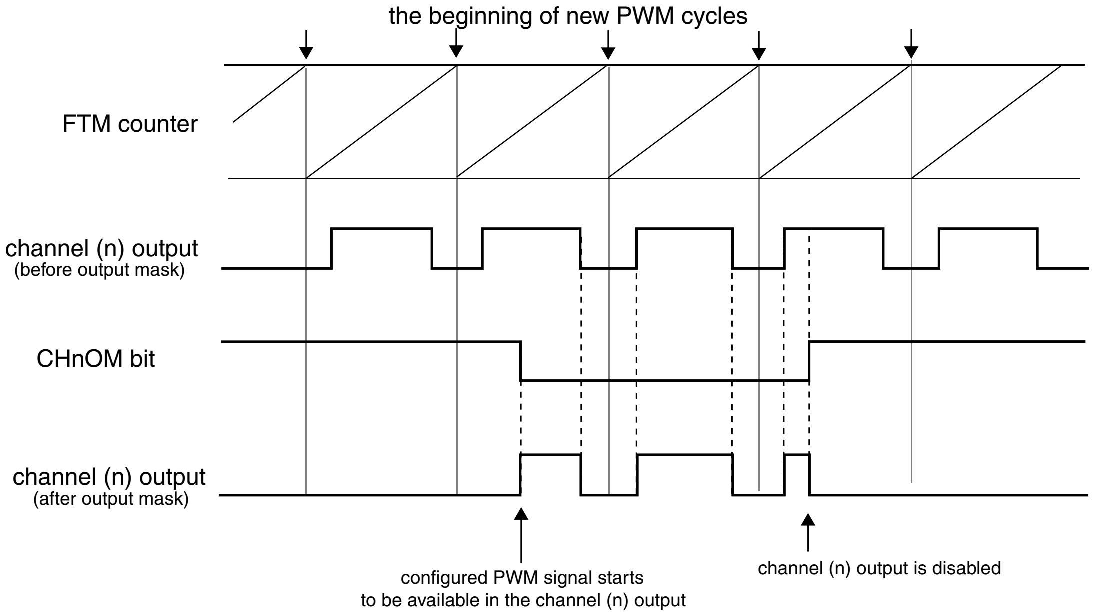

the beginning of new PWM cycles
FTM counter
channel (n) output
(before output mask)
CHnOM bit
channel (n) output
(after output mask)
configured PWM signal starts
to be available in the channel (n) output
channel (n) output is disabled

Figure 40-273. Output Mask with POLn = 0

The following table shows the output mask result before the polarity control.

Table 40-304. Output Mask Result for Channel (n) (Before the Polarity Control)

CHnOM	Output Mask Input	Output Mask Result
0	inactive state	inactive state
	active state	active state
1	inactive state	inactive state
	active state	

Note

It is expected the output mask feature be used only in combine mode.

40.4.16 Fault Control

The fault control is enabled if (FTMEN = 1) and (FAULTM[1:0] ≠ 0:0).

FTM can have up to four fault inputs. FAULTnEN bit (where n = 0, 1, 2, 3) enables the fault input n and FFLTRnEN bit enables the fault input n filter. FFVAL[3:0] bits select the value of the enabled filter in each enabled fault input.

First each fault input signal is synchronized by the system clock (see the synchronizer block in the following figure). Following synchronization, the fault input n signal enters the filter block. When there is a state change in the fault input n signal, the 5-bit counter is reset and starts counting up. As long as the new state is stable on the fault input n, the counter continues to increment. If the 5-bit counter overflows (the counter exceeds the value of the FFVAL[3:0] bits), the new fault input n value is validated. It is then transmitted as a pulse edge to the edge detector.

If the opposite edge appears on the fault input n signal before validation (counter overflow), 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 FFVAL[3:0] bits (× system clock) is regarded as a glitch and is not passed on to the edge detector.

The fault input n filter is disabled when the FFVAL[3:0] bits are zero or when FAULTnEN = 0. In this case the fault input n signal is delayed 2 rising edges of the system clock and the FAULTFn bit is set on 3th rising edge of the system clock after a rising edge occurs on the fault input n.

If FFVAL[3:0] ≠ 0000 and FAULTnEN = 1, then the fault input n signal is delayed (3 + FFVAL[3:0]) rising edges of the system clock, that is, the FAULTFn bit is set (4 + FFVAL[3:0]) rising edges of the system clock after a rising edge occurs on the fault input n.

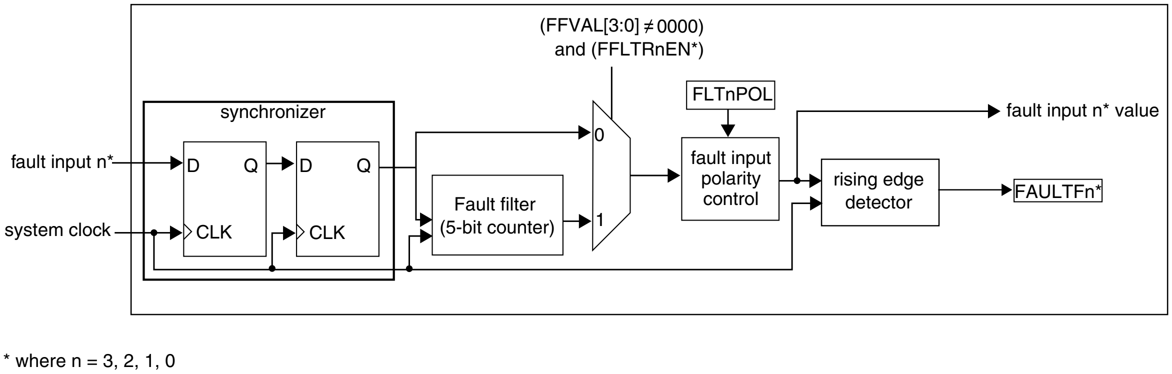


Figure 40-274. Fault Input n Control Block Diagram

If the fault control and fault input n are enabled and a rising edge at the fault input n signal is detected, then the FAULTFn bit is set. The FAULTF bit is the logic OR of FAULTFn[3:0] bits (see the following figure).

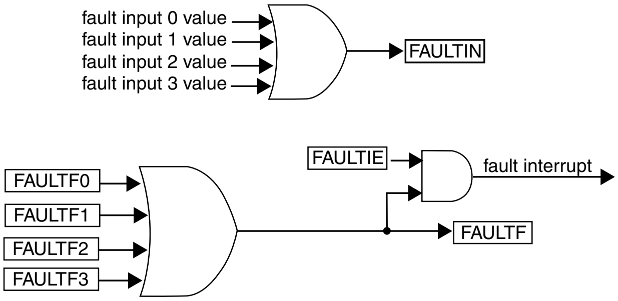


Figure 40-275. FAULTF and FAULTIN Bits and Fault Interrupt

If the fault control is enabled (FAULTM[1:0] ≠ 0:0), a fault condition has occurred (rising edge at the logic OR of the enabled fault inputs) and (FAULTEN = 1), then channels (n) and (n+1) output are forced to their safe value (the channel (n) output is forced to the value of POL(n) and the channel (n+1) is forced to the value of POL(n+1)).

The fault interrupt is generated when (FAULTF = 1) and (FAULTIE = 1). This interrupt request remains set until:

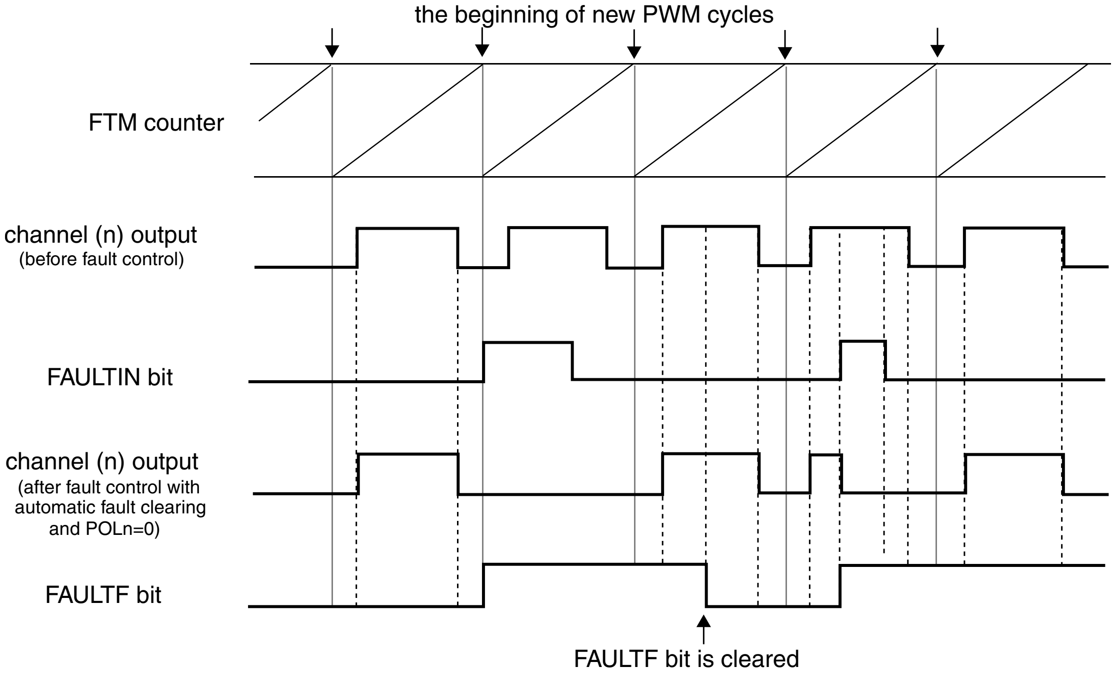
- Software clears the FAULTF bit (by reading FAULTF bit as 1 and writing 0 to it)
- Software clears the FAULTIE bit
- A reset occurs

Note

It is expected that the fault control be used only in combine mode.

40.4.16.1 Automatic Fault Clearing

If the automatic fault clearing is selected (FAULTM[1:0] = 1:1), then the channels output disabled by fault control is again enabled when the fault input signal (FAULTIN) returns to zero and a new PWM cycle begins (see the following figure).

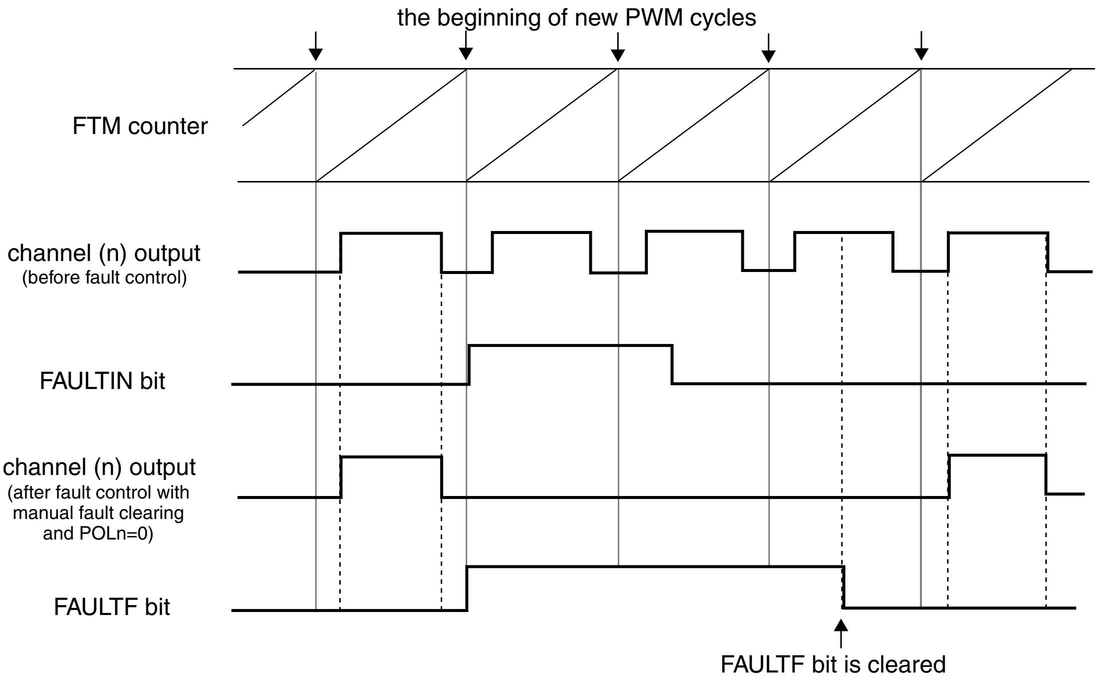


NOTE
The channel (n) output is after the fault control with automatic fault clearing and POLn = 0.

Figure 40-276. Fault Control with Automatic Fault Clearing

40.4.16.2 Manual Fault Clearing

If the manual fault clearing is selected (FAULTM[1:0] = 0:1 or 1:0), then the channels output disabled by fault control is again enabled when the FAULTF bit is cleared and a new PWM cycle begins (see the following figure).



NOTE
The channel (n) output is after the fault control with manual fault clearing and POLn = 0.

Figure 40-277. Fault Control with Manual Fault Clearing

40.4.16.3 Fault Inputs Polarity Control

The FLTjPOL bit selects the fault input j polarity (where j = 0, 1, 2, 3).

- If FLTjPOL = 0, the fault j input polarity is high, so the logical one at the fault input j indicates a fault.
- If FLTjPOL = 1, the fault j input polarity is low, so the logical zero at the fault input j indicates a fault.

40.4.17 Polarity Control

The POLn bit selects the channel (n) output polarity.

- If POLn = 0, the channel (n) output polarity is high, so the logical one is the active state and the logical zero is the inactive state.
- If POLn = 1, the channel (n) output polarity is low, so the logical zero is the active state and the logical one is the inactive state.

Note

It is expected that the polarity control be used only in combine mode.

40.4.18 Initialization

The initialization forces the CHnOI bit value to the channel (n) output when a one is written to the INIT bit.

The initialization depends on COMP and DTEN bits. The following table shows the values that channels (n) and (n+1) are forced by initialization when the COMP and DTEN bits are zero.

Table 40-305. Initialization Behavior when (COMP = 0 and DTEN = 0)

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	0	is forced to zero	is forced to zero
0	1	is forced to zero	is forced to one
1	0	is forced to one	is forced to zero
1	1	is forced to one	is forced to one

The following table shows the values that channels (n) and (n+1) are forced by initialization when (COMP = 1) or (DTEN = 1).

Table 40-306. Initialization Behavior when (COMP = 1 or DTEN = 1)

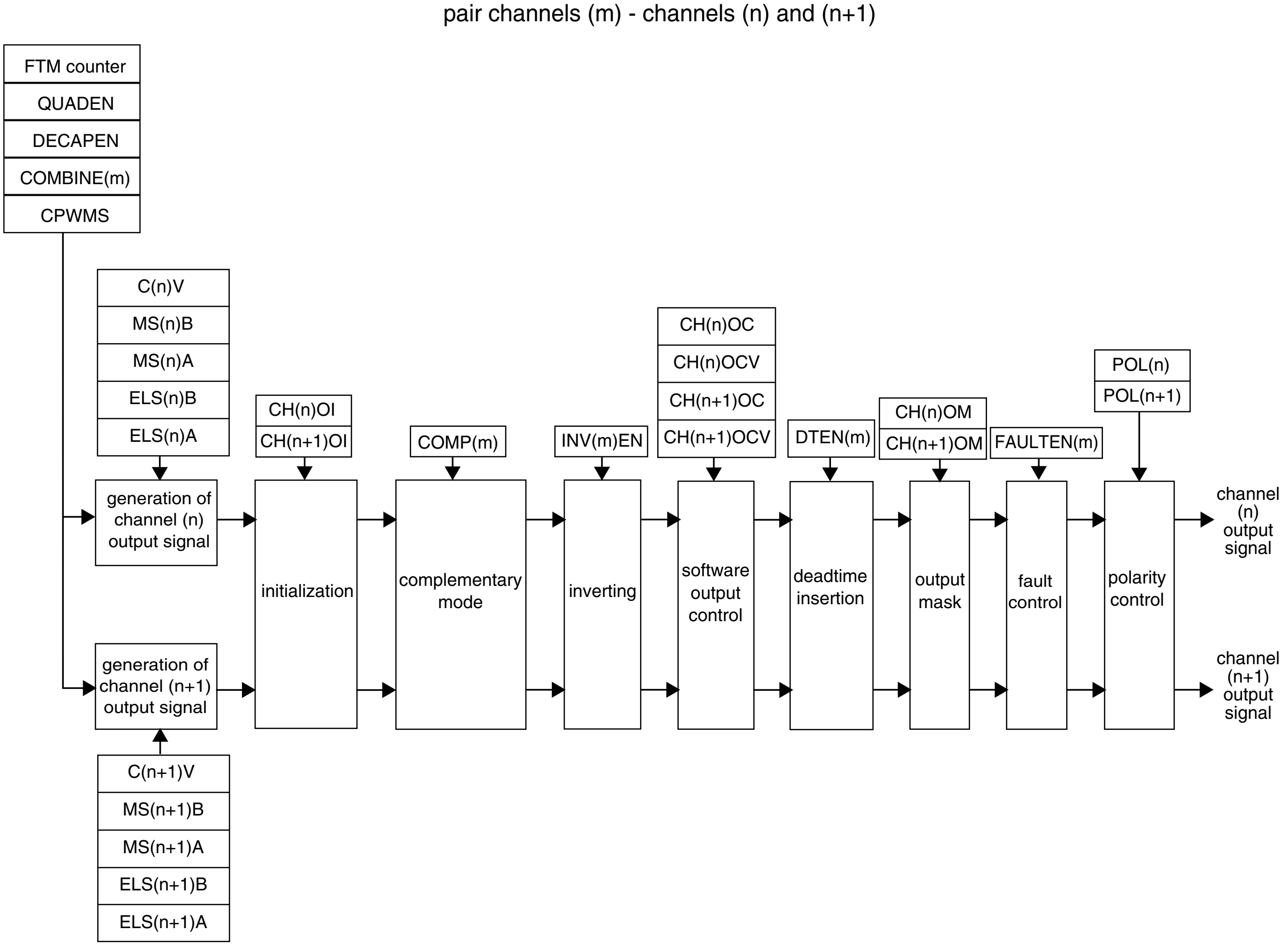
CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	X	is forced to zero	is forced to one
1	X	is forced to one	is forced to zero

Note

It is expected that the initialization feature be used only in combine mode and with disabled FTM counter (see the description of the CLKS field in the Status and Control register).

40.4.19 Features Priority

The following figure shows the priority of the features used at the generation of channels (n) and (n+1) outputs signals.



NOTE

The channels (n) and (n+1) are in output compare, EPWM, CPWM or combine modes.

Figure 40-278. Priority of the Features Used at the Generation of Channels (n) and (n+1) Outputs Signals

Note

It is expected that the initialization feature (Initialization) is not used with inverting (Inverting) and software output control (Software Output Control) features.

40.4.20 Channel Trigger Output

If CHjTRIG = 1 (where j = 0, 1, 2, 3, 4, or 5), then the FTM generates a trigger when the channel (j) match occurs (FTM counter = C(j)V).

The channel trigger output provides a trigger signal that is used for on-chip modules.

The FTM is able to generate multiple triggers in one PWM period. Since each trigger is generated for a specific channel, several channels are required to implement this functionality. This behavior is described in the following figure.

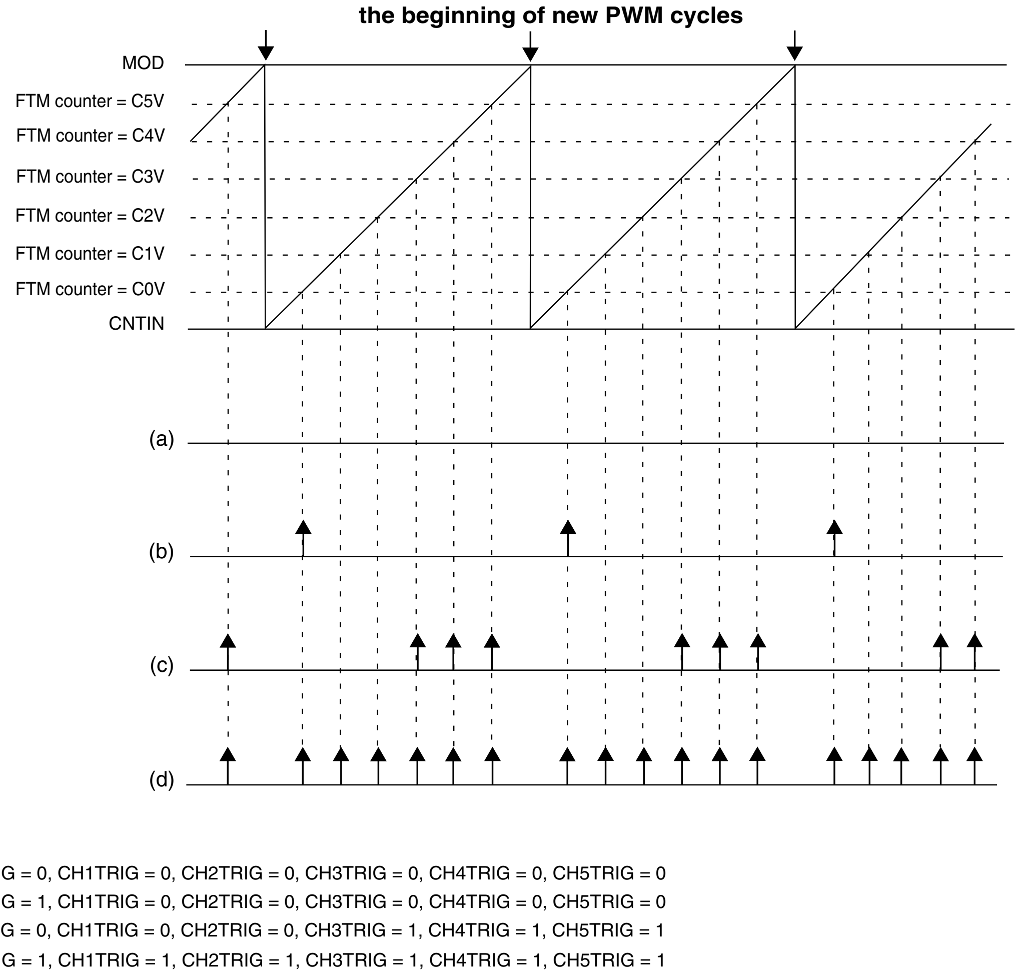

the beginning of new PWM cycles
MOD
FTM counter = C5V
FTM counter = C4V
FTM counter = C3V
FTM counter = C2V
FTM counter = C1V
FTM counter = C0V
CNTIN
(a)
(b)
(c)
(d)
NOTE
(a) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0
(b) CH0TRIG = 1, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0
(c) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1
(d) CH0TRIG = 1, CH1TRIG = 1, CH2TRIG = 1, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1

Figure 40-279. Channel Match Trigger

Note

It is expected that the channel match trigger be used only in combine mode.

40.4.21 Initialization Trigger

If INITTRIGEN = 1, then the FTM generates a trigger when the FTM counter is updated with the CNTIN register value in the following cases.

- The FTM counter is automatically updated with the CNTIN register value by the selected counting mode.

- When there is a write to CNT register
- When there is the FTM counter synchronization (FTM Counter Synchronization)
- If (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits

The following figures show the cases.

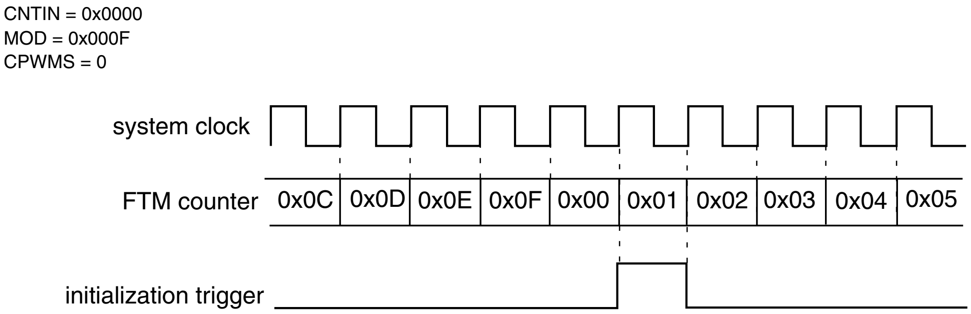


Figure 40-280. Initialization Trigger Is Generated When the FTM Counting Achieves the CNTIN Register Value

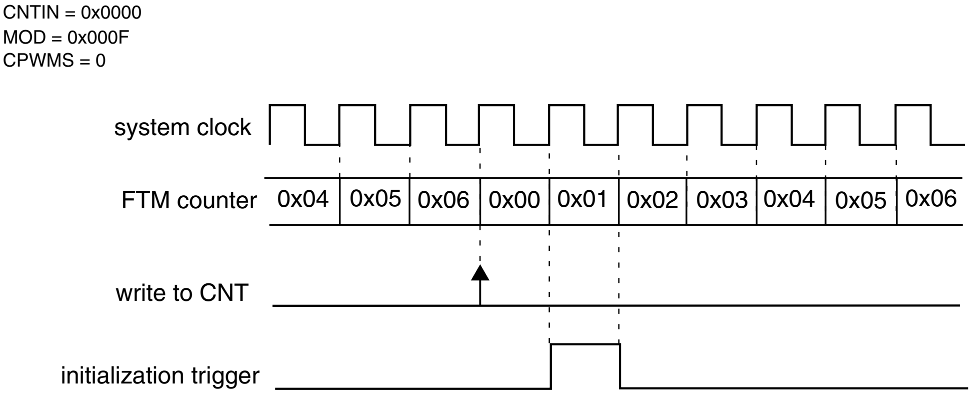


Figure 40-281. Initialization Trigger Is Generated When There Is a Write to CNT Register

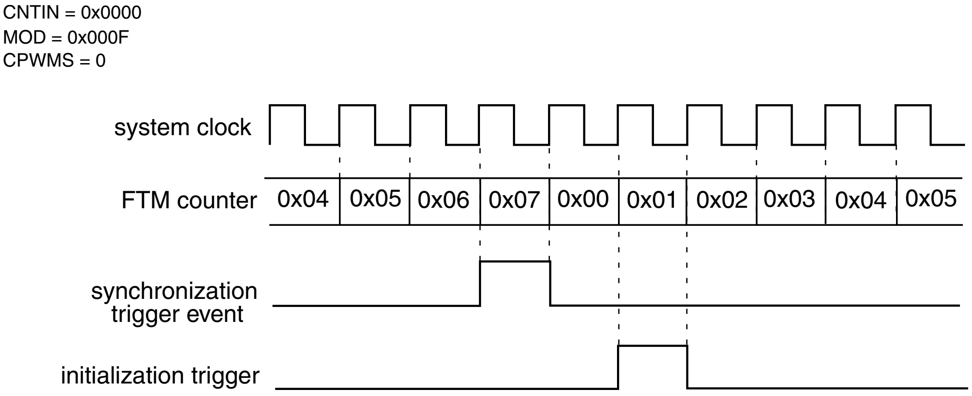


Figure 40-282. Initialization Trigger Is Generated When There Is the FTM Counter Synchronization

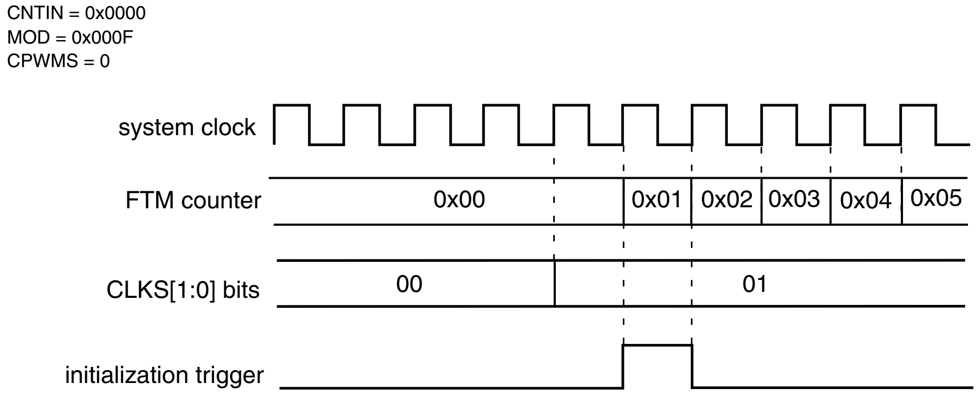


Figure 40-283. Initialization Trigger Is Generated If (CNT = CNTIN), (CLKS[1:0] = 0:0), and a Value Different From Zero Is Written to CLKS[1:0] Bits

The initialization trigger output provides a trigger signal that is used for on-chip modules.

Note

It is expected that the initialization trigger be used only in combine mode.

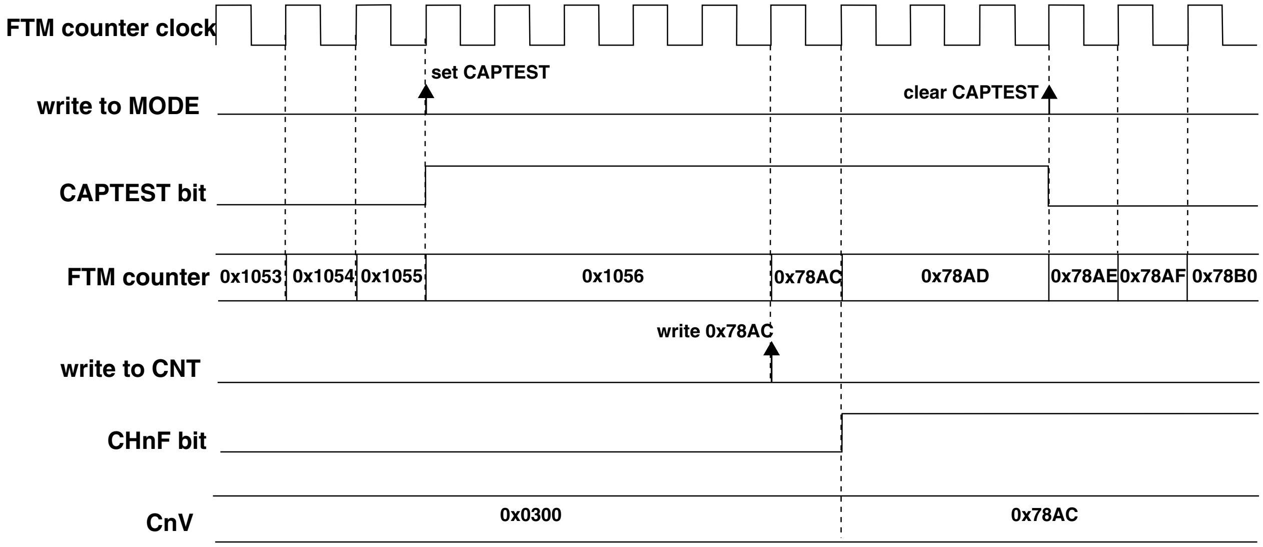
40.4.22 Capture Test Mode

The capture test mode allows to test the CnV registers, the FTM counter and the interconnection logic between the FTM counter and CnV registers.

In this test mode, all channels must be configured for input capture mode (Input Capture Mode) and FTM counter must be configured to the up counting (Up Counting).

When the capture test mode is enabled (CAPTEST = 1), the FTM counter is frozen and any write to CNT register updates directly the FTM counter (see the following figure). After it was written, all CnV registers are updated with the written value to CNT register and CHnF bits are set. Therefore, the FTM counter is updated with its next value according to its configuration (its next value depends on CNTIN, MOD, and the written value to FTM counter).

The next reads of CnV registers return the written value to the FTM counter and the next reads of CNT register return FTM counter next value.



NOTE
- FTM counter configuration: (FTMEN = 1), (QUADEN = 0), (CAPTEST = 1), (CPWMS = 0), (CNTIN = 0x0000), and (MOD = 0xFFFF)
- FTM channel n configuration: input capture mode - (DECAPEN = 0), (COMBINE = 0), and (MSnB:MSnA = 0:0)

Figure 40-284. Capture Test Mode

40.4.23 DMA

The channel generates a DMA transfer request according to DMA and CHnIE bits (see the following table).

Table 40-307. Channel DMA Transfer Request

DMA	CHnIE	Channel DMA Transfer Request	Channel Interrupt
0	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
0	1	The channel DMA transfer request is not generated.	The channel interrupt is generated if (CHnF = 1).
1	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
1	1	The channel DMA transfer request is generated if (CHnF = 1).	The channel interrupt is not generated.

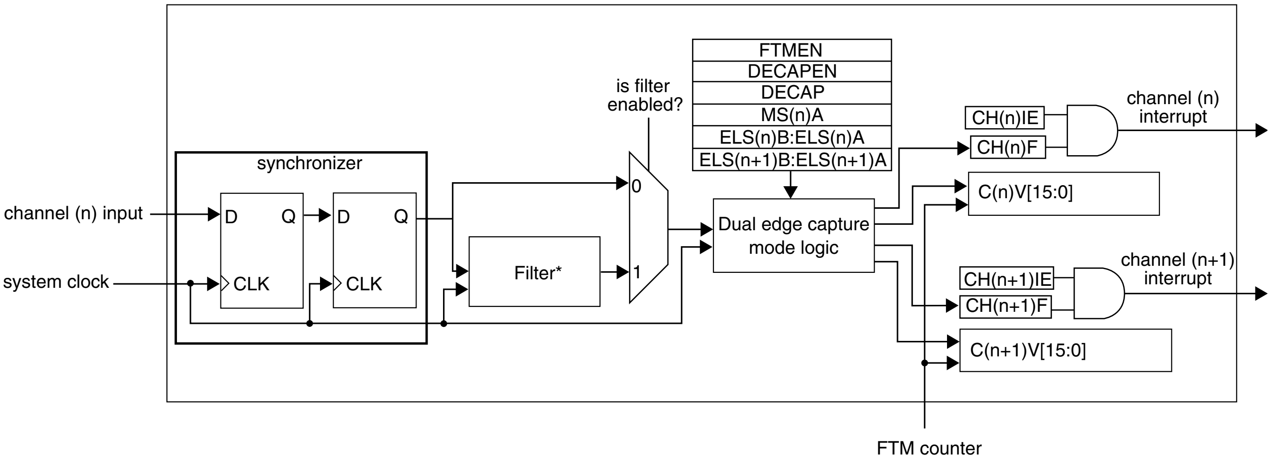
If DMA = 1, the CHnF bit is cleared either by channel DMA transfer done or reading CnSC while CHnF is set and then writing a zero to CHnF bit according to CHnIE bit (see the following table).

Table 40-308. Clear CHnF Bit when DMA = 1

CHnIE	How CHnF Bit Can Be Cleared
0	CHnF bit is cleared either when the channel DMA transfer is done or by reading CnSC while CHnF is set and then writing a 0 to CHnF bit.
1	CHnF bit is cleared when the channel DMA transfer is done.

40.4.24 Dual Edge Capture Mode

The dual edge capture mode is selected if FTMEN = 1 and DECAPEN = 1. This mode allows to measure a pulse width or period of the signal on the input of channel (n) of a channel pair. The channel (n) filter can be active in this mode when n is 0 or 2.



* Filtering function for dual edge capture mode is only available in the channels 0 and 2

Figure 40-285. Dual Edge Capture Mode Block Diagram

The MS(n)A bit defines if the dual edge capture mode is one-shot or continuous.

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). If both ELS(n)B:ELS(n)A and ELS(n+1)B:ELS(n+1)A bits select the same edge, then it is the period measurement. If these bits select different edges, then it is a pulse width measurement.

In the dual edge capture mode, only channel (n) input is used and channel (n+1) 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 and (CH(n)F = 1), 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 FTM counter when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the value of FTM counter when the selected edge by channel (n+1) is detected at channel (n) input.

In this mode, the pair channels coherency mechanism ensures coherent data when the C(n)V and C(n+1)V registers are read. The only requirement is that C(n)V must be read before C(n+1)V.

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.
- It is expected that the dual edge capture mode 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 and the FTM counter in free running counter mode (Free Running Counter).

40.4.24.1 One-Shot Capture Mode

The one-shot capture mode is selected when (FTMEN = 1), (DECAPEN = 1), and (MS(n)A = 0). In this capture mode, only one pair of edges at the channel (n) input is captured. The ELS(n)B:ELS(n)A bits select the first edge to be captured, and ELS(n +1)B:ELS(n+1)A bits select the second edge to be captured.

The edge captures are enabled while DECAP bit is set. For each new measurement in one-shot capture mode, first the CH(n)F and CH(n+1) bits must be cleared, and then the DECAP bit must be set.

In this mode, the DECAP bit is automatically cleared by FTM when the edge selected by channel (n+1) is captured. Therefore, while DECAP bit is set, the one-shot capture is in process. When this bit is cleared, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

Similarly, when the CH(n+1)F bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

40.4.24.2 Continuous Capture Mode

The continuous capture mode is selected when (FTMEN = 1), (DECAPEN = 1), and (MS(n)A = 1). In this capture mode, the edges at the channel (n) input are captured continuously. The ELS(n)B:ELS(n)A bits select the initial edge to be captured, and ELS(n+1)B:ELS(n+1)A bits select the final edge to be captured.

The edge captures are enabled while DECAP bit is set. For the initial use, first the CH(n)F and CH(n+1)F bits must be cleared, and then DECAP bit must be set to start the continuous measurements.

When the CH(n+1)F bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers. The latest captured values are always available in these registers even after the DECAP bit is cleared.

In this mode, it is possible to clear only the CH(n+1)F bit. Therefore, when the CH(n+1)F bit is set again, the latest captured values are available in C(n)V and C(n+1)V registers.

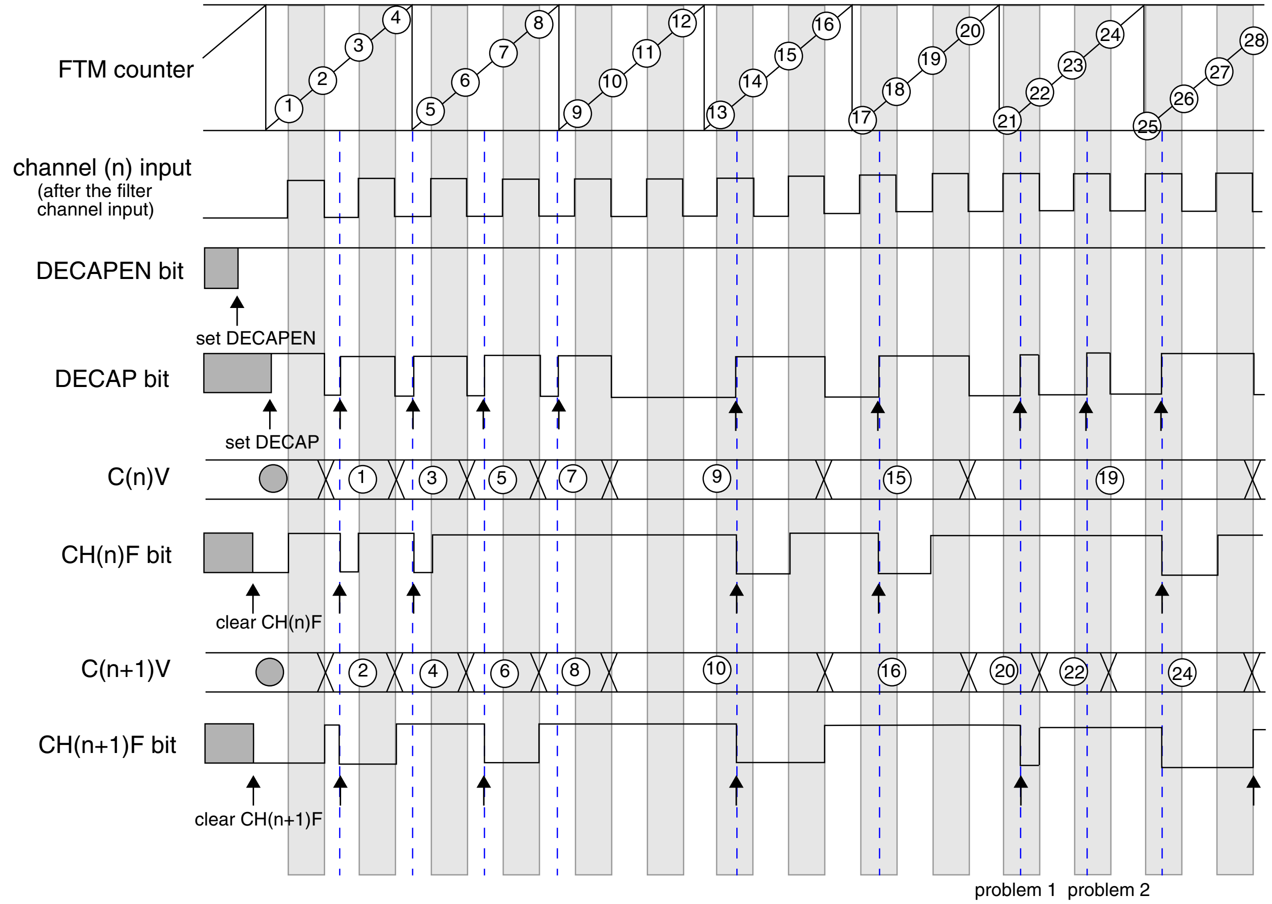
For a new sequence of the measurements in the dual edge capture – continuous mode, it is recommended to clear the CH(n)F and CH(n+1)F bits to start new measurements.

40.4.24.3 Pulse Width Measurement

If the channel (n) is configured to capture rising edges (ELS(n)B:ELS(n)A = 0:1) and the channel (n+1) to capture falling edges (ELS(n+1)B:ELS(n+1)A = 1:0), then the positive polarity pulse width is measured. If the channel (n) is configured to capture falling edges (ELS(n)B:ELS(n)A = 1:0) and the channel (n+1) to capture rising edges (ELS(n +1)B:ELS(n+1)A = 0:1), then the negative polarity pulse width is measured.

The pulse width measurement can be made in one-shot capture mode (One-Shot Capture Mode) or continuous capture mode (Continuous Capture Mode).

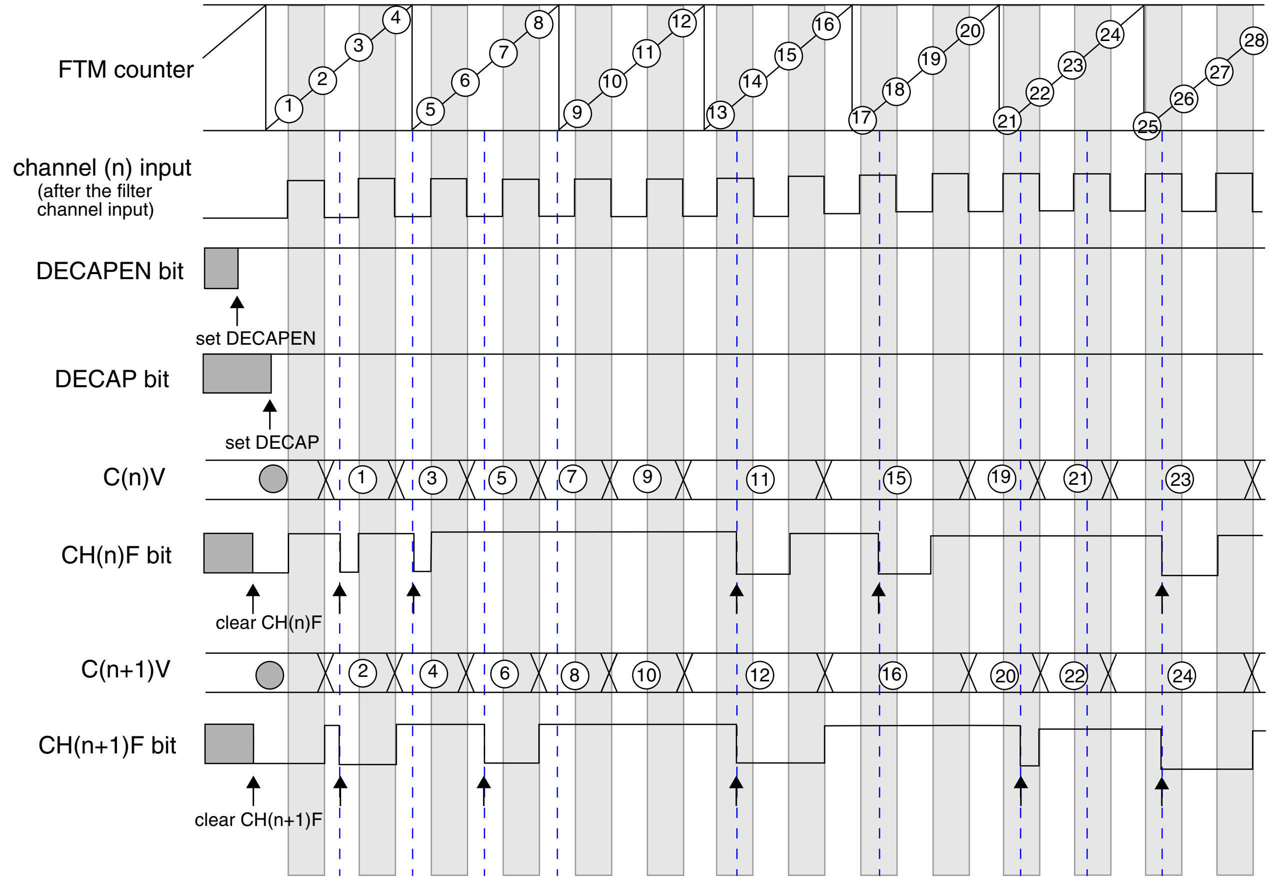
The following figure shows an example of the dual edge capture – one-shot mode used to measure the positive polarity pulse width. The DECAPEN bit selects the dual edge capture mode, so it keeps set in all operation mode. The DECAP bit is set to enable the measurement of next positive polarity pulse width. The CH(n)F bit is set when the first edge of this pulse is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set and DECAP bit is cleared when the second edge of this pulse is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. Both DECAP and CH(n+1)F bits indicate when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.
- Problem 1: channel (n) input = 1, set DECAP, not clear CH(n)F, and clear CH(n+1)F.
- Problem 2: channel (n) input = 1, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.

Figure 40-286. Dual Edge Capture – One-Shot Mode for Positive Polarity Pulse Width Measurement

The following figure shows an example of the dual edge capture – continuous mode used to measure the positive polarity pulse width. The DECAPEN bit selects the dual edge capture mode, so it keeps set in all operation mode. While the DECAP bit is set the configured measurements are made. The CH(n)F bit is set when the first edge of the positive polarity pulse is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set when the second edge of this pulse is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. The CH(n+1)F bit indicates when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.

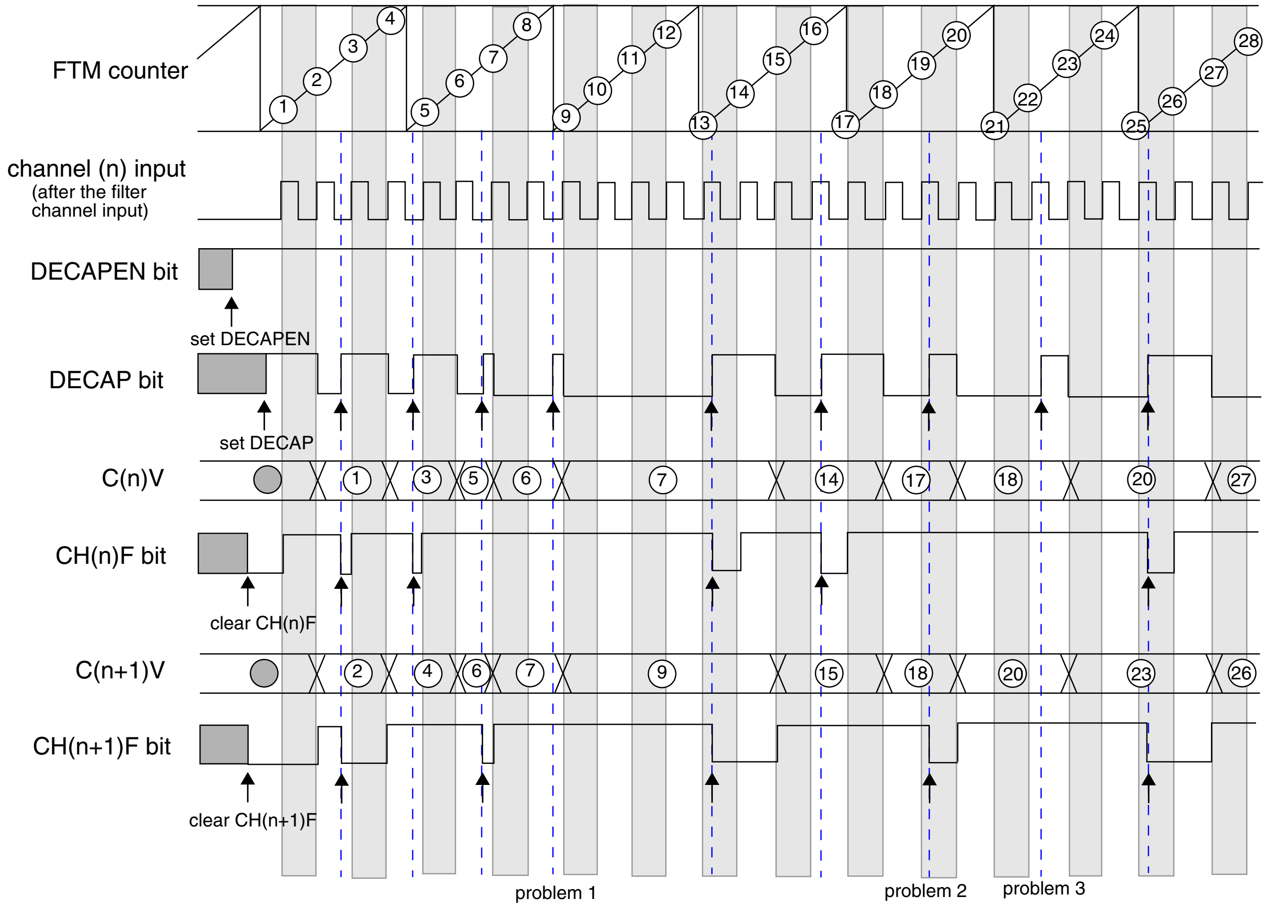
Figure 40-287. Dual Edge Capture – Continuous Mode for Positive Polarity Pulse Width Measurement

40.4.24.4 Period Measurement

If the channels (n) and (n+1) are configured to capture consecutive edges of the same polarity, then the period of the channel (n) input signal is measured. If both channels (n) and (n+1) are configured to capture rising edges (ELS(n)B:ELS(n)A = 0:1 and ELS(n+1)B:ELS(n+1)A = 0:1), then the period between two consecutive rising edges is measured. If both channels (n) and (n+1) are configured to capture falling edges (ELS(n)B:ELS(n)A = 1:0 and ELS(n+1)B:ELS(n+1)A = 1:0), then the period between two consecutive falling edges is measured.

The period measurement can be made in one-shot capture mode (One-Shot Capture Mode) or continuous capture mode (Continuous Capture Mode).

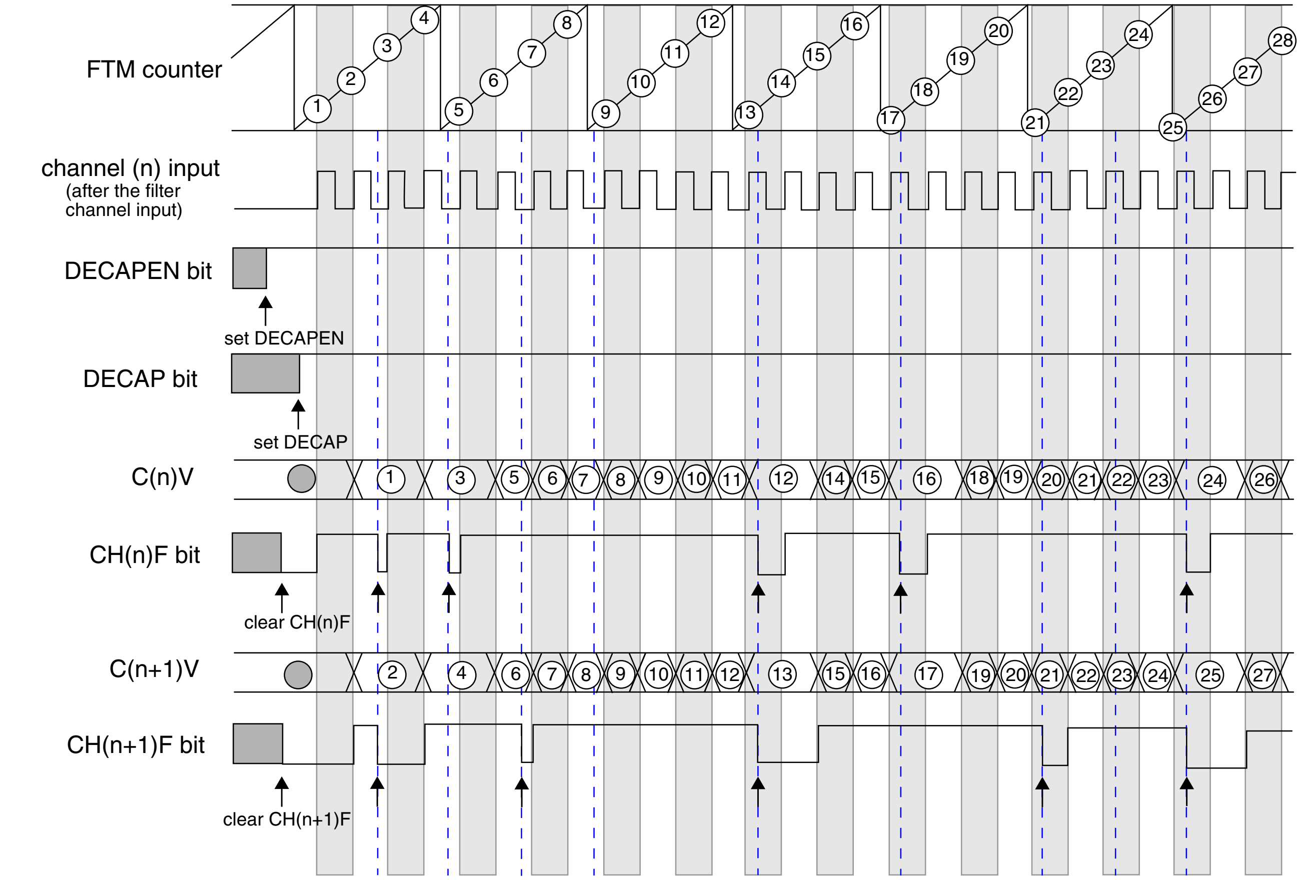
The following figure shows an example of the dual edge capture – one-shot mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the dual edge capture mode, so it keeps set in all operation mode. The DECAP bit is set to enable the measurement of next period. The CH(n)F bit is set when the first rising edge is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set and DECAP bit is cleared when the second rising edge is detected, that is, the edge selected by ELS(n+1)B:ELS(n+1)A bits. Both DECAP and CH(n+1)F bits indicate when two selected edges were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.
- Problem 1: channel (n) input = 0, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.
- Problem 2: channel (n) input = 1, set DECAP, not clear CH(n)F, and clear CH(n+1)F.
- Problem 3: channel (n) input = 1, set DECAP, not clear CH(n)F, and not clear CH(n+1)F.

Figure 40-288. Dual Edge Capture – One-Shot Mode to Measure of the Period Between Two Consecutive Rising Edges

The following figure shows an example of the dual edge capture – continuous mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the dual edge capture mode, so it keeps set in all operation mode. While the DECAP bit is set the configured measurements are made. The CH(n)F bit is set when the first rising edge is detected, that is, the edge selected by ELS(n)B:ELS(n)A bits. The CH(n+1)F bit is set when the second rising edge is detected, that is, the edge selected by ELS(n +1)B:ELS(n+1)A bits. The CH(n+1)F bit indicates when two edges of the period were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear CH(n)F, and clear CH(n+1)F are made by the user.

Figure 40-289. Dual Edge Capture – Continuous Mode to Measure of the Period Between Two Consecutive Rising Edges

40.4.24.5 Read Coherency Mechanism

The dual edge capture mode implements a read coherency mechanism between the FTM counter value captured in C(n)V and C(n+1)V registers. The read coherency mechanism is illustrated in the following figure. In this example, the channels (n) and (n+1) are in dual edge capture – continuous mode for positive polarity pulse width measurement. Thus, the channel (n) is configured to capture the FTM counter value when there is a rising edge at channel (n) input signal, and channel (n+1) to capture the FTM counter value when there is a falling edge at channel (n) input signal.

When a rising edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n) capture buffer. The channel (n) capture buffer value is transferred to C(n)V register when a falling edge occurs in the channel (n) input signal. C(n)V register has the FTM counter value when the previous rising edge occurred, and the channel (n) capture buffer has the FTM counter value when the last rising edge occurred.

When a falling edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n+1) capture buffer. The channel (n+1) capture buffer value is transferred to C(n+1)V register when the C(n)V register is read.

In the following figure, the read of C(n)V returns the FTM counter value when the event 1 occurred and the read of C(n+1)V returns the FTM counter value when the event 2 occurred.

C(n)V register must be read prior to C(n+1)V register in dual edge capture one-shot and continuous modes for the read coherency mechanism works properly.

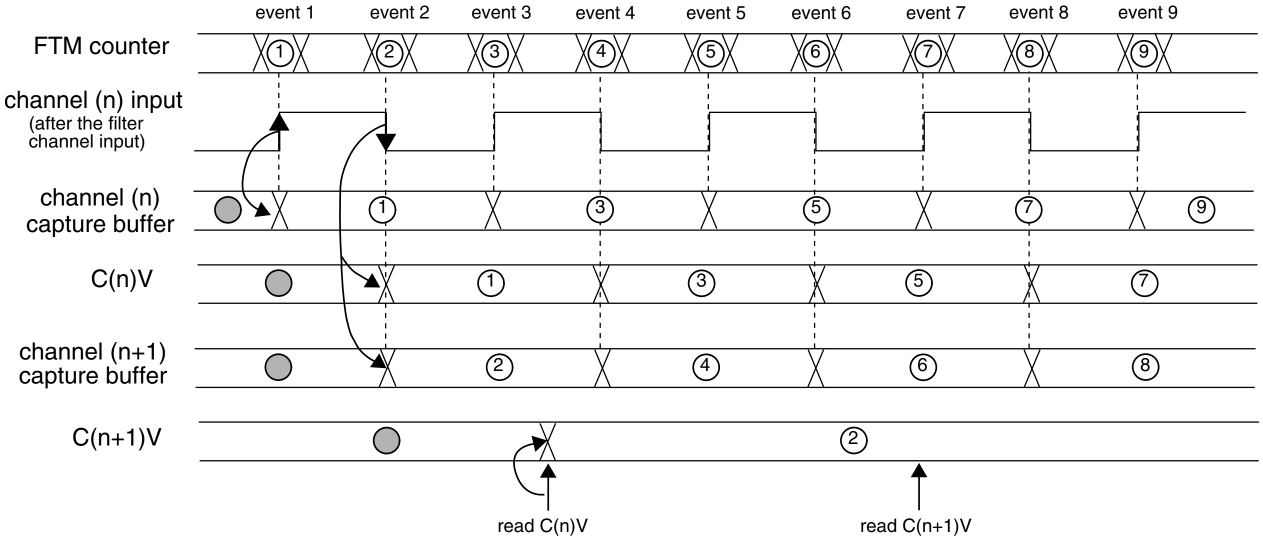


Figure 40-290. Dual Edge Capture Mode Read Coherency Mechanism

40.4.25 Quadrature Decoder Mode

The quadrature decoder mode is selected if (FTMEN = 1) and (QUADEN = 1). The quadrature decoder mode uses the input signals phase A and B to control the FTM counter increment and decrement. The following figure shows the quadrature decoder block diagram.

Each one of input signals phase A and B has a filter that is equivalent to the filter used in the channels input (Filter for Input Capture Mode). The phase A input filter is enabled by PHAFLTREN bit and this filter’s value is defined by CH0FVAL[3:0] bits

(CH(n)FVAL[3:0] bits in FILTER0 register). The phase B input filter is enabled by PHBFLTREN bit and this filter's value is defined by CH1FVAL[3:0] bits (CH(n +1)FVAL[3:0] bits in FILTER0 register).

Except for CH0FVAL[3:0] and CH1FVAL[3:0] bits, no channel logic is used in quadrature decoder mode.

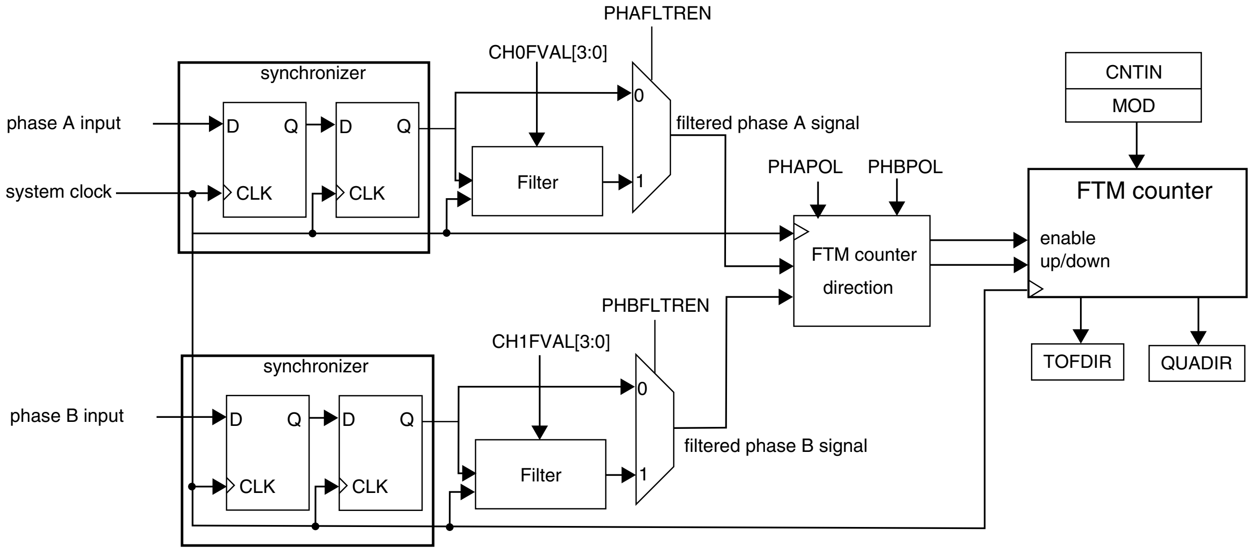


Figure 40-291. Quadrature Decoder Block Diagram

Note

It is important to notice that the FTM counter is clocked by the phase A and B input signals when quadrature decoder mode is selected. Therefore it is expected that the quadrature decoder be used only with the FTM channels in input capture or output compare modes.

The PHAPOL bit selects the polarity of the phase A input, and the PHBPOL bit selects the polarity of the phase B input.

The QUADMODE selects the encoding mode used in the quadrature decoder mode. If QUADMODE = 1, then the count and direction encoding mode (see the following figure) is enabled. In this mode, the phase B input value indicates the counting direction (FTM counter increment or decrement), and the phase A input defines the counting rate (FTM counter is updated when there is a rising edge at phase A input signal).

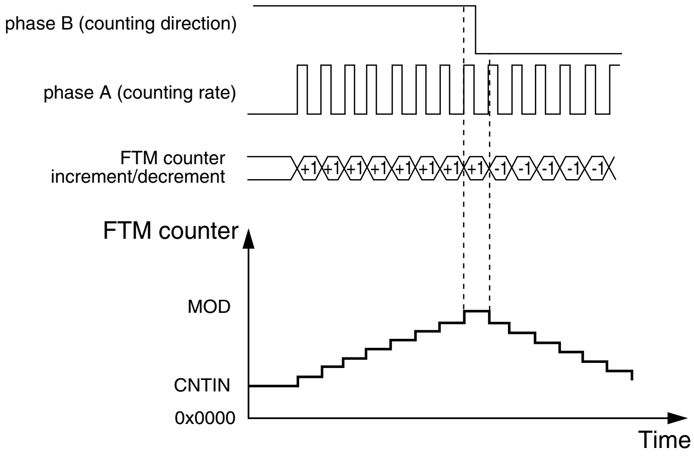


Figure 40-292. Quadrature Decoder – Count and Direction Encoding Mode

If QUADMODE = 0, then the phase A and phase B encoding mode (see the following figure) is enabled. In this mode, the relationship between phase A and B signals indicates the counting direction, and phase A and B signals define the counting rate (FTM counter is updated when there is an edge either at the phase A or phase B signals).

If PHAPOL = 0 and PHBPOL = 0, then the FTM counter increment happens when:

- there is a rising edge at phase A signal and phase B signal is at logic zero;
- there is a rising edge at phase B signal and phase A signal is at logic one;
- there is a falling edge at phase B signal and phase A signal is at logic zero;
- there is a falling edge at phase A signal and phase B signal is at logic one;

and the FTM counter decrement happens when:

- there is a falling edge at phase A signal and phase B signal is at logic zero;
- there is a falling edge at phase B signal and phase A signal is at logic one;
- there is a rising edge at phase B signal and phase A signal is at logic zero;
- there is a rising edge at phase A signal and phase B signal is at logic one.

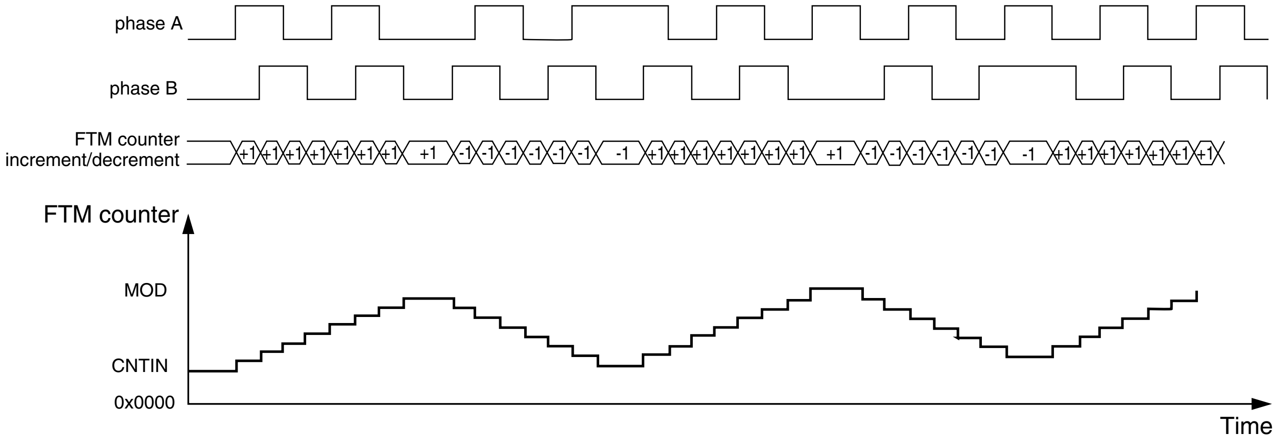


Figure 40-293. Quadrature Decoder – Phase A and Phase B Encoding Mode

The following figure shows the FTM counter overflow in up counting. In this case, when the FTM counter changes from MOD to CNTIN, TOF and TOFDIR bits are set. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was up when the FTM counter overflow occurred.

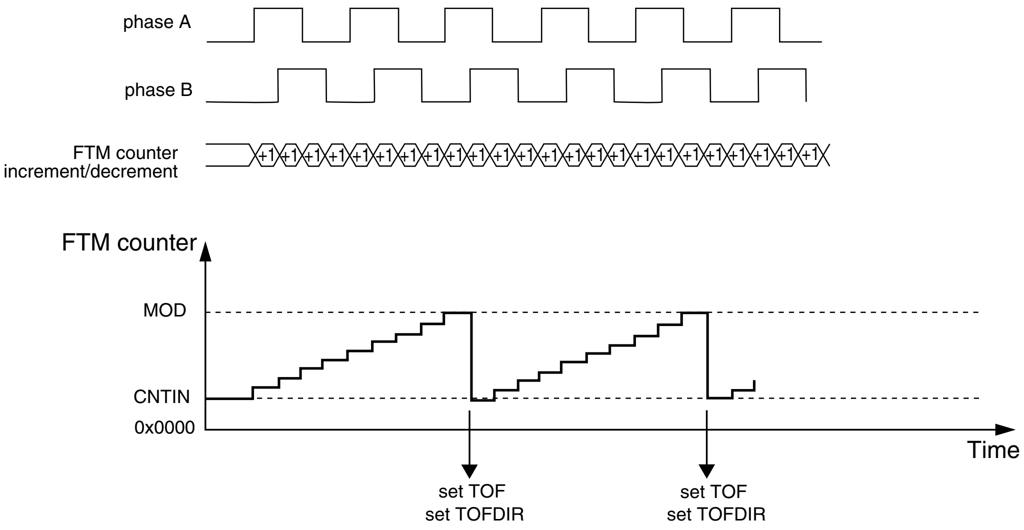


Figure 40-294. FTM Counter Overflow in Up Counting for Quadrature Decoder Mode

The following figure shows the FTM counter overflow in down counting. In this case, when the FTM counter changes from CNTIN to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was down when the FTM counter overflow occurred.

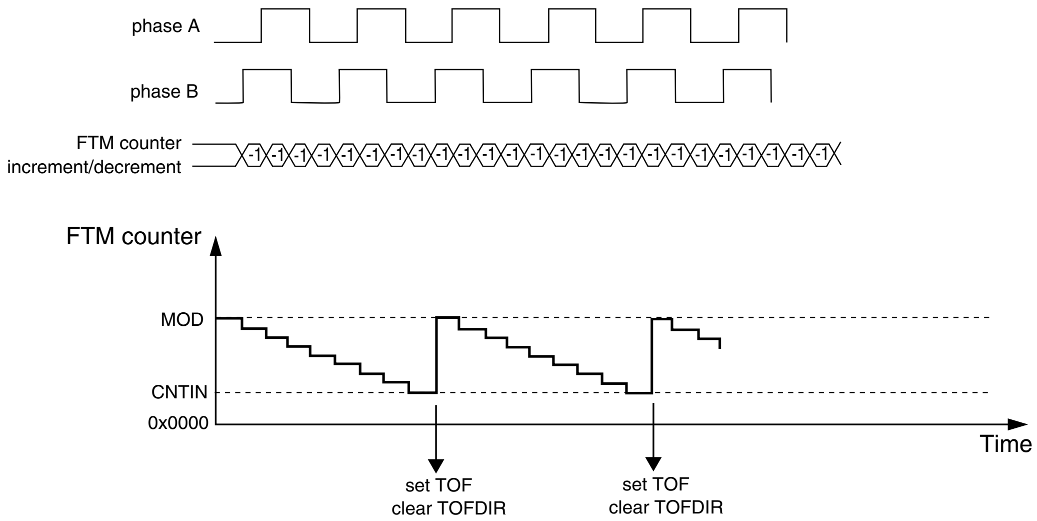


Figure 40-295. FTM Counter Overflow in Down Counting for Quadrature Decoder Mode

40.4.25.1 Quadrature Decoder Boundary Conditions

The following figures are examples of motor jittering which causes the FTM counter transitions as indicated by these figures. It is expected to observe these behaviors in motor position control applications.

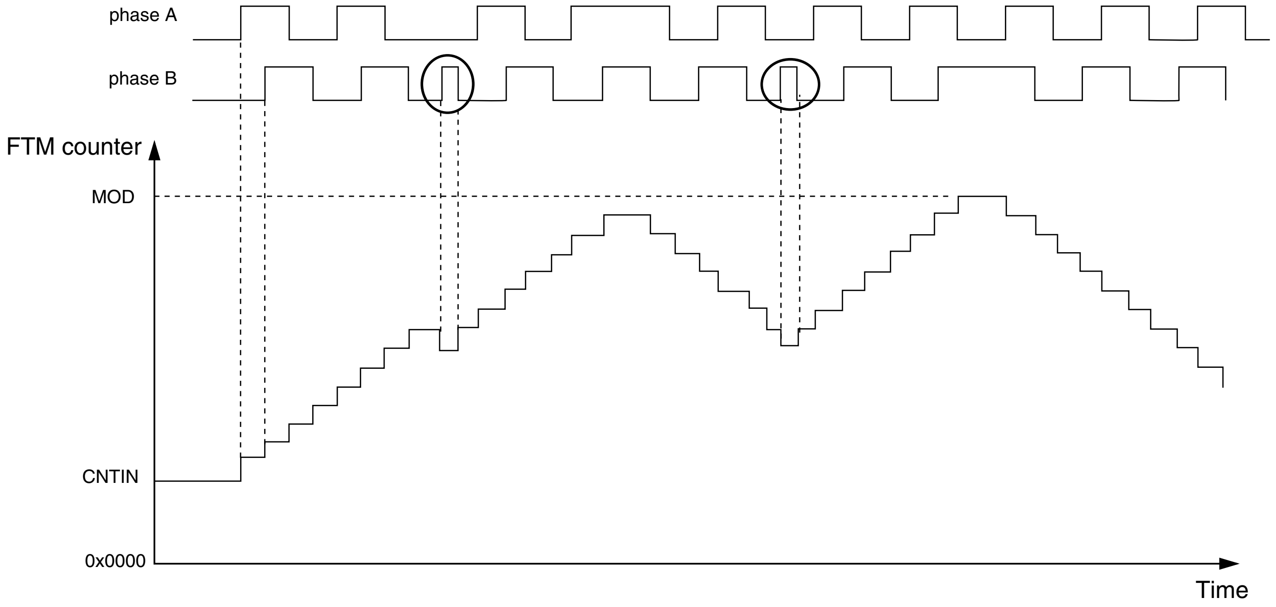


Figure 40-296. Motor Position Jittering in a Mid Count Value

The following figure shows motor jittering produced by the phase B and A pulses respectively. The first highlighted transition causes a jitter on the FTM counter value near the maximum count value (MOD). The second indicated transition occurs on phase A and causes the FTM counter transition between the maximum and minimum count values which are defined by MOD and CNTIN registers.

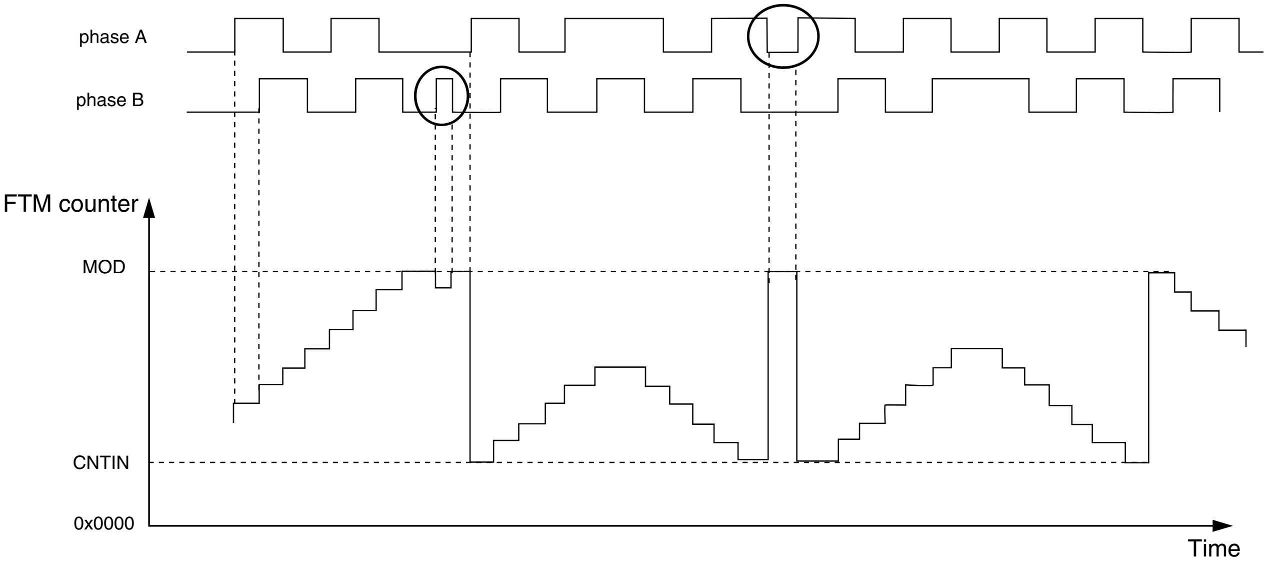


Figure 40-297. Motor Position Jittering Near Maximum and Minimum Count Value

The appropriate settings of the phase A and phase B input filters are important to avoid glitches that may cause oscillation on the FTM counter value. The preceding figures show examples of oscillations that can be caused by poor input filter setup. Thus, it is important to guarantee a minimum pulse width to avoid these oscillations.

40.4.26 BDM Mode

When the chip is in BDM mode, the BDMODE[1:0] bits select the behavior of the FTM counter, the CH(n)F bit, the channels output, and the writes to the MOD, CNTIN, and C(n)V registers according to the following table.

Table 40-309. FTM Behavior When the Chip Is in BDM Mode

BDMMODE	FTM Counter	CH(n)F Bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
00	Stopped	can be set	Functional mode	Writes to these registers bypass the registers buffers
01	Stopped	is not set	The channels outputs are forced to their safe value according to POLn bit	Writes to these registers bypass the registers buffers

Table continues on the next page...

Table 40-309. FTM Behavior When the Chip Is in BDM Mode (continued)

BDMMODE	FTM Counter	CH(n)F Bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
10	Stopped	is not set	The channels outputs are frozen when the chip enters in BDM mode	Writes to these registers bypass the registers buffers
11	Functional mode	can be set	Functional mode	Functional mode

Note that if BDMMODE[1:0] = 2’b00 then the channels outputs remain at the value when the chip enters in BDM mode, since the FTM counter is stopped. However, the following situations modify the channels outputs in this BDM mode.

- Write any value to CNT register (Counter Reset). In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for those channels set to output compare mode.
- FTM counter is reset by PWM synchronization mode (FTM Counter Synchronization). In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for channels in output compare mode.
- In the channels outputs initialization (Initialization), the channel (n) output is forced to the CH(n)OI bit value when the value 1 is written to INIT bit.

Note

It is expected that the BDMMODE[1:0] = 2’b00 is not used with the fault control (Fault Control). Even if the fault control is enabled and a fault condition exists, the channels outputs values are as defined above.

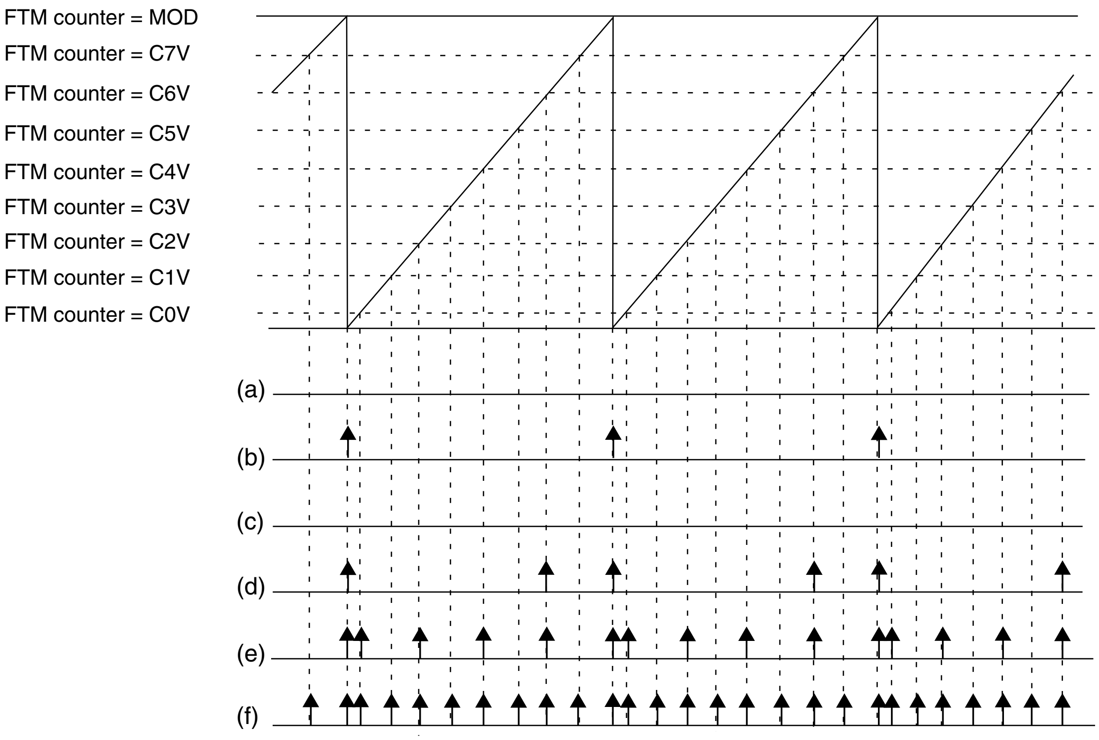
40.4.27 Intermediate Load

The PWMLOAD register allows software to update the MOD, CNTIN, and C(n)V registers with the content of the register buffer at a defined load point. In this case it is not required to use the PWM synchronization control.

A possible loading point is when the FTM counter wraps from MOD value to CNTIN value. This loading point is always enabled. Another possible loading point is at the channel (j) match (FTM counter = C(j)V). This loading point is enabled when CHjSEL = 1. The following figure shows some examples of enabled loading points.

After enabling the loading points, the LDOK bit needs to be set for the load to occur. In this case the load occurs at the next enabled loading point according to the following conditions:

- If a new value was written to the MOD register, then the MOD register is updated with its write buffer value.
- If a new value was written to the CNTIN register and CNTINC = 1, then the CNTIN register is updated with its write buffer value.
- If a new value was written to the C(n)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1), then the C(n)V register is updated with its write buffer value.
- If a new value was written to the C(n+1)V register and SYNCENm = 1 – where m indicates the pair channels (n) and (n+1), then the C(n+1)V register is updates with its write buffer value.



NOTE

(a) LDOK = 0, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
(b) LDOK = 1, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
(c) LDOK = 0, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 1, CH4SEL = 0, CH5SEL = 0, CH6SEL = 0, CH7SEL = 0
(d) LDOK = 1, CH0SEL = 0, CH1SEL = 0, CH2SEL = 0, CH3SEL = 0, CH4SEL = 0, CH5SEL = 0, CH6SEL = 1, CH7SEL = 0
(e) LDOK = 1, CH0SEL = 1, CH1SEL = 0, CH2SEL = 1, CH3SEL = 0, CH4SEL = 1, CH5SEL = 0, CH6SEL = 1, CH7SEL = 0
(f) LDOK = 1, CH0SEL = 1, CH1SEL = 1, CH2SEL = 1, CH3SEL = 1, CH4SEL = 1, CH5SEL = 1, CH6SEL = 1, CH7SEL = 1

Figure 40-298. Loading Points for Intermediate Load

NOTE

- If ELSjB and ELSjA bits are different from zero, then the channel (j) output signal is generated according to the configured output mode. If ELSjB and ELSjA bits are zero, then the generated signal is not available on channel (j) output.
- If CHjIE = 1, then the channel (j) interrupt is generated when the channel (j) match occurs.
- At the intermediate load neither the channels outputs nor the FTM counter are changed. Software must set the intermediate load at a safe point in time.
- It is expected that the intermediate load feature be used only in combine mode.

40.4.28 Global Time Base (GTB)

The global time base (GTB) is a FTM function that allows the synchronization of multiple FTM modules on a chip. The following figure shows an example of the GTB feature used to synchronize two FTM modules. In this case, the FTM A and B channels can behave as if just one FTM module was used, that is, a global time base.

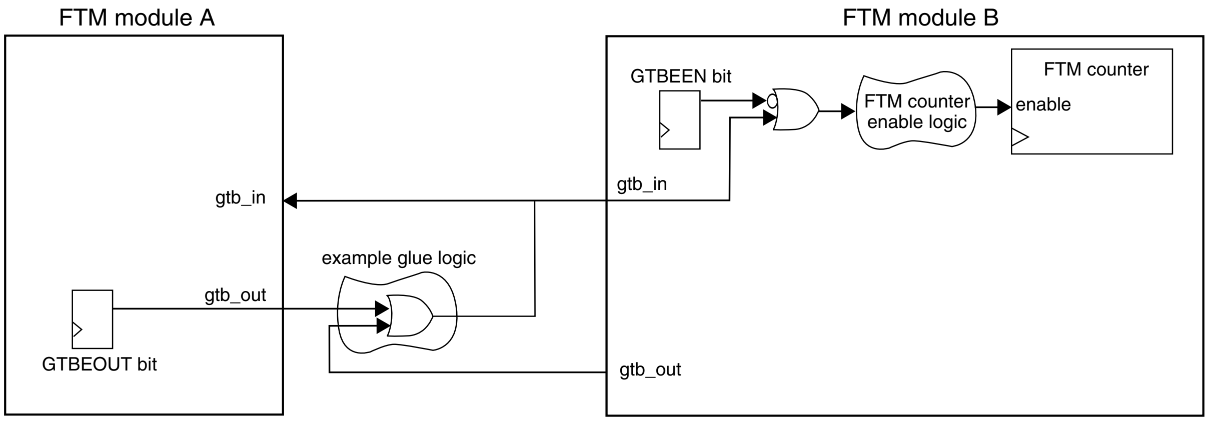


Figure 40-299. Global Time Base (GTB) Block Diagram

The GTB functionality is implemented by the GTBEEN and GTBEOUT bits in the CONF register, the internal input signal *gtb_in* and the internal output signal *gtb_out*. The GTBEEN bit enables gtb_in to control the FTM counter enable signal:

- If GTBEEN = 0, each one of FTM modules works independently according to their configured mode.
- If GTBEEN = 1, the FTM counter update is enabled only when gtb_in is 1.

In the configuration described in the preceding figure, FTM modules A and B have their FTM counters enabled if at least one of the gtb_out signals from one of the FTM modules is 1. There are several possible configurations for the interconnection of the gtb_in and gtb_out signals (represented by the example glue logic shown in the figure). Note that these configurations are chip-dependent and implemented outside of the FTM modules. See the Chip Configuration details for the chip's specific implementation.

NOTE

- In order to use the internal GTB signals to synchronize the FTM counter of different FTM modules, the configuration of each FTM module should guarantee that its FTM counter starts counting as soon as the gtb_in signal is 1.
- The GTB feature does not provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during FTM operation. The GTB feature only allows the FTM counters to *start* their operation synchronously.

40.4.28.1 Enabling the global time base (GTB)

To enable the GTB feature, follow these steps for each participating FTM module:

1. Stop the FTM counter: Write 00b to SC[CLKS].
2. Program the FTM module to the intended configuration. (The operation mode needs to be consistent across all participating modules.)
3. Write 1 to CONF[GTBEEN] and write 0 to CONF[GTBEOUT] at the same time.
4. Select the intended FTM counter clock source in SC[CLKS]. (The clock source needs to be consistent across all participating modules.)
5. Reset the FTM counter: Write any value to the CNT register.

To initiate the GTB feature, follow these steps for the FTM module used as the time base:

1. Write 1 to CONF[GTBEOUT].
2. If needed, configure the GTB glue logic connecting the FTM modules within the chip. Some chips do not require configuration of glue logic. See the Chip Configuration details for the chip's specific implementation.

40.5 Reset Overview

The FTM is reset whenever any chip reset occurs.

When the FTM exits from reset:

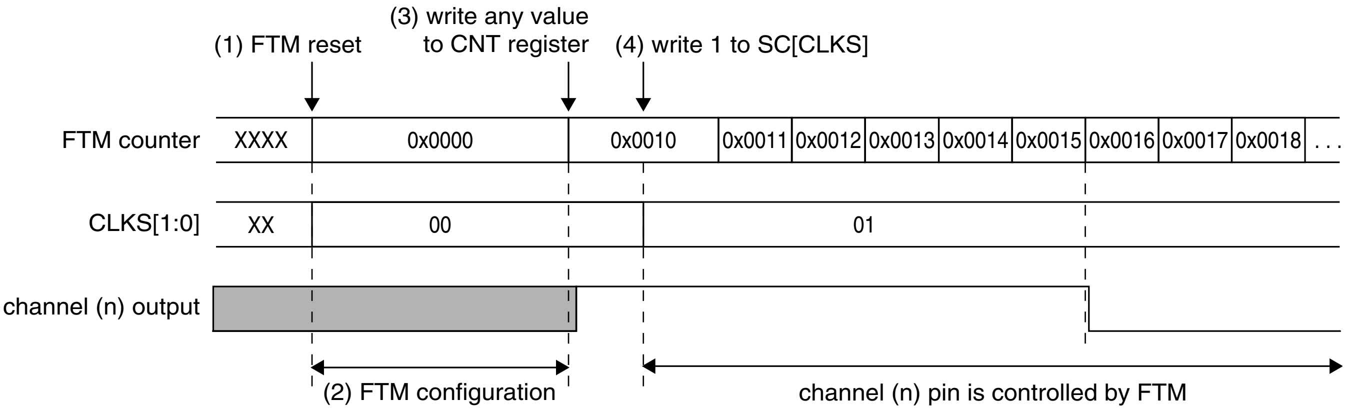
- the FTM counter and the prescaler counter are zero and are stopped (CLKS[1:0] = 00b);
- the timer overflow interrupt is zero (Timer Overflow Interrupt);
- the channels interrupts are zero (Channel (n) Interrupt);
- the fault interrupt is zero (Fault Interrupt);
- the channels are in input capture mode (Input Capture Mode);
- the channels outputs are zero;
- the channels pins are not controlled by FTM (ELS(n)B:ELS(n)A = 0:0) ().

The following figure shows the FTM behavior after the reset. At the reset (item 1), the FTM counter is disabled (see the description of the CLKS field in the Status and Control register), its value is updated to zero and the pins are not controlled by FTM ().

After the reset, the FTM should be configurated (item 2). It is necessary to define the FTM counter mode, the FTM counting limits (MOD and CNTIN registers value), the channels mode and CnV registers value according to the channels mode.

Thus, it is recommended to write any value to CNT register (item 3). This write updates the FTM counter with the CNTIN register value and the channels output with its initial value (except for channels in output compare mode) (Counter Reset).

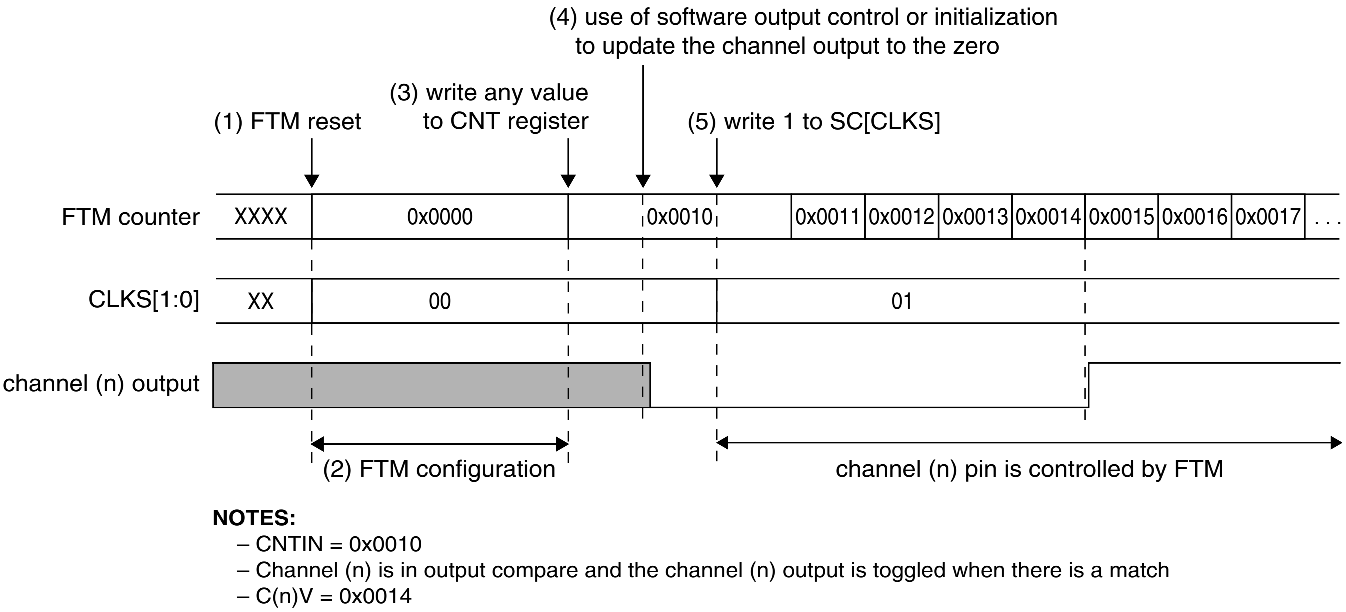
The next step is to select the FTM counter clock by the CLKS[1:0] bits (item 4). It is important to highlight that the pins are only controlled by FTM when CLKS[1:0] bits are different from zero ().



NOTES:
– CNTIN = 0x0010
– Channel (n) is in low-true combine mode with CNTIN < C(n)V < C(n+1)V < MOD
– C(n)V = 0x0015

Figure 40-300. FTM Behavior After Reset When the Channel (n) Is in Combine Mode

The following figure shows an example when the channel (n) is in output compare mode and the channel (n) output is toggled when there is a match. In the output compare mode, the channel output is not updated to its initial value when there is a write to CNT register (item 3). In this case, it is recommended to use the software output control (Software Output Control) or the initialization (Initialization) to update the channel output to the selected value (item 4).



NOTES:
– CNTIN = 0x0010
– Channel (n) is in output compare and the channel (n) output is toggled when there is a match
– C(n)V = 0x0014

Figure 40-301. FTM Behavior After Reset When the Channel (n) Is in Output Compare Mode

40.6 FTM Interrupts

This section describes FTM interrupts.

40.6.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

40.6.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

40.6.3 Fault Interrupt

The fault interrupt is generated when (FAULTIE = 1) and (FAULTF = 1).

Chapter 41
Periodic Interrupt Timer (PIT)

41.1 Introduction

NOTE

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

The PIT timer module is an array of timers that can be used to raise interrupts and trigger DMA channels.

41.1.1 Block Diagram

The following figure shows the PIT block diagram.

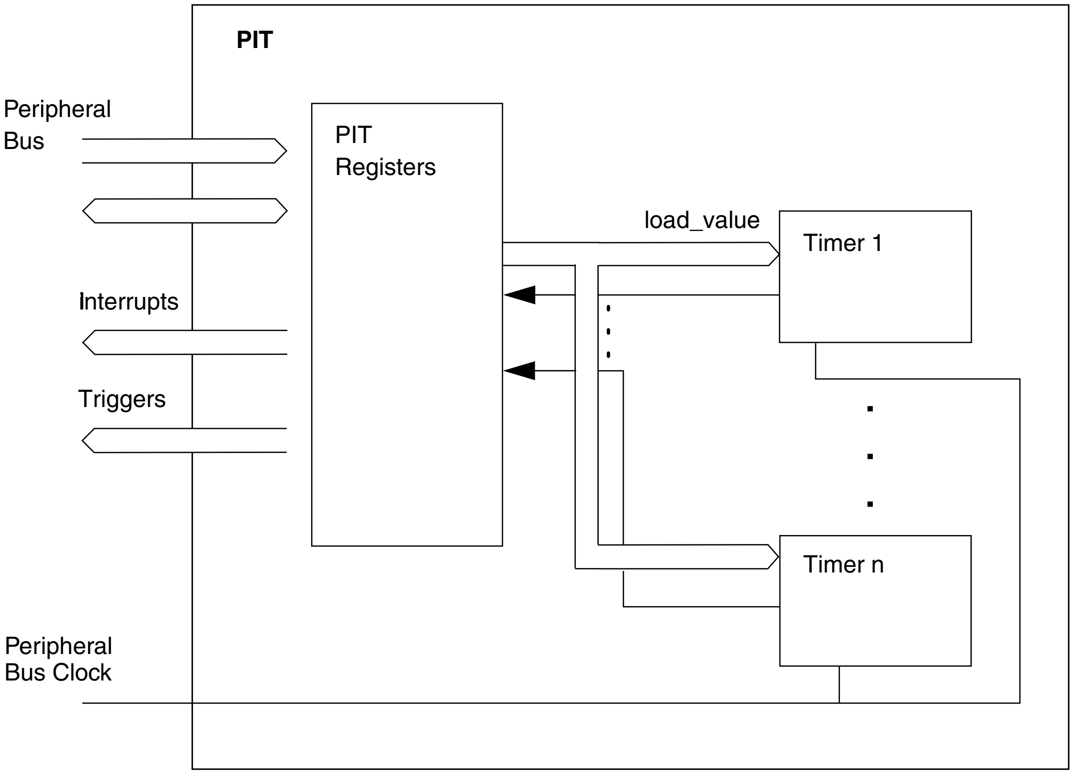


Figure 41-1. Block diagram of the PIT

NOTE

Refer to the Chip Configuration information for the number of PIT channels used in this MCU.

41.1.2 Features

The main features of this block are:

- Timers can generate DMA trigger pulses
- Timers can generate interrupts
- All interrupts are maskable
- Independent timeout periods for each timer

41.2 Signal Description

The PIT module has no external pins.

41.3 Memory Map/Register Description

This section provides a detailed description of all registers accessible in the PIT module.

NOTE

Reserved registers will read as 0, writes will have no effect.

NOTE

Refer to the Chip Configuration information for the number of PIT channels used in this MCU.

PIT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_7000	PIT Module Control Register (PIT_MCR)	32	R/W	0000_0002h	41.3.1/ 1184
4003_7100	Timer Load Value Register (PIT_LDVAL0)	32	R/W	0000_0000h	41.3.2/ 1185
4003_7104	Current Timer Value Register (PIT_CVAL0)	32	R/W	0000_0000h	41.3.3/ 1185
4003_7108	Timer Control Register (PIT_TCTRL0)	32	R/W	0000_0000h	41.3.4/ 1186
4003_710C	Timer Flag Register (PIT_TFLG0)	32	R/W	0000_0000h	41.3.5/ 1187
4003_7110	Timer Load Value Register (PIT_LDVAL1)	32	R/W	0000_0000h	41.3.2/ 1185
4003_7114	Current Timer Value Register (PIT_CVAL1)	32	R/W	0000_0000h	41.3.3/ 1185
4003_7118	Timer Control Register (PIT_TCTRL1)	32	R/W	0000_0000h	41.3.4/ 1186
4003_711C	Timer Flag Register (PIT_TFLG1)	32	R/W	0000_0000h	41.3.5/ 1187
4003_7120	Timer Load Value Register (PIT_LDVAL2)	32	R/W	0000_0000h	41.3.2/ 1185
4003_7124	Current Timer Value Register (PIT_CVAL2)	32	R/W	0000_0000h	41.3.3/ 1185
4003_7128	Timer Control Register (PIT_TCTRL2)	32	R/W	0000_0000h	41.3.4/ 1186
4003_712C	Timer Flag Register (PIT_TFLG2)	32	R/W	0000_0000h	41.3.5/ 1187

Table continues on the next page...

PIT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_7130	Timer Load Value Register (PIT_LDVAL3)	32	R/W	0000_0000h	41.3.2/ 1185
4003_7134	Current Timer Value Register (PIT_CVAL3)	32	R/W	0000_0000h	41.3.3/ 1185
4003_7138	Timer Control Register (PIT_TCTRL3)	32	R/W	0000_0000h	41.3.4/ 1186
4003_713C	Timer Flag Register (PIT_TFLG3)	32	R/W	0000_0000h	41.3.5/ 1187

41.3.1 PIT Module Control Register (PIT_MCR)

This register controls whether the timer clocks should be enabled and whether the timers should run in debug mode.

Address: PIT_MCR is 4003_7000h base + 0h offset = 4003_7000h

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																														MDIS	FRZ
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	1	0

PIT_MCR field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value zero.
1 MDIS	Module Disable This is used to disable the module clock. This bit must be enabled before any other setup is done. 0 Clock for PIT Timers is enabled. 1 Clock for PIT Timers is disabled.
0 FRZ	Freeze Allows the timers to be stopped when the device enters debug mode. 0 Timers continue to run in debug mode. 1 Timers are stopped in debug mode.

41.3.2 Timer Load Value Register (PIT_LDVAL*n*)

These registers select the timeout period for the timer interrupts.

Addresses: PIT_LDVAL0 is 4003_7000h base + 100h offset = 4003_7100h
PIT_LDVAL1 is 4003_7000h base + 110h offset = 4003_7110h
PIT_LDVAL2 is 4003_7000h base + 120h offset = 4003_7120h
PIT_LDVAL3 is 4003_7000h base + 130h offset = 4003_7130h

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	TSV																															
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

PIT_LDVAL*n* field descriptions

Field	Description
31–0 TSV	Timer Start Value Bits These bits set the timer start value. The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again. Writing a new value to this register will not restart the timer, instead the value will be loaded once the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again.

41.3.3 Current Timer Value Register (PIT_CVAL*n*)

These registers indicate the current timer position.

Addresses: PIT_CVAL0 is 4003_7000h base + 104h offset = 4003_7104h
PIT_CVAL1 is 4003_7000h base + 114h offset = 4003_7114h
PIT_CVAL2 is 4003_7000h base + 124h offset = 4003_7124h
PIT_CVAL3 is 4003_7000h base + 134h offset = 4003_7134h

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	TVL																															
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

PIT_CVAL*n* field descriptions

Field	Description
31–0 TVL	Current Timer Value If the timer is enabled, these bits represent the current timer value. If the timer is disabled, do not use this field as its value is unreliable.

PIT_CVAL*n* field descriptions (continued)

Field	Description
	NOTE: The timer uses a downcounter. The timer values are frozen in debug mode if the MCR[FRZ] bit is set.

41.3.4 Timer Control Register (PIT_TCTRL*n*)

These register contain the control bits for each timer.

Addresses: PIT_TCTRL0 is 4003_7000h base + 108h offset = 4003_7108h
PIT_TCTRL1 is 4003_7000h base + 118h offset = 4003_7118h
PIT_TCTRL2 is 4003_7000h base + 128h offset = 4003_7128h
PIT_TCTRL3 is 4003_7000h base + 138h offset = 4003_7138h

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																														TIE	TEN
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

PIT_TCTRL*n* field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value zero.
1 TIE	Timer Interrupt Enable Bit. When an interrupt is pending (TIF set), enabling the interrupt will immediately cause an interrupt event. To avoid this, the associated TIF flag must be cleared first. 0 Interrupt requests from Timer n are disabled. 1 Interrupt will be requested whenever TIF is set.
0 TEN	Timer Enable Bit. This bit enables or disables the timer. 0 Timer n is disabled. 1 Timer n is active.

41.3.5 Timer Flag Register (PIT_TFLG*n*)

These registers hold the PIT interrupt flags.

Addresses: PIT_TFLG0 is 4003_7000h base + 10Ch offset = 4003_710Ch
PIT_TFLG1 is 4003_7000h base + 11Ch offset = 4003_711Ch
PIT_TFLG2 is 4003_7000h base + 12Ch offset = 4003_712Ch
PIT_TFLG3 is 4003_7000h base + 13Ch offset = 4003_713Ch

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																															TIF
W																																w1c
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

PIT_TFLG*n* field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value zero.
0 TIF	Timer Interrupt Flag. TIF is set to 1 at the end of the timer period. This flag can be cleared only by writing it with 1. Writing 0 has no effect. If enabled (TIE = 1), TIF causes an interrupt request. 0 Time-out has not yet occurred. 1 Time-out has occurred.

41.4 Functional Description

This section provides the functional description of the module.

41.4.1 General

This section gives detailed information on the internal operation of the module. Each timer can be used to generate trigger pulses as well as to generate interrupts. Each interrupt is available on a separate interrupt line.

41.4.1.1 Timers

The timers generate triggers at periodic intervals, when enabled. They load their start values, as specified in their LDVAL registers, then count down until they reach 0. Then they load their respective start value again. Each time a timer reaches 0, it will generate a trigger pulse and set the interrupt flag.

All interrupts can be enabled or masked (by setting the TIE bits in the TCTRL registers). A new interrupt can be generated only after the previous one is cleared.

If desired, the current counter value of the timer can be read via the CVAL registers.

The counter period can be restarted, by first disabling, then enabling the timer with the TEN bit (see the following figure).

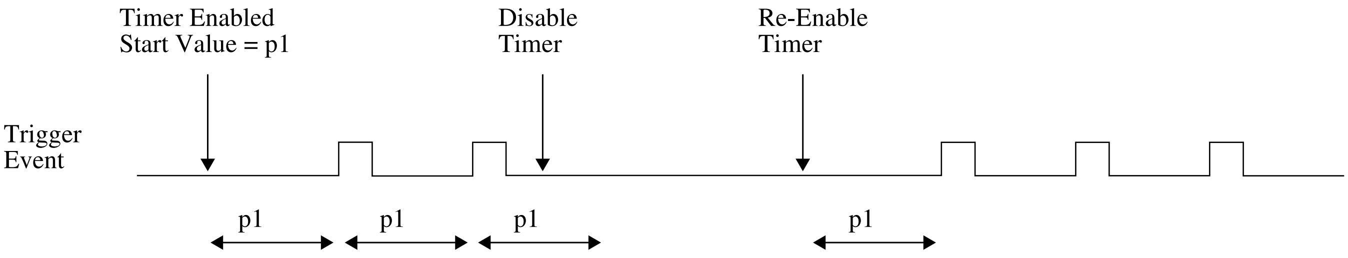


Figure 41-23. Stopping and Starting a Timer

The counter period of a running timer can be modified, by first disabling the timer, setting a new load value and then enabling the timer again (see the following figure).

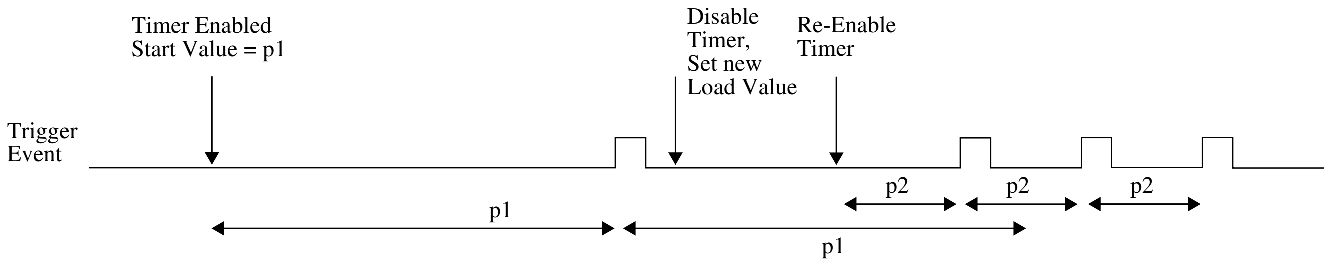


Figure 41-24. Modifying Running Timer Period

It is also possible to change the counter period without restarting the timer by writing the LDVAL register with the new load value. This value will then be loaded after the next trigger event (see the following figure).

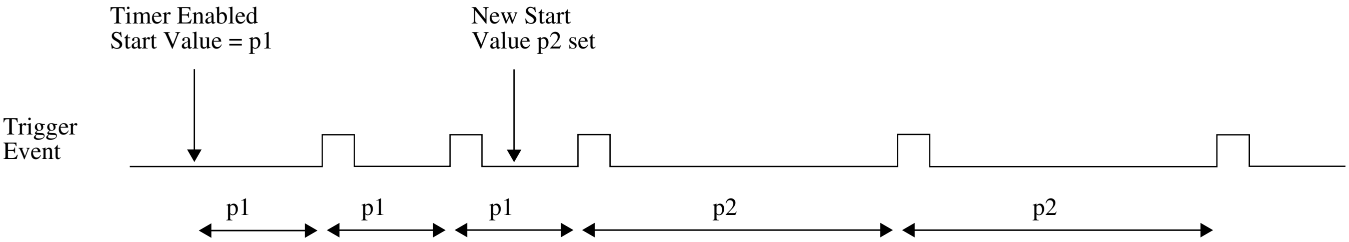


Figure 41-25. Dynamically Setting a New Load Value

41.4.1.2 Debug Mode

In debug mode, the timers will be frozen based on FRZ bit in PIT module control register. This is intended to aid software development, allowing the developer to halt the processor, investigate the current state of the system (for example, the timer values) and then continue the operation.

41.4.2 Interrupts

All of the timers support interrupt generation. Refer to the MCU specification for related vector addresses and priorities.

Timer interrupts can be enabled by setting the TIE bits. The timer interrupt flags (TIF) are set to 1 when a timeout occurs on the associated timer, and are cleared to 0 by writing a 1 to that TIF bit.

41.5 Initialization and Application Information

In the example configuration:

- The PIT clock has a frequency of 50 MHz.
- Timer 1 creates an interrupt every 5.12 ms.
- Timer 3 creates a trigger event every 30 ms.

First the PIT module must be activated by writing a 0 to the MDIS bit in the MCR.

The 50 MHz clock frequency equates to a clock period of 20 ns. Timer 1 needs to trigger every 5.12 ms/20 ns = 256000 cycles and timer 3 every 30 ms/20 ns = 1500000 cycles. The value for the LDVAL register trigger is calculated as:

LDVAL trigger = (period / clock period) -1

This means LDVAL1 should be written with 0x0003E7FF, and LDVAL3 should be written with 0x0016E35F.

The interrupt for Timer 1 is enabled by setting TIE in the TCTRL1 register. The timer is started by writing 1 to bit TEN in the TCTRL1 register.

Timer 3 shall be used only for triggering. Therefore Timer 3 is started by writing a 1 to bit TEN in the TCTRL3 register, bit TIE stays at 0.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;


// Timer 1
PIT_LDVAL1 = 0x0003E7FF; // setup timer 1 for 256000 cycles
PIT_TCTRL1 = TIE; // enable Timer 1 interrupts
PIT_TCTRL1 |= TEN; // start Timer 1


// Timer 3
PIT_LDVAL3 = 0x0016E35F; // setup timer 3for 1500000 cycles
PIT_TCTRL3 |= TEN; // start Timer 3
```

Chapter 42
Low power timer (LPTMR)

42.1 Introduction

NOTE

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

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.

42.1.1 Features

The LPTMR module's features 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

42.1.2 Modes of operation

42.1.2.1 Run mode

In run mode, the LPTMR operates normally.

42.1.2.2 Wait mode

In wait mode, the LPTMR continues to operate normally and may be configured to exit the low power mode by generating an interrupt request.

42.1.2.3 Stop mode

In stop mode, the LPTMR continues to operate normally and may be configured to exit the low power mode by generating an interrupt request.

42.1.2.4 Low leakage modes

In low leakage modes, the LPTMR continues to operate normally and may be configured to exit the low power mode by generating an interrupt request.

42.1.2.5 Debug modes

During debug mode, the LPTMR continues to operate normally.

42.2 LPTMR signal descriptions

Table 42-1. LPTMR signal descriptions

Signal	Description	I/O
LPTMR_ALT*n*	Pulse counter input pin	I

42.2.1 Detailed signal descriptions

Table 42-2. 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 LPTMR counter register to increment. Negation-If configured for pulse counter mode with active low input then negation cause the LPTMR counter register to increment.
		Timing	Assertion or negation may occur at any time; input may assert asynchronously to the bus clock.

42.3 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.

LPTMR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_0000	Low Power Timer Control Status Register (LPTMR0_CSR)	32	R/W	0000_0000h	42.3.1/ 1194
4004_0004	Low Power Timer Prescale Register (LPTMR0_PSR)	32	R/W	0000_0000h	42.3.2/ 1195
4004_0008	Low Power Timer Compare Register (LPTMR0_CMR)	32	R/W	0000_0000h	42.3.3/ 1197
4004_000C	Low Power Timer Counter Register (LPTMR0_CNR)	32	R	0000_0000h	42.3.4/ 1197

42.3.1 Low Power Timer Control Status Register (LPTMRx_CSR)

Addresses: LPTMR0_CSR is 4004_0000h base + 0h offset = 4004_0000h

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								TCF	TIE	TPS		TPP	TFC	TMS	TEN
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LPTMRx_CSR field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 TCF	Timer Compare Flag The timer compare flag is set when the LPTMR is enabled and the LPTMR Counter Register equals the LPTMR Compare Register and increments. This Timer Compare Flag is cleared when the LPTMR is disabled or a logic one is written to the Timer Compare Flag. 0 LPTMR Counter Register has not equaled the LPTMR Compare Register and incremented 1 LPTMR Counter Register has equaled the LPTMR Compare Register and incremented
6 TIE	Timer Interrupt Enable When the Timer Interrupt Enable is set, the LPTMR Interrupt is generated whenever the Timer Compare Flag is also set. 0 Timer Interrupt Disabled. 1 Timer Interrupt Enabled.
5–4 TPS	Timer Pin Select The Timer Pin Select configures the input source to be used in Pulse Counter mode. The Timer Pin Select should only be altered when the LPTMR is disabled. The input connections vary by device. See the Chip Configuration details for information on the connections to these inputs. 00 Pulse counter input 0 is selected. 01 Pulse counter input 1 is selected. 10 Pulse counter input 2 is selected. 11 Pulse counter input 3 is selected.
3 TPP	Timer Pin Polarity

Table continues on the next page...

LPTMRx_CSR field descriptions (continued)

Field	Description
	The Timer Pin Polarity configures the polarity of the input source in Pulse Counter mode. The Timer Pin Polarity should only be changed when the LPTMR is disabled. 0 Pulse Counter input source is active high, and LPTMR Counter Register will increment on the rising edge. 1 Pulse Counter input source is active low, and LPTMR Counter Register will increment on the falling edge.
2 TFC	Timer Free Running Counter When clear the Timer Free Running Counter configures the LPTMR Counter Register to reset whenever the Timer Compare Flag is set. When set, the Timer Free Running Counter configures the LPTMR Counter Register to reset on overflow. The Timer Free Running Counter should only be altered when the LPTMR is disabled. 0 LPTMR Counter Register is reset whenever the Timer Compare Flag is set. 1 LPTMR Counter Register is reset on overflow.
1 TMS	Timer Mode Select The Timer Mode Select configures the mode of the LPTMR. The Timer Mode Select should only be altered when the LPTMR is disabled. 0 Time Counter mode. 1 Pulse Counter mode.
0 TEN	Timer Enable When the Timer Enable bit is clear, it resets the LPTMR internal logic (including the LPTMR Counter Register and Timer Compare Flag). When the Timer Enable bit is set, the LPTMR is enabled. When writing 1 to this bit, bits LPTMR_CSR[5:1] should not be altered. 0 LPTMR is disabled and internal logic is reset. 1 LPTMR is enabled.

42.3.2 Low Power Timer Prescale Register (LPTMRx_PSR)

Addresses: LPTMR0_PSR is 4004_0000h base + 4h offset = 4004_0004h

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																									PRESCALE				PBYP	PCS	
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

LPTMRx_PSR field descriptions

Field	Description
31–7 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

LPTMRx_PSR field descriptions (continued)

Field	Description
6–3 PRESCALE	Prescale Value The Prescaler Value register field configures the size of the Prescaler (in Time Counter mode) or width of the Glitch Filter (in Pulse Counter mode). The Prescale Value should only be altered when the LPTMR is disabled. 0000 Prescaler divides the prescaler clock by 2; Glitch Filter does not support this configuration. 0001 Prescaler divides the prescaler clock by 4; Glitch Filter recognizes change on input pin after 2 rising clock edges. 0010 Prescaler divides the prescaler clock by 8; Glitch Filter recognizes change on input pin after 4 rising clock edges. 0011 Prescaler divides the prescaler clock by 16; Glitch Filter recognizes change on input pin after 8 rising clock edges. 0100 Prescaler divides the prescaler clock by 32; Glitch Filter recognizes change on input pin after 16 rising clock edges. 0101 Prescaler divides the prescaler clock by 64; Glitch Filter recognizes change on input pin after 32 rising clock edges. 0110 Prescaler divides the prescaler clock by 128; Glitch Filter recognizes change on input pin after 64 rising clock edges. 0111 Prescaler divides the prescaler clock by 256; Glitch Filter recognizes change on input pin after 128 rising clock edges. 1000 Prescaler divides the prescaler clock by 512; Glitch Filter recognizes change on input pin after 256 rising clock edges. 1001 Prescaler divides the prescaler clock by 1024; Glitch Filter recognizes change on input pin after 512 rising clock edges. 1010 Prescaler divides the prescaler clock by 2048; Glitch Filter recognizes change on input pin after 1024 rising clock edges. 1011 Prescaler divides the prescaler clock by 4096; Glitch Filter recognizes change on input pin after 2048 rising clock edges. 1100 Prescaler divides the prescaler clock by 8192; Glitch Filter recognizes change on input pin after 4096 rising clock edges. 1101 Prescaler divides the prescaler clock by 16384; Glitch Filter recognizes change on input pin after 8192 rising clock edges. 1110 Prescaler divides the prescaler clock by 32768; Glitch Filter recognizes change on input pin after 16384 rising clock edges. 1111 Prescaler divides the prescaler clock by 65536; Glitch Filter recognizes change on input pin after 32768 rising clock edges.
2 PBYP	Prescaler Bypass When the Prescaler Bypass is set the selected prescaler clock (in Time Counter mode) or selected input source (in Pulse Counter mode) directly clocks the LPTMR Counter Register. When the Prescaler Bypass is clear, the LPTMR Counter Register is clocked by the output of the prescaler/glitch filter. The Prescaler Bypass should only be altered when the LPTMR is disabled. 0 Prescaler/Glitch Filter is enabled. 1 Prescaler/Glitch Filter is bypassed.
1–0 PCS	Prescaler Clock Select The Prescaler Clock Select selects the clock to be used by the LPTMR prescaler/glitch filter. The Prescaler Clock Select should only be altered when the LPTMR is disabled. The clock connections vary by device. See the Chip Configuration details for information on the connections to these inputs.

Table continues on the next page...

LPTMRx_PSR field descriptions (continued)

Field	Description
	00 Prescaler/glitch filter clock 0 selected 01 Prescaler/glitch filter clock 1 selected 10 Prescaler/glitch filter clock 2 selected 11 Prescaler/glitch filter clock 3 selected

42.3.3 Low Power Timer Compare Register (LPTMRx_CMR)

Addresses: LPTMR0_CMR is 4004_0000h base + 8h offset = 4004_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	0																COMPARE															
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

LPTMRx_CMR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 COMPARE	Compare Value When the LPTMR is enabled and the LPTMR Counter Register equals the value in the LPTMR Compare Register and increments, the Timer Compare Flag is set and the Hardware Trigger asserts until the next time the LPTMR Counter Register increments. If the LPTMR Compare Register is zero, the Hardware Trigger will remain asserted until the LPTMR is disabled. If the LPTMR is enabled, the LPTMR Compare Register should only be altered when the Timer Compare Flag is set.

42.3.4 Low Power Timer Counter Register (LPTMRx_CNR)

Addresses: LPTMR0_CNR is 4004_0000h base + Ch offset = 4004_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	0																COUNTER															
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

LPTMRx_CNR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 COUNTER	Counter Value

LPTMRx_CNR field descriptions (continued)

Field	Description

42.4 Functional description

42.4.1 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 should be disabled before entering the mode.

The LPTMR is reset only on global POR or LVD. When configuring the LPTMR registers, the control status register should be initially written with the timer disabled, before configuring the LPTMR prescale register and compare register. The timer enable should then 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.

42.4.2 LPTMR clocking

The LPTMR prescaler/glitch filter can be clocked by one of four clocks. The clock source should be enabled before the LPTMR is enabled.

NOTE

> The clock source selected 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 LPTMR counter register 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.

42.4.3 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.

The prescaler/glitch filter configuration must not be altered when the LPTMR is enabled.

42.4.3.1 Prescaler enabled

In time counter mode when the prescaler is enabled, the output of the prescaler directly clocks the LPTMR counter register. When the LPTMR is enabled, the LPTMR counter register will increment every 2^2 to 2^{16} prescaler clock cycles. After the LPTMR is enabled, the first increment of the LPTMR counter register will take an additional one or two prescaler clock cycles due to synchronization logic.

42.4.3.2 Prescaler bypassed

In time counter mode when the prescaler is bypassed, the selected prescaler clock increments the LPTMR counter register 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.

42.4.3.3 Glitch filter

In pulse counter mode when the glitch filter is enabled, the output of the glitch filter directly clocks the LPTMR counter register. When the LPTMR is first enabled, the output of the glitch filter is asserted (logic one for active high and logic zero for active low). If the selected input source remains negated for at least 2^1 to 2^{15} consecutive prescaler clock rising edges, then the glitch filter output will also negate. If the selected input source remains asserted for at least 2^1 to 2^{15} consecutive prescaler clock rising edges, then the glitch filter output will also assert. Note that the input is only sampled on the rising clock edge.

The LPTMR counter register will increment each time the glitch filter output asserts. In pulse counter mode, the maximum rate at which the LPTMR counter register 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.

42.4.3.4 Glitch filter bypassed

In pulse counter mode when the glitch filter is bypassed, the selected input source increments the LPTMR counter register every time it asserts. Before the LPTMR is first enabled, the selected input source is forced to asserted. This is to prevent the LPTMR counter register from incrementing if the selected input source is already asserted when the LPTMR is first enabled.

42.4.4 LPTMR compare

When the LPTMR counter register equals the value of the LPTMR compare register and increments, the following events occur:

- Timer compare flag is set
- LPTMR interrupt is generated if Timer Interrupt Enable is also set
- LPTMR hardware trigger is generated
- LPTMR counter register is reset if the free running counter bit is clear

When the LPTMR is enabled, the LPTMR compare register can only be altered when the timer compare flag is set. When updating the LPTMR compare register, the LPTMR compare register must be written and the timer compare flag must be cleared before the LPTMR counter has incremented past the new LPTMR compare value.

42.4.5 LPTMR counter

The LPTMR counter register increments by one on every:

- prescaler clock (time counter mode with prescaler bypassed)
- prescaler output (time counter mode with prescaler enabled)
- input source assertion (pulse counter mode with glitch filter bypassed)
- glitch filter output (pulse counter mode with glitch filter enabled).

The LPTMR counter register is reset when the LPTMR is disabled or if the counter register overflows. If the CSR[TFC] control bit is set then the LPTMR counter register is also reset whenever the CSR[TCF] status flag is set.

The LPTMR counter register continues incrementing when the core is halted in debug mode.

The LPTMR counter register cannot be initialized, but can be read at any time. On each read of the LPTMR counter register, software must first write to the LPTMR counter register with any value. This will synchronize and register the current value of the LPTMR counter register into a temporary register. The contents of the temporary register are returned on each read of the LPTMR counter register.

When reading the LPTMR counter register, 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.

42.4.6 LPTMR hardware trigger

The LPTMR hardware trigger asserts at the same time the timer compare flag is set and can be used to trigger hardware events in other peripherals without software intervention. The hardware trigger is always enabled.

When the LPTMR compare register is set to zero with the free running counter bit clear, the LPTMR hardware trigger will assert on the first compare and does not negate. When the LPTMR compare register is set to a non-zero value (or if the free running counter bit is set) the LPTMR hardware trigger will assert on each compare and negate on the following increment of the LPTMR counter register.

42.4.7 LPTMR interrupt

The LPTMR interrupt is generated whenever the CSR[TIE] and CSR[TCF] are set. The CSR[TCF] is cleared by disabling the LPTMR or by writing a logic one to it.

The CSR[TIE] can be altered and the 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).

Chapter 43
Carrier Modulator Transmitter (CMT)

43.1 Introduction

NOTE

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

The carrier modulator transmitter (CMT) module provides means to generate the protocol timing and carrier signals for a wide variety of encoding schemes. The CMT incorporates hardware to off-load the critical and/or lengthy timing requirements associated with signal generation from the CPU, releasing much of its bandwidth to handle other tasks such as code data generation, data decompression, or keyboard scanning. The CMT does not include dedicated hardware configurations for specific protocols but is intended to be sufficiently programmable in its function to handle the timing requirements of most protocols with minimal CPU intervention. When the modulator is disabled, certain CMT registers can be used to change the state of the infrared output (CMT_IRO) signal directly. This feature allows for the generation of future protocol timing signals not readily producible by the current architecture.

43.2 Features

The features of this module include:

- Four modes of operation
  - Time; with independent control of high and low times
  - Baseband
  - Frequency shift key (FSK)
  - Direct software control of CMT_IRO signal

- Extended space operation in time, baseband, and FSK modes
- Selectable input clock divider
- Interrupt on end of cycle
  - Ability to disable CMT_IRO signal and use as timer interrupt

43.3 Block Diagram

The following figure is the CMT block diagram.

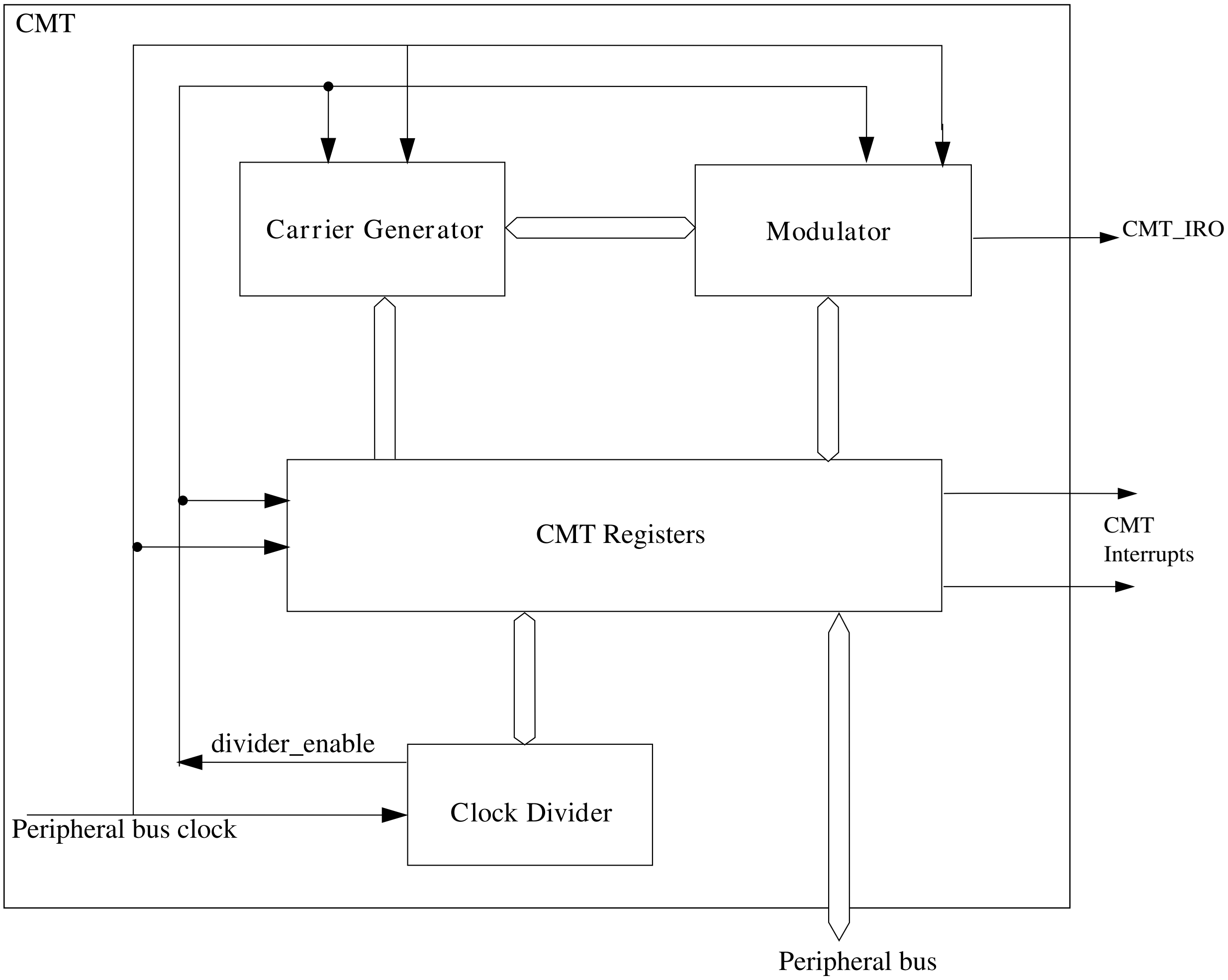


Figure 43-1. CMT Module Block Diagram

43.4 Modes of Operation

The CMT module operates in the following modes.

- Time—When operating in time mode, the user independently defines the high and low times of the carrier signal to determine both period and duty cycle.
- Baseband—When MSC[BASE] bit is set, the carrier output (f_{cg}) to the modulator is held high continuously to allow for the generation of baseband protocols.
- Frequency shift key (FSK)—This mode allows the carrier generator to alternate between two sets of high and low times . When operating in FSK mode, the generator will toggle between the two sets when instructed by the modulator, allowing the user to dynamically switch between two carrier frequencies without CPU intervention.

The following table summarizes the CMT's modes.

Table 43-1. CMT Modes of Operation

Mode	MSC[MCGEN][1]	MSC[BASE][2]	MSC[FSK][2]	MSC[EXSPC]	Comment
Time	1	0	0	0	f_{cg} controlled by primary high and low registers. f_{cg} transmitted to CMT_IRO signal when modulator gate is open.
Baseband	1	1	X	0	f_{cg} is always high. CMT_IRO signal high when modulator gate is open.
FSK	1	0	1	0	f_{cg} control alternates between primary high/low registers and secondary high/low registers. f_{cg} transmitted to CMT_IRO signal when modulator gate is open.
Extended Space	1	X	X	1	Setting MSC[EXSPC] bit causes subsequent modulator cycles to be spaces (modulator out not asserted) for the duration of the modulator period (mark and space times).
IRO Latch	0	X	X	X	OC[IROL] bit controls state of CMT_IRO signal.

[1] To prevent spurious operation, initialize all data and control registers before beginning a transmission (MSC[MCGEN]=1).
[2] These bits are not double buffered and should not be changed during a transmission (while MSC[MCGEN]=1).

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.

43.4.1 Wait Mode Operation

During wait mode, the CMT if enabled, will continue to operate normally . However, there is no change in operating modes of CMT while in wait mode, because the CPU is not operating.

43.4.2 Stop Mode Operation

This section describes the CMT stop mode operations.

43.4.2.1 Normal Stop Mode Operation

During Normal Stop mode, clocks to the CMT module are halted . No registers are affected.

Because the clocks are halted, the CMT will resume upon exit from Normal Stop. Software should ensure that the Normal Stop mode is not entered while the modulator is still in operation to prevent the CMT_IRO signal from being asserted while in Normal Stop mode. This may require a time-out period from the time that MSC[MCGEN] bit is cleared to allow the last modulator cycle to complete.

43.4.2.2 Low Power Stop Mode Operation

During Low Power Stop mode, the CMT module is completely powered off internally and the CMT_IRO signal state at the time that Low Power Stop mode is entered is latched and held. To prevent the CMT_IRO signal from being asserted while in Low Power Stop mode, software should assure that the signal is not active when entering Low Power Stop mode. Upon wake-up from Low Power Stop mode, the CMT module will be in the reset state.

43.5 CMT External Signal Descriptions

This table shows the description of the external signal.

Table 43-2. CMT Signal Descriptions

Signal	Description	I/O
CMT_IRO	Infrared Output	O

43.5.1 CMT_IRO — Infrared Output

This output signal is driven by the modulator output when MSC[MCGEN] is set and OC[IROPEN] is set. The CMT_IRO signal starts a valid transmission with a delay, after MSC[MCGEN] bit be asserted to high, that can be calculated based on two register bits. The following table shows how to calculate this delay.

If MSC[MCGEN] bit is cleared and OC[IROPEN] bit is set, the signal is driven by OC[IROL] bit. This enables user software to directly control the state of the CMT_IRO signal by writing to OC[IROL] bit. If OC[IROPEN] bit is cleared, the signal is disabled and is not driven by the CMT module. Therefore, CMT can be configured as a modulo timer for generating periodic interrupts without causing signal activity.

Table 43-3. CMT_IRO signal delay calculation

Condition	Delay (bus clock cycles)
MSC[CMTDIV] = 0	PPS[PPSDIV] + 2
MSC[CMTDIV] > 0	(PPS{PPSDIV] × 2) + 3

43.6 Memory Map/Register Definition

The following registers control and monitor CMT operation.

The address of a register is the sum of a base address and an address offset. The base address is defined at the chip level. The address offset is defined at the module level.

CMT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_2000	CMT Carrier Generator High Data Register 1 (CMT_CGH1)	8	R/W	Undefined	43.6.1/ 1208
4006_2001	CMT Carrier Generator Low Data Register 1 (CMT_CGL1)	8	R/W	Undefined	43.6.2/ 1209
4006_2002	CMT Carrier Generator High Data Register 2 (CMT_CGH2)	8	R/W	Undefined	43.6.3/ 1210
4006_2003	CMT Carrier Generator Low Data Register 2 (CMT_CGL2)	8	R/W	Undefined	43.6.4/ 1210
4006_2004	CMT Output Control Register (CMT_OC)	8	R/W	00h	43.6.5/ 1211
4006_2005	CMT Modulator Status and Control Register (CMT_MSC)	8	R/W	00h	43.6.6/ 1212
4006_2006	CMT Modulator Data Register Mark High (CMT_CMD1)	8	R/W	Undefined	43.6.7/ 1214
4006_2007	CMT Modulator Data Register Mark Low (CMT_CMD2)	8	R/W	Undefined	43.6.8/ 1214
4006_2008	CMT Modulator Data Register Space High (CMT_CMD3)	8	R/W	Undefined	43.6.9/ 1215
4006_2009	CMT Modulator Data Register Space Low (CMT_CMD4)	8	R/W	Undefined	43.6.10/ 1215
4006_200A	CMT Primary Prescaler Register (CMT_PPS)	8	R/W	00h	43.6.11/ 1216
4006_200B	CMT Direct Memory Access (CMT_DMA)	8	R/W	00h	43.6.12/ 1217

43.6.1 CMT Carrier Generator High Data Register 1 (CMT_CGH1)

This data register contain the primary high value for generating the carrier output.

Address: CMT_CGH1 is 4006_2000h base + 0h offset = 4006_2000h

Bit	7	6	5	4	3	2	1	0
Read Write	PH							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

CMT_CGH1 field descriptions

Field	Description
7–0 PH	Primary Carrier High Time Data Value When selected, these bits contain the number of input clocks required to generate the carrier high time period. When operating in Time mode, this register is always selected. When operating in FSK mode, this register and the secondary register pair are alternately selected under control of the modulator. The primary carrier high time value is undefined out of reset. These bits must be written to non-zero values before the carrier generator is enabled to avoid spurious results.

43.6.2 CMT Carrier Generator Low Data Register 1 (CMT_CGL1)

This data register contain the primary low value for generating the carrier output.

Address: CMT_CGL1 is 4006_2000h base + 1h offset = 4006_2001h

Bit	7	6	5	4	3	2	1	0
Read Write	PL							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

CMT_CGL1 field descriptions

Field	Description
7–0 PL	Primary Carrier Low Time Data Value When selected, these bits contain the number of input clocks required to generate the carrier low time period. When operating in Time mode, this register is always selected. When operating in FSK mode, this register and the secondary register pair are alternately selected under control of the modulator. The primary carrier low time value is undefined out of reset. These bits must be written to non-zero values before the carrier generator is enabled to avoid spurious results.

43.6.3 CMT Carrier Generator High Data Register 2 (CMT_CGH2)

This data register contain the secondary high value for generating the carrier output.

Address: CMT_CGH2 is 4006_2000h base + 2h offset = 4006_2002h

Bit	7	6	5	4	3	2	1	0
Read Write	SH							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CMT_CGH2 field descriptions

Field	Description
7–0 SH	Secondary Carrier High Time Data Value When selected, these bits contain the number of input clocks required to generate the carrier high time period. When operating in Time mode, this register is never selected. When operating in FSK mode, this register and the primary register pair are alternately selected under control of the modulator. The secondary carrier high time value is undefined out of reset. These bits must be written to nonzero values before the carrier generator is enabled when operating in FSK mode.

43.6.4 CMT Carrier Generator Low Data Register 2 (CMT_CGL2)

This data register contain the secondary low value for generating the carrier output.

Address: CMT_CGL2 is 4006_2000h base + 3h offset = 4006_2003h

Bit	7	6	5	4	3	2	1	0
Read Write	SL							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CMT_CGL2 field descriptions

Field	Description
7–0 SL	Secondary Carrier Low Time Data Value When selected, these bits contain the number of input clocks required to generate the carrier low time period. When operating in Time mode, this register is never selected. When operating in FSK mode, this register and the primary register pair are alternately selected under control of the modulator. The secondary carrier low time value is undefined out of reset. These bits must be written to nonzero values before the carrier generator is enabled when operating in FSK mode.

43.6.5 CMT Output Control Register (CMT_OC)

This register is used to control the IRO signal of the CMT module.

Address: CMT_OC is 4006_2000h base + 4h offset = 4006_2004h

Bit	7	6	5	4	3	2	1	0
Read	IROL	CMTPOL	IROPEN	0				
Write								
Reset	0	0	0	0	0	0	0	0

CMT_OC field descriptions

Field	Description
7 IROL	IRO Latch Control Reading IROL reads the state of the IRO latch. Writing to IROL changes the state of the CMT_IRO signal when MSC[MCGEN] bit is cleared and the IROPEN bit is set.
6 CMTPOL	CMT Output Polarity The CMTPOL bit controls the polarity of the CMT_IRO signal of the CMT. 0 CMT_IRO signal is active low 1 CMT_IRO signal is active high
5 IROPEN	IRO Pin Enable The IROPEN bit is used to enable and disable the CMT_IRO signal. When CMT_IRO signal is enabled, it is an output that drives out either the CMT transmitter output or the state of the IROL bit depending on whether MSC[MCGEN] bit is set or not. Also, the state of the output is either inverted or not depending on the state of the CMTPOL bit. When CMT_IRO signal is disabled, it is in a high impedance state so as not to draw any current. This signal is disabled during reset. 0 CMT_IRO signal disabled 1 CMT_IRO signal enabled as output
4–0 Reserved	This read-only field is reserved and always has the value zero.

43.6.6 CMT Modulator Status and Control Register (CMT_MSC)

The MSC register contains the modulator and carrier generator enable (MCGEN), end of cycle interrupt enable (EOCIE), FSK mode select (FSK), baseband enable (BASE), extended space (EXSPC), prescaler (CMTDIV) bits, and the end of cycle (EOCF) status bit.

Address: CMT_MSC is 4006_2000h base + 5h offset = 4006_2005h

Bit	7	6	5	4	3	2	1	0
Read	EOCF	CMTDIV		EXSPC	BASE	FSK	EOCIE	MCGEN
Write								
Reset	0	0	0	0	0	0	0	0

CMT_MSC field descriptions

Field	Description
7 EOCF	End Of Cycle Status Flag The EOCF bit is set when: • The modulator is not currently active and the MCGEN bit is set to begin the initial CMT transmission. • At the end of each modulation cycle while the MCGEN bit is set. This is recognized when a match occurs between the contents of the space period register and the down counter. At this time, the counter is initialized with the (possibly new) contents of the mark period buffer, CMT_CMD1 and CMT_CMD2, and the space period register is loaded with the (possibly new) contents of the space period buffer, CMT_CMD3 and CMT_CMD4. This flag is cleared by a read of the MSC register followed by an access of CMD2 or CMD4 or by the DMA transfer. 0 No end of modulation cycle occurrence since flag last cleared 1 End of modulator cycle has occurred
6–5 CMTDIV	CMT Clock Divide Prescaler The Secondary Prescaler causes the CMT to be clocked at the IF signal frequency, or the IF frequency divided by 2 ,4, or 8. Since these bits are not double buffered, they should not be changed during a transmission. 00 IF ÷ 1 01 IF ÷ 2 10 IF ÷ 4 11 IF ÷ 8
4 EXSPC	Extended Space Enable The EXSPC bit enables extended space operation.

Table continues on the next page...

CMT_MSC field descriptions (continued)

Field	Description
	0 Extended space disabled 1 Extended space enabled
3 BASE	Baseband Enable When set, the BASE bit disables the carrier generator and forces the carrier output high for generation of baseband protocols. When BASE is cleared, the carrier generator is enabled and the carrier output toggles at the frequency determined by values stored in the carrier data registers. This bit is cleared by reset. This bit is not double buffered and should not be written to during a transmission. 0 Baseband mode disabled 1 Baseband mode enabled
2 FSK	FSK Mode Select The FSK bit enables FSK operation. 0 CMT operates in Time or Baseband mode 1 CMT operates in FSK mode
1 EOCIE	End of Cycle Interrupt Enable A CPU interrupt will be requested when EOCF is set if EOCIE is high. 0 CPU interrupt disabled 1 CPU interrupt enabled
0 MCGEN	Modulator and Carrier Generator Enable Setting MCGEN will initialize the carrier generator and modulator and will enable all clocks. Once enabled, the carrier generator and modulator will function continuously. When MCGEN is cleared, the current modulator cycle will be allowed to expire before all carrier and modulator clocks are disabled (to save power) and the modulator output is forced low. To prevent spurious operation, the user should initialize all data and control registers before enabling the system. 0 Modulator and carrier generator disabled 1 Modulator and carrier generator enabled

43.6.7 CMT Modulator Data Register Mark High (CMT_CMD1)

The contents of this register are transferred to the modulator down counter upon the completion of a modulation period.

Address: CMT_CMD1 is 4006_2000h base + 6h offset = 4006_2006h

Bit	7	6	5	4	3	2	1	0
Read Write	MB[15:8]							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

CMT_CMD1 field descriptions

Field	Description
7–0 MB[15:8]	These bits control the upper mark periods of the modulator for all modes.

43.6.8 CMT Modulator Data Register Mark Low (CMT_CMD2)

The contents of this register are transferred to the modulator down counter upon the completion of a modulation period.

Address: CMT_CMD2 is 4006_2000h base + 7h offset = 4006_2007h

Bit	7	6	5	4	3	2	1	0
Read Write	MB[7:0]							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:

- x = Undefined at reset.

CMT_CMD2 field descriptions

Field	Description
7–0 MB[7:0]	These bits control the lower mark periods of the modulator for all modes.

43.6.9 CMT Modulator Data Register Space High (CMT_CMD3)

The contents of this register are transferred to the space period register upon the completion of a modulation period.

Address: CMT_CMD3 is 4006_2000h base + 8h offset = 4006_2008h

Bit	7	6	5	4	3	2	1	0
Read Write	SB[15:8]							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CMT_CMD3 field descriptions

Field	Description
7–0 SB[15:8]	These bits control the upper space periods of the modulator for all modes.

43.6.10 CMT Modulator Data Register Space Low (CMT_CMD4)

The contents of this register are transferred to the space period register upon the completion of a modulation period.

Address: CMT_CMD4 is 4006_2000h base + 9h offset = 4006_2009h

Bit	7	6	5	4	3	2	1	0
Read Write	SB[7:0]							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

CMT_CMD4 field descriptions

Field	Description
7–0 SB[7:0]	These bits control the lower space periods of the modulator for all modes.

43.6.11 CMT Primary Prescaler Register (CMT_PPS)

This register is used to set the primary prescaler bits (PPSDIV).

Address: CMT_PPS is 4006_2000h base + Ah offset = 4006_200Ah

Bit	7	6	5	4	3	2	1	0
Read	0				PPSDIV			
Write	▨	▨	▨	▨				
Reset	0	0	0	0	0	0	0	0

CMT_PPS field descriptions

Field	Description
7–4 Reserved	This read-only field is reserved and always has the value zero.
3–0 PPSDIV	Primary Prescaler Divider The primary prescaler divides the CMT clock to generate the Intermediate Frequency clock enable to the secondary prescaler. 0000 Bus Clock ÷ 1 0001 Bus Clock ÷ 2 0010 Bus Clock ÷ 3 0011 Bus Clock ÷ 4 0100 Bus Clock ÷ 5 0101 Bus Clock ÷ 6 0110 Bus Clock ÷ 7 0111 Bus Clock ÷ 8 1000 Bus Clock ÷ 9 1001 Bus Clock ÷ 10 1010 Bus Clock ÷ 11 1011 Bus Clock ÷ 12 1100 Bus Clock ÷ 13 1101 Bus Clock ÷ 14 1110 Bus Clock ÷ 15 1111 Bus Clock ÷ 16

43.6.12 CMT Direct Memory Access (CMT_DMA)

This register is used to enable/disable direct memory access (DMA).

Address: CMT_DMA is 4006_2000h base + Bh offset = 4006_200Bh

Bit	7	6	5	4	3	2	1	0
Read	0							DMA
Write								
Reset	0	0	0	0	0	0	0	0

CMT_DMA field descriptions

Field	Description
7–1 Reserved	This read-only field is reserved and always has the value zero.
0 DMA	DMA Enable This bit enables the DMA protocol. 0 DMA transfer request and done are disabled 1 DMA transfer request and done are enabled

43.7 Functional Description

The CMT module consists primarily of clock divider, carrier generator and modulator.

43.7.1 Clock Divider

The CMT was originally designed to be based on 8 MHz bus clock that could be divided by 1, 2, 4 or 8 times accordingly with the specification. To be compatible with higher bus frequency, the Primary Prescaler (PPS) was developed to receive a higher frequency and generate a clock enable signal called Intermediate Frequency (IF). This IF should be approximately equal to 8 MHz and will work as a clock enable to the Secondary Prescaler. The following figure shows the clock divider block diagram.

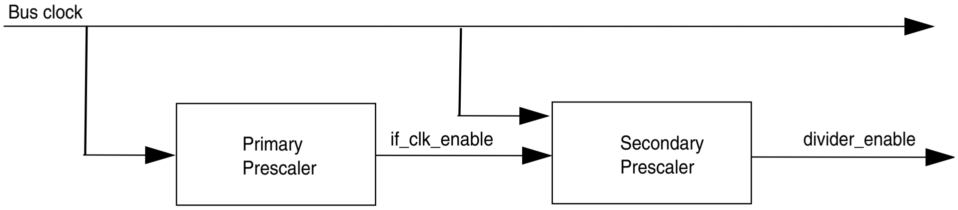


Figure 43-14. Clock Divider Block Diagram

For compatibility with previous versions of CMT, when bus clock = 8 MHz, the PPS should be configured to zero. The PPS counter is selected according to the bus clock to generate an intermediate frequency approximately equal to 8 MHz.

43.7.2 Carrier Generator

The carrier generator resolution is 125 ns when operating with an 8 MHz intermediate frequency signal and the Secondary Prescaler is set to divide by 1 (MSC[CMTDIV] = 00). The carrier generator can generate signals with periods between 250 ns (4 MHz) and 127.5 µs (7.84 kHz) in steps of 125 ns. The following table shows the relationship between the clock divide bits and the carrier generator resolution, minimum carrier generator period, and minimum modulator period.

Table 43-17. Clock Divider

Bus Clock (MHz)	MSC[CMTDIV]	Carrier Generator Resolution (µs)	Min. Carrier Generator Period (µs)	Min. Modulator Period (µs)
8	00	0.125	0.25	1.0
8	01	0.25	0.5	2.0
8	10	0.5	1.0	4.0
8	11	1.0	2.0	8.0

The possible duty cycle options depend upon the number of counts required to complete the carrier period. For example, 1.6 MHz signal has a period of 625 ns and will therefore require 5 x 125 ns counts to generate. These counts may be split between high and low times, so the duty cycles available will be 20% (one high, four low), 40% (two high, three low), 60% (three high, two low) and 80% (four high, one low).

For lower frequency signals with larger periods, higher resolution (as a percentage of the total period) duty cycles are possible.

The carrier signal is generated by counting a register-selected number of input clocks (125 ns for an 8 MHz bus) for both the carrier high time and the carrier low time. The period is determined by the total number of clocks counted. The duty cycle is determined by the ratio of high time clocks to total clocks counted. The high and low time values are user programmable and are held in two registers.

An alternate set of high/low count values is held in another set of registers to allow the generation of dual frequency FSK (frequency shift keying) protocols without CPU intervention.

Note

> Only non-zero data values are allowed. The carrier generator will not work if any of the count values are equal to zero.

MSC[MCGEN] bit must be set and MSC[BASE] bit must be cleared to enable carrier generator clocks. When MSC[BASE] bit is set, the carrier output to the modulator is held high continuously. Following figure represents the block diagram of the clock generator.

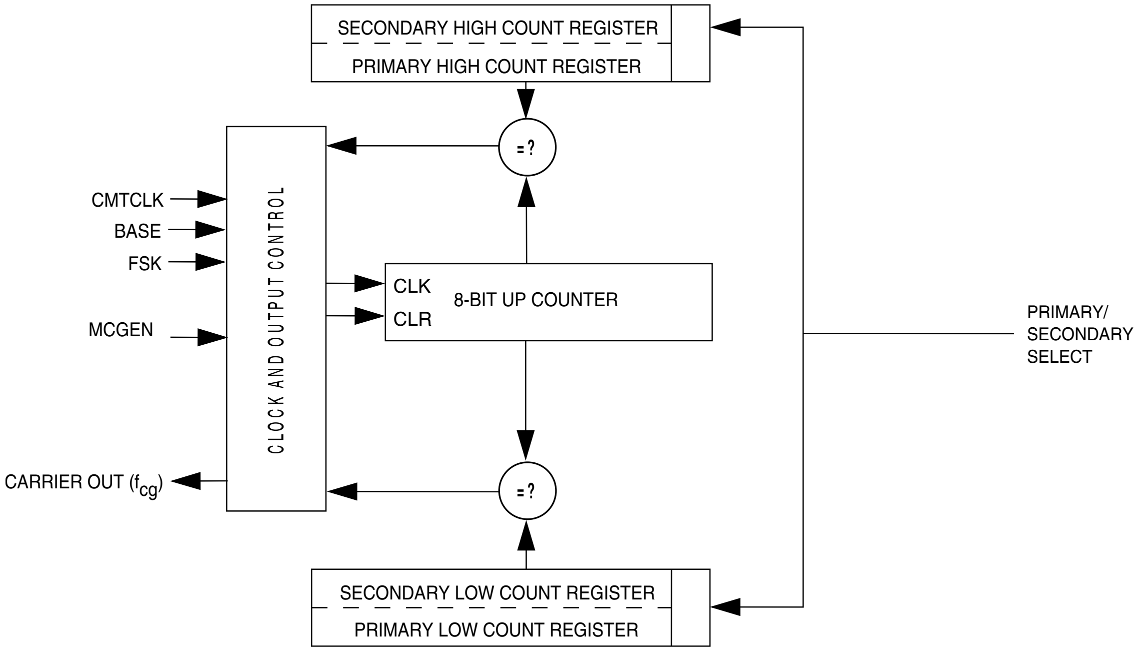


Figure 43-15. Carrier Generator Block Diagram

The high/low time counter is an 8-bit up counter. After each increment, the contents of the counter are compared with the appropriate high or low count value register. When the compare value is reached, the counter is reset to a value of 0x01, and the compare is redirected to the other count value register.

Assuming that the high time count compare register is currently active, a valid compare will cause the carrier output to be driven low. The counter will continue to increment (starting at reset value of 0x01). When the value stored in the selected low count value register is reached, the counter will again be reset and the carrier output will be driven high.

The cycle repeats, automatically generating a periodic signal which is directed to the modulator . The lowest frequency (maximum period) and highest frequency (minimum period) which can be generated are defined as:

$f_{max} = f_{CMTCLK} \div (2 \text{ x } 1)$ Hz

$f_{min} = f_{CMTCLK} \div (2 \text{ x } (2^8 - 1))$ Hz

In the general case, the carrier generator output frequency is:

$f_{cg} = f_{CMTCLK} \div (\text{Highcount} + \text{Lowcount})$ Hz

Where: $0 < \text{Highcount} < 256$ and

$0 < \text{Lowcount} < 256$

The duty cycle of the carrier signal is controlled by varying the ratio of high time to low + high time. As the input clock period is fixed, the duty cycle resolution will be proportional to the number of counts required to generate the desired carrier period.

$$\text{DutyCycle} = \frac{\text{Highcount}}{\text{Highcount+Lowcount}}$$

43.7.3 Modulator

The modulator block controls the state of the infrared out signal (IRO) . The modulator output is gated on to the IRO signal when the modulator/carrier generator is enabled . When the modulator/carrier generator is disabled, the IRO signal is controlled by the state of the IRO latch. OC[CMTPOL] enables the IRO signal to be active high or active low.

In CMT modes, the modulator functions as givenbelow:

- In Time mode, the modulator can gate the carrier onto the modulator output.
- In Baseband mode, the modulator can control the logic level of the modulator output.
- In FSK mode, the modulator can count carrier periods and instruct the carrier generator to alternate between two carrier frequencies whenever a modulation period (mark + space counts) expires.

The modulator provides a simple method to control protocol timing. The modulator has a minimum resolution of 1.0 μs with an 8 MHz . It can count bus clocks (to provide real-time control) or it can count carrier clocks (for self-clocked protocols).

The modulator includes a 17-bit down counter with underflow detection. The counter is loaded from the 16-bit modulation mark period buffer registers, CMD1 and CMD2. The most significant bit is loaded with a logic zero and serves as a sign bit. When the counter holds a positive value, the modulator gate is open and the carrier signal is driven to the transmitter block.

When the counter underflows, the modulator gate is closed and a 16-bit comparator is enabled which compares the logical complement of the value of the down counter with the contents of the modulation space period register which has been loaded from the registers, CMD3 and CMD4.

When a match is obtained, the cycle repeats by opening the modulator gate, reloading the counter with the contents of CMD1 and CMD2, and reloading the modulation space period register with the contents of CMD3 and CMD4.

The activation of modulation space period is done when the carrier signal is low to prohibit cutting off the high pulse of a carrier signal. If the carrier signal is high, the modulator extends the mark period until the carrier signal become low. To de-assert the space period and assert the mark period, the carrier signal must have gone low to assure that a space period is not erroneously shortened.

Should the contents of the modulation space period register be all zeroes, the match will be immediate and no space period will be generated (for instance, for FSK protocols that require successive bursts of different frequencies).

MSC[MCGEN] must be set to enable the modulator timer.

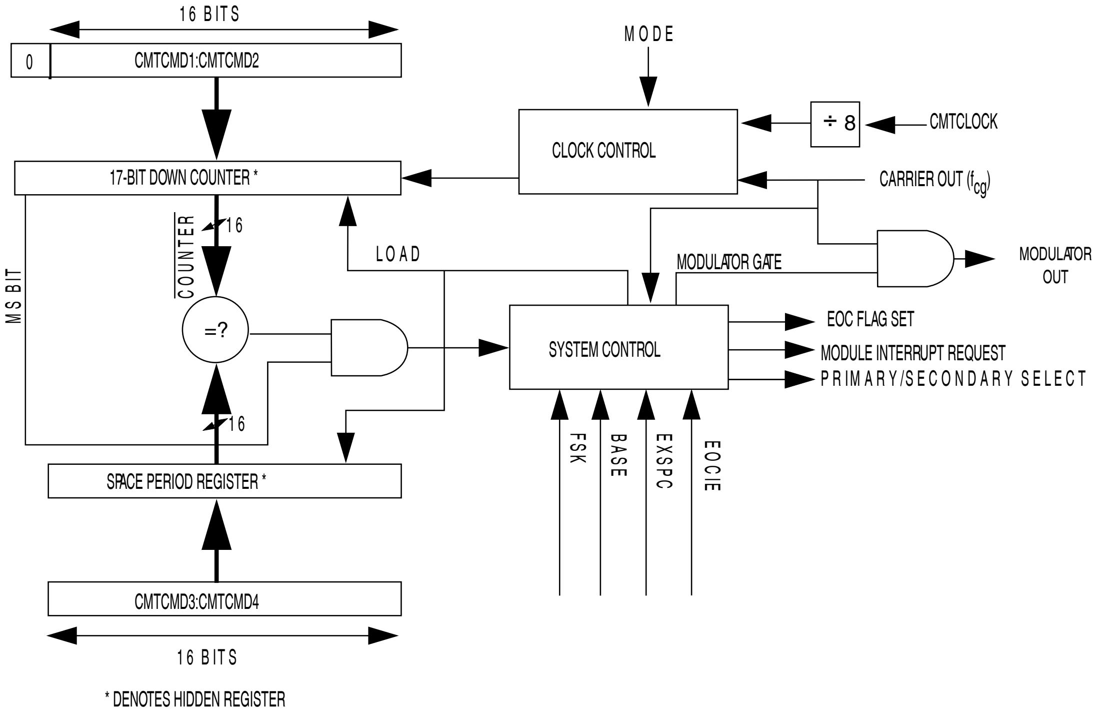


Figure 43-16. Modulator Block Diagram

43.7.3.1 Time Mode

When the modulator operates in time mode (MSC[MCGEN] bit is set, MSC[BASE] and MSC[FSK] bits are cleared), the modulation mark period consists of an integer number of CMTCLK ÷ 8 clock periods. The modulation space period consists of zero or an integer number of CMTCLK ÷ 8 clock periods. With an 8 MHz IF and MSC[CMTDIV] = 00, the modulator resolution is 1 μs and has a maximum mark and space period of about 65.535 ms each . See the following figure for an example of the time mode and baseband mode outputs.

The mark and space time equations for time and baseband mode are:

$t_{mark} = (CMD1:CMD2 + 1) \div (f_{CMTCLK} \div 8)$

$t_{space} = CMD3:CMD4 \div (f_{CMTCLK} \div 8)$

where CMD1:CMD2 and CMD3:CMD4 are the decimal values of the concatenated registers.

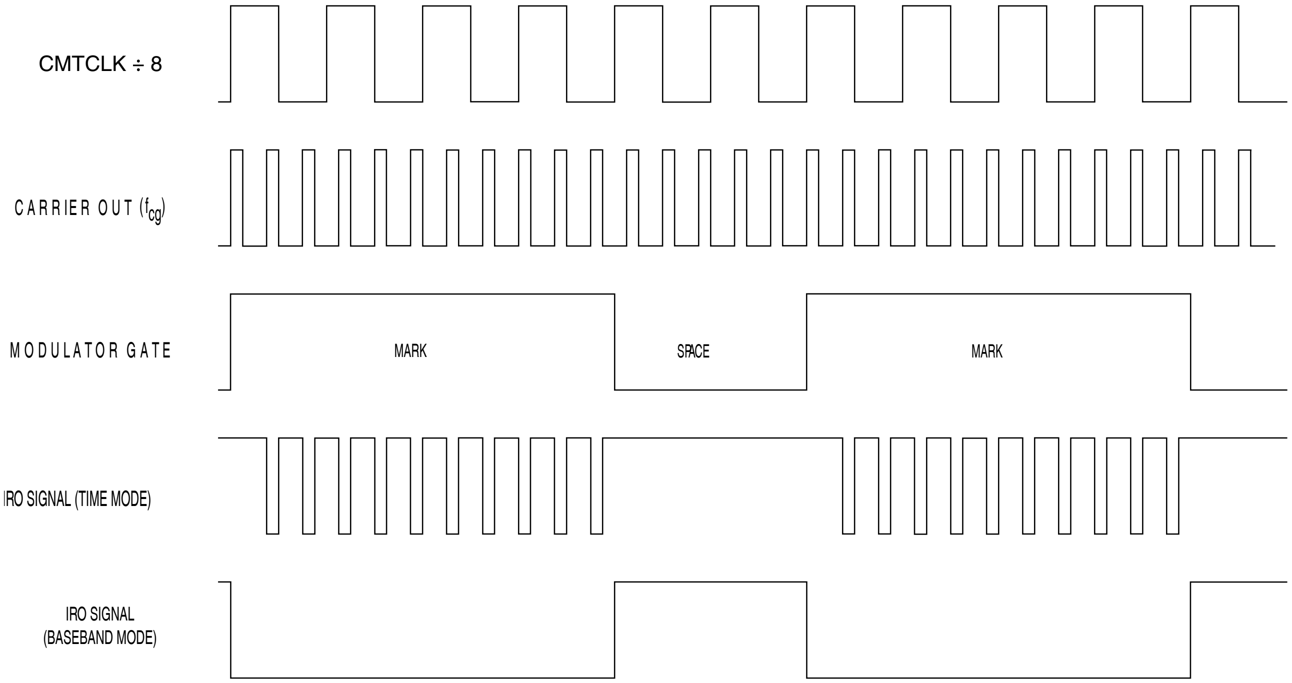


Figure 43-17. Example: CMT Output in Time and Baseband Modes with OC[CMTPOL]=0

43.7.3.2 Baseband Mode

Baseband mode (MSC[MCGEN] and MSC[BASE] bits are set) is a derivative of time mode, where the mark and space period is based on (CMTCLK ÷ 8) counts. The mark and space calculations are the same as in time mode. In this mode, the modulator output will be at a logic 1 for the duration of the mark period and at a logic 0 for the duration of a space period. See Figure 43-17 for an example of the output for both baseband and time modes. In the example, the carrier out frequency (f_{cg}) is generated with a high count of 0x01 and a low count of 0x02 that results in a divide of 3 of CMTCLK with a 33% duty cycle. The modulator down counter was loaded with the value 0x0003 and the space period register with 0x0002.

Note

The waveforms in Figure 43-17 and Figure 43-18 are for the purpose of conceptual illustration and are not meant to represent precise timing relationships between the signals shown.

43.7.3.3 FSK Mode

When the modulator operates in FSK mode (MSC[MCGEN] and MSC[FSK] bits are set, and MSC[BASE] bit is cleared), the modulation mark and space periods consist of an integer number of carrier clocks (space period can be zero). When the mark period expires, the space period is transparently started (as in time mode). The carrier generator toggles between primary and secondary data register values whenever the modulator space period expires.

The space period provides an interpulse gap (no carrier). If CMD3:CMD4 = 0x0000, then the modulator and carrier generator will switch between carrier frequencies without a gap or any carrier glitches (zero space).

Using timing data for carrier burst and interpulse gap length calculated by the CPU, FSK mode can automatically generate a phase-coherent, dual-frequency FSK signal with programmable burst and interburst gaps.

The mark and space time equations for FSK mode are:

$t_{mark} = (\text{CMD1:CMD2} + 1) \div f_{cg}$

$t_{space} = \text{CMD3:CMD4} \div f_{cg}$

Where f_{cg} is the frequency output from the carrier generator. The example in figure below shows what the IRO signal looks like in FSK mode with the following values: CMD1:CMD2 = 0x0003, CMD3:CMD4 = 0x0002, primary carrier high count = 0x01, primary carrier low count = 0x02, secondary carrier high count = 0x03, and secondary carrier low count = 0x01.

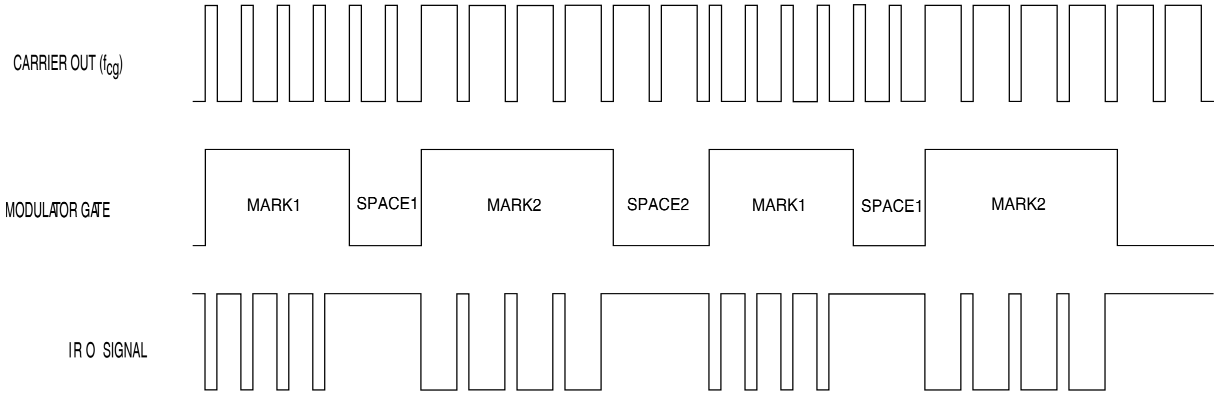


Figure 43-18. Example: CMT Output in FSK Mode

43.7.4 Extended Space Operation

In either time, baseband or FSK mode, the space period can be made longer than the maximum possible value of the space period register . Setting MSC[EXSPC] bit will force the modulator to treat the next modulation period (beginning with the next load of the counter and space period register) as a space period equal in length to the mark and space counts combined . Subsequent modulation periods will consist entirely of these extended space periods with no mark periods . Clearing MSC[EXSPC] will return the modulator to standard operation at the beginning of the next modulation period .

43.7.4.1 EXSPC Operation in Time Mode

To calculate the length of an extended space in time or baseband mode, add the mark and space times and multiply by the number of modulation periods when MSC[EXSPC] is set.

$t_{exspace} = (t_{mark} + t_{space})$ x (number of modulation periods)

For an example of extended space operation, see the following figure.

Note

> The extended space enable feature can be used to emulate a zero mark event.

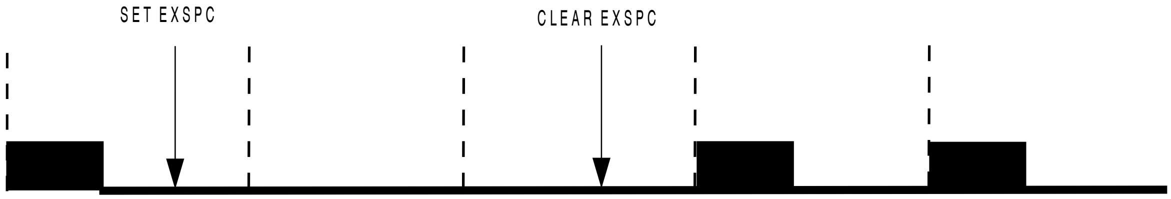


Figure 43-19. Extended Space Operation

43.7.4.2 EXSPC Operation in FSK Mode

In FSK mode, the modulator continues to count carrier out clocks, alternating between the primary and secondary registers at the end of each modulation period.

To calculate the length of an extended space in FSK mode, one needs to know whether MSC[EXSPC] bit was set on a primary or secondary modulation period, as well as the total number of both primary and secondary modulation periods completed while MSC[EXSPC] bit is high. A status bit for the current modulation is not accessible to the

CPU. If necessary, software should maintain tracking of the current modulation cycle (primary or secondary). The extended space period ends at the completion of the space period time of the modulation period during which MSC[EXSPC] bit is cleared.

If MSC[EXSPC] bit was set during a primary modulation cycle, use the equation:

$t_{exspace} = (t_{space})_p + (t_{mark} + t_{space})_s + (t_{mark} + t_{space})_p + ...$

Where the subscripts p and s refer to mark and space times for the primary and secondary modulation cycles.

If MSC[EXSPC] bit was set during a secondary modulation cycle, use the equation:

$t_{exspace} = (t_{space})_s + (t_{mark} + t_{space})_p + (t_{mark} + t_{space})_s + ...$

43.8 CMT Interrupts and DMA

The CMT generates an Interrupt request or a DMA transfer request according to MSC[EOCIE], MSC[EOCF], DMA[DMA] bits.

Table 43-18. DMA Transfer Request x CMT Interrupt Request

MSC[EOCF]	DMA[DMA]	MSC[EOCIE]	DMA transfer request	CMT interrupt request
0	X	X	0	0
1	X	0	0	0
1	0	1	0	1
1	1	1	1	0

MSC[EOCF] is set when:

- The modulator is not currently active and MSC[MCGEN] bit is set to begin the initial CMT transmission
- At the end of each modulation cycle (when the counter is reloaded from CMD1:CMD2) while MSC[MCGEN] bit is set

In the case where MSC[MCGEN] bit is cleared and then set before the end of the modulation cycle, MSC[EOCF] bit will not be set when MSC[MCGEN] is set, but will become set at the end of the current modulation cycle.

When MSC[MCGEN] becomes disabled, the CMT module does not set the EOC flag at the end of the last modulation cycle.

If MSC[EOCIE] bit is high when MSC[EOCF] bit is set, the CMT module will generate an interrupt request or a DMA transfer request.

MSC[EOCF] bit must be cleared to prevent from being generated another event (interrupt or DMA request) after exiting the service routine. See following table.

Table 43-19. How to clear MSC[EOCF] bit

DMA[DMA]	MSC[EOCIE]	Description
0	X	MSC[EOCF] bit is cleared by reading the CMT modulator status and control register MSC followed by an access of CMD2 or CMD4.
1	X	MSC[EOCF] bit is cleared by the CMT DMA transfer done.

The EOC interrupt is coincident with loading the down-counter with the contents of CMD1:CMD2 and loading the space period register with the contents of CMD3:CMD4. The EOC interrupt provides a means for the user to reload new mark/space values into the modulator data registers. Modulator data register updates will take effect at the end of the current modulation cycle. Note that the down-counter and space period register are updated at the end of every modulation cycle, irrespective of interrupt handling and the state of the EOCF flag.

Chapter 44
Real Time Clock (RTC)

44.1 Introduction

NOTE

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

44.1.1 Features

The RTC module features include:

- Independent power supply, POR and 32 kHz crystal oscillator
- 32-bit seconds counter with roll-over protection and 32-bit alarm
- 16-bit prescaler with compensation that can correct errors between 0.12 ppm and 3906 ppm
- Register write protection
  - Lock register requires VBAT POR or software reset to enable write access
  - Access control registers require system reset to enable read and/or write access
- 1 Hz square wave output
- 64-bit monotonic counter with roll-over protection
- Tamper time seconds register that records when the time was invalidated
- Up to 6 tamper sources plus software initiated tamper for invalidating counters:
  - Voltage, temperature and clock out-of-range analog tamper detectors
  - Flash security disable, test mode entry

44.1.2 Modes of operation

The RTC operates in one of two modes of operation, chip power-up and chip power-down.

During chip power-down, RTC is powered from the backup power supply (VBAT) and is electrically isolated from the rest of the chip but continues to increment the time counter (if enabled) and retain the state of the RTC registers. The RTC registers are not accessible.

During chip power-up, RTC remains powered from the backup power supply (VBAT). All RTC registers are accessible by software and all functions are operational. If enabled, the 32.768 kHz clock can be supplied to the rest of the chip.

44.1.3 RTC signal descriptions

Table 44-1. RTC signal descriptions

Signal	Description	I/O
EXTAL32	32.768 kHz oscillator input	I
XTAL32	32.768 kHz oscillator output	O
RTC_CLKOUT	1Hz square-wave output	O
$\overline{\text{RTC\_WAKEUP}}$	Wakeup for external device	I/O

44.1.3.1 RTC clock output

The clock to the seconds counter is available on the RTC_CLKOUT signal. It is a 1Hz square wave output.

44.1.3.2 RTC wakeup pin

The RTC wakeup pin is an open drain, active low, output that allows the RTC to wakeup the chip via an external component. The wakeup pin asserts when the wakeup pin enable is set, the RTC interrupt is asserted and the chip is powered down. The wakeup pin does not assert from the RTC seconds interrupt.

The wakeup pin is optional and may not be implemented on all devices.

44.2 Register definition

All registers must be accessed using 32-bit writes and all register accesses incur three wait states.

Write accesses to any register and read accesses to tamper and monotonic registers by non-supervisor mode software, when the supervisor access bit in the control register is clear, will terminate with a bus error.

Read accesses to other registers by non-supervisor mode software complete as normal.

Writing to a register protected by the write access register or lock register does not generate a bus error, but the write will not complete.

Reading a register protected by the read access register does not generate a bus error, but the register will read zero.

RTC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_D000	RTC Time Seconds Register (RTC_TSR)	32	R/W	0000_0000h	44.2.1/ 1232
4003_D004	RTC Time Prescaler Register (RTC_TPR)	32	R/W	0000_0000h	44.2.2/ 1233
4003_D008	RTC Time Alarm Register (RTC_TAR)	32	R/W	0000_0000h	44.2.3/ 1233
4003_D00C	RTC Time Compensation Register (RTC_TCR)	32	R/W	0000_0000h	44.2.4/ 1234
4003_D010	RTC Control Register (RTC_CR)	32	R/W	0000_0000h	44.2.5/ 1235
4003_D014	RTC Status Register (RTC_SR)	32	R/W	0000_0001h	44.2.6/ 1237
4003_D018	RTC Lock Register (RTC_LR)	32	R/W	0000_FFFFh	44.2.7/ 1238
4003_D01C	RTC Interrupt Enable Register (RTC_IER)	32	R/W	0000_0007h	44.2.8/ 1240
4003_D020	RTC Tamper Time Seconds Register (RTC_TTSR)	32	R	See section	44.2.9/ 1241
4003_D024	RTC Monotonic Enable Register (RTC_MER)	32	R/W	0000_0000h	44.2.10/ 1241

Table continues on the next page...

RTC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_D028	RTC Monotonic Counter Low Register (RTC_MCLR)	32	R/W	0000_0000h	44.2.11/ 1242
4003_D02C	RTC Monotonic Counter High Register (RTC_MCHR)	32	R/W	0000_0000h	44.2.12/ 1242
4003_D030	RTC Tamper Enable Register (RTC_TER)	32	R/W	0000_0000h	44.2.13/ 1243
4003_D034	RTC Tamper Detect Register (RTC_TDR)	32	R/W	See section	44.2.14/ 1244
4003_D038	RTC Tamper Trim Register (RTC_TTR)	32	R/W	0000_0000h	44.2.15/ 1245
4003_D03C	RTC Tamper Interrupt Register (RTC_TIR)	32	R/W	0000_0000h	44.2.16/ 1246
4003_D800	RTC Write Access Register (RTC_WAR)	32	R/W	0000_FFFFh	44.2.17/ 1247
4003_D804	RTC Read Access Register (RTC_RAR)	32	R/W	0000_FFFFh	44.2.18/ 1249

44.2.1 RTC Time Seconds Register (RTC_TSR)

Address: RTC_TSR is 4003_D000h base + 0h offset = 4003_D000h

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	TSR																															
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

RTC_TSR field descriptions

Field	Description
31–0 TSR	Time Seconds Register When the time counter is enabled, the TSR is read only and increments once a second provided SR[TOF] or SR[TIF] are not set. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TSR can be read or written. Writing to the TSR when the time counter is disabled will clear the SR[TOF] and/or the SR[TIF]. Writing to the TSR register with zero is supported, but not recommended since TSR will read as zero when SR[TIF] or SR[TOF] are set (indicating the time is invalid).

44.2.2 RTC Time Prescaler Register (RTC_TPR)

Address: RTC_TPR is 4003_D000h base + 4h offset = 4003_D004h

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																TPR															
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

RTC_TPR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 TPR	Time Prescaler Register When the time counter is enabled, the TPR is read only and increments every 32.768 kHz clock cycle. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TPR can be read or written. The TSR[TSR] increments when bit 14 of the TPR transitions from a logic one to a logic zero.

44.2.3 RTC Time Alarm Register (RTC_TAR)

Address: RTC_TAR is 4003_D000h base + 8h offset = 4003_D008h

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	TAR																															
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

RTC_TAR field descriptions

Field	Description
31–0 TAR	Time Alarm Register When the time counter is enabled, the SR[TAF] is set whenever the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. Writing to the TAR clears the SR[TAF].

44.2.4 RTC Time Compensation Register (RTC_TCR)

Address: RTC_TCR is 4003_D000h base + Ch offset = 4003_D00Ch

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	CIC								TCV								CIR								TCR							
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

RTC_TCR field descriptions

Field	Description
31–24 CIC	Compensation Interval Counter Current value of the compensation interval counter. If the compensation interval counter equals zero then it is loaded with the contents of the CIR. If the CIC does not equal zero then it is decremented once a second.
23–16 TCV	Time Compensation Value Current value used by the compensation logic for the present second interval. Updated once a second if the CIC equals 0 with the contents of the TCR field. If the CIC does not equal zero then it is loaded with zero (compensation is not enabled for that second increment).
15–8 CIR	Compensation Interval Register Configures the compensation interval in seconds from 1 to 256 to control how frequently the TCR should adjust the number of 32.768 kHz cycles in each second. The value written should be one less than the number of seconds (for example, write zero to configure for a compensation interval of one second). This register is double buffered and writes do not take affect until the end of the current compensation interval.
7–0 TCR	Time Compensation Register Configures the number of 32.768 kHz clock cycles in each second. This register is double buffered and writes do not take affect until the end of the current compensation interval. 80h Time prescaler register overflows every 32896 clock cycles. ... ... FFh Time prescaler register overflows every 32769 clock cycles. 00h Time prescaler register overflows every 32768 clock cycles. 01h Time prescaler register overflows every 32767 clock cycles. ... ... 7Fh Time prescaler register overflows every 32641 clock cycles.

44.2.5 RTC Control Register (RTC_CR)

Address: RTC_CR is 4003_D000h base + 10h offset = 4003_D010h

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	Reserved	SC2P	SC4P	SC8P	SC16P	CLKO	OSCE	0				UM	SUP	WPE	SWR
W		0														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RTC_CR field descriptions

Field	Description
31–15 Reserved	This read-only field is reserved and always has the value zero.
14 Reserved	This field is reserved. It must always be written to 0.
13 SC2P	Oscillator 2pF load configure 0 Disable the load. 1 Enable the additional load.
12 SC4P	Oscillator 4pF load configure 0 Disable the load. 1 Enable the additional load.
11 SC8P	Oscillator 8pF load configure 0 Disable the load. 1 Enable the additional load.
10 SC16P	Oscillator 16pF load configure 0 Disable the load. 1 Enable the additional load.

Table continues on the next page...

RTC_CR field descriptions (continued)

Field	Description
9 CLKO	Clock Output 0 The 32kHz clock is output to other peripherals 1 The 32kHz clock is not output to other peripherals
8 OSCE	Oscillator Enable 0 32.768 kHz oscillator is disabled. 1 32.768 kHz oscillator is enabled. After setting this bit, wait the oscillator startup time before enabling the time counter to allow the 32.768 kHz clock time to stabilize.
7–4 Reserved	This read-only field is reserved and always has the value zero.
3 UM	Update Mode Allows the SR[TCE] to be written even when the Status Register is locked. When set, the SR[TCE] can always be written if the SR[TIF] or SR[TOF] are set or if the SR[TCE] is clear. Allows the monotonic enable register to be written when it is locked. When set, the monotonic enable register can always be written if the SR[TIF] or SR[MOF] are set or if the montonic counter enable is clear. Allows the tamper detect register to be written when it is locked. When set, the tamper detect register can always be written if the SR[TIF] is clear. 0 Registers cannot be written when locked. 1 Registers can be written when locked under limited conditions.
2 SUP	Supervisor Access Configures non-supervisor mode write access to all RTC registers and non-supervisor mode read access to RTC tamper/monotonic registers 0 Non-supervisor mode write accesses are not supported and generate a bus error. 1 Non-supervisor mode write accesses are supported.
1 WPE	Wakeup Pin Enable The wakeup pin is optional and not available on all devices. 0 Wakeup pin is disabled. 1 Wakeup pin is enabled and wakeup pin asserts if the RTC interrupt asserts and the chip is powered down.
0 SWR	Software Reset 0 No effect 1 Resets all RTC registers except for the SWR bit and the RTC_WAR and RTC_RAR registers. The SWR bit is cleared after VBAT POR and by software explicitly clearing it.

44.2.6 RTC Status Register (RTC_SR)

Address: RTC_SR is 4003_D000h base + 14h offset = 4003_D014h

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											TCE	MOF	TAF	TOF	TIF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

RTC_SR field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value zero.
4 TCE	Time Counter Enable When time counter is disabled the TSR register and TPR register are writeable, but do not increment. When time counter is enabled the TSR register and TPR register are not writeable, but increment. 0 Time counter is disabled. 1 Time counter is enabled.
3 MOF	Monotonic Overflow Flag Monotonic overflow flag is set when the monotonic counter is enabled and the monotonic counter high overflows. The monotonic counter does not increment and will read as zero when this bit is set. This bit is cleared by writing the monotonic counter high register when the monotonic counter is disabled. 0 Monotonic counter overflow has not occurred. 1 Monotonic counter overflow has occurred and monotonic counter is read as zero.
2 TAF	Time Alarm Flag Time alarm flag is set when the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. This bit is cleared by writing the TAR register. 0 Time alarm has not occurred. 1 Time alarm has occurred.
1 TOF	Time Overflow Flag Time overflow flag is set when the time counter is enabled and overflows. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. 0 Time overflow has not occurred. 1 Time overflow has occurred and time counter is read as zero.

Table continues on the next page...

RTC_SR field descriptions (continued)

Field	Description
0 TIF	Time Invalid Flag The time invalid flag is set on VBAT POR or software reset. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled. The time invalid flag is also set if an enabled tamper source is detected or on any write to the tamper time seconds register. Software must disable the tamper source or clear the tamper detect flag before cleaing the time invalid flag. The monotonic counter register is held in reset whenever the time invalid flag is set. 0 Time is valid. 1 Time is invalid and time counter is read as zero.

44.2.7 RTC Lock Register (RTC_LR)

Address: RTC_LR is 4003_D000h base + 18h offset = 4003_D018h

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																TIL	TTL	TDL	TEL	MCHL	MCLL	MEL	TTSL	1	LRL	SRL	CRL	TCL	1		
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

RTC_LR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15 TIL	Tamper Interrupt Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Tamper interrupt register is locked and writes are ignored. 1 Tamper interrupt register is not locked and writes complete as normal.
14 TTL	Tamper Trim Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Tamper trim register is locked and writes are ignored. 1 Tamper trim register is not locked and writes complete as normal.
13 TDL	Tamper Detect Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Tamper detect register is locked and writes are ignored. 1 Tamper detect register is not locked and writes complete as normal.
12 TEL	Tamper Enable Lock

Table continues on the next page...

RTC_LR field descriptions (continued)

Field	Description
	Once cleared, this bit can only be set by VBAT POR or software reset. 0 Tamper enable register is locked and writes are ignored. 1 Tamper enable register is not locked and writes complete as normal.
11 MCHL	Monotonic Counter High Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Monotonic counter high register is locked and writes are ignored. 1 Monotonic counter high register is not locked and writes complete as normal.
10 MCLL	Monotonic Counter Low Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Monotonic counter low register is locked and writes are ignored. 1 Monotonic counter low register is not locked and writes complete as normal.
9 MEL	Monotonic Enable Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Monotonic enable register is locked and writes are ignored. 1 Monotonic enable register is not locked and writes complete as normal.
8 TTSL	Tamper Time Seconds Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Tamper time seconds register is locked and writes are ignored. 1 Tamper time seconds register is not locked and writes complete as normal.
7 Reserved	This read-only field is reserved and always has the value one.
6 LRL	Lock Register Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Lock register is locked and writes are ignored. 1 Lock register is not locked and writes complete as normal.
5 SRL	Status Register Lock Once cleared, this bit can only be set by VBAT POR or software reset. 0 Status register is locked and writes are ignored. 1 Status register is not locked and writes complete as normal.
4 CRL	Control Register Lock Once cleared, this bit can only be set by VBAT POR. 0 Control register is locked and writes are ignored. 1 Control register is not locked and writes complete as normal.
3 TCL	Time Compensation Lock

Table continues on the next page...

RTC_LR field descriptions (continued)

Field	Description
	Once cleared, this bit can only be set by VBAT POR or software reset. 0 Time compensation register is locked and writes are ignored. 1 Time compensation register is not locked and writes complete as normal.
2–0 Reserved	This read-only field is reserved and always has the value one.

44.2.8 RTC Interrupt Enable Register (RTC_IER)

Address: RTC_IER is 4003_D000h base + 1Ch offset = 4003_D01Ch

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																								Reserved			TSIE	MOIE	TAIE	TOIE	TIIE
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	1	1	1

RTC_IER field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7–5 Reserved	This field is reserved.
4 TSIE	Time Seconds Interrupt Enable The seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector. It is generated once a second and requires no software overhead (there is no corresponding status flag to clear). 0 Seconds interrupt is disabled. 1 Seconds interrupt is enabled.
3 MOIE	Monotonic Overflow Interrupt Enable 0 Monotonic overflow flag does not generate an interrupt. 1 Monotonic overflow flag does generate an interrupt.
2 TAIE	Time Alarm Interrupt Enable 0 Time alarm flag does not generate an interrupt. 1 Time alarm flag does generate an interrupt.
1 TOIE	Time Overflow Interrupt Enable 0 Time overflow flag does not generate an interrupt. 1 Time overflow flag does generate an interrupt.
0 TIIE	Time Invalid Interrupt Enable

Table continues on the next page...

RTC_IER field descriptions (continued)

Field	Description
	0 Time invalid flag does not generate an interrupt. 1 Time invalid flag does generate an interrupt.

44.2.9 RTC Tamper Time Seconds Register (RTC_TTSR)

Address: RTC_TTSR is 4003_D000h base + 20h offset = 4003_D020h

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	TTS																															
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

RTC_TTSR field descriptions

Field	Description
31–0 TTS	Tamper Time Seconds If the time invalid flag is set then reading this register returns the contents of the time seconds register at the point at which the time invalid flag was set. If the time invalid flag is clear then reading this register returns zero. Writing the tamper time seconds register with any value will set the time invalid flag.

44.2.10 RTC Monotonic Enable Register (RTC_MER)

Address: RTC_MER is 4003_D000h base + 24h offset = 4003_D024h

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																											MCE	0			
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

RTC_MER field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value zero.
4 MCE	Monotonic Counter Enable 0 Writes to the monotonic counter load the counter with the value written. 1 Writes to the monotonic counter increment the counter.
3–0 Reserved	This read-only field is reserved and always has the value zero.

44.2.11 RTC Monotonic Counter Low Register (RTC_MCLR)

Address: RTC_MCLR is 4003_D000h base + 28h offset = 4003_D028h

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="32" rowspan="2">MCL</td></tr>
<tr><td>W</td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

RTC_MCLR field descriptions

Field	Description
31–0 MCL	Monotonic Counter Low When the time invalid flag is set, the monotonic counter is held in reset. When the monotonic counter enable is clear, a write to this register will load the counter with the value written. When the monotonic counter enable is set, a write to this register will cause it to increment. A write to monotonic counter low that causes it to overflow will also increment monotonic counter high.

44.2.12 RTC Monotonic Counter High Register (RTC_MCHR)

Address: RTC_MCHR is 4003_D000h base + 2Ch offset = 4003_D02Ch

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="32" rowspan="2">MCH</td></tr>
<tr><td>W</td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

RTC_MCHR field descriptions

Field	Description
31–0 MCH	Monotonic Counter High When the time invalid flag is set, the monotonic counter is held in reset. When the monotonic counter enable is clear, a write to this register will load the counter with the value written. When the monotonic counter enable is set, a write to this register will cause it to increment. A write to monotonic counter low that causes it to overflow will also increment monotonic counter high.

44.2.13 RTC Tamper Enable Register (RTC_TER)

Address: RTC_TER is 4003_D000h base + 30h offset = 4003_D030h

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																										TME	FSE	TTE	CTE	VTE	0
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

RTC_TER field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5 TME	Test Mode Enable 0 Tamper source disabled. 1 Set the time invalid flag if the test mode flag is set.
4 FSE	Flash Security Enable 0 Tamper source disabled. 1 Set the time invalid flag if the flash security flag is set.
3 TTE	Temperature Tamper Enable When set, this bit enables the analog temperature tamper detect circuit. 0 Tamper source disabled. 1 Set the time invalid flag if the temperature tamper flag is set.
2 CTE	Clock Tamper Enable When set, this bit enables the analog clock tamper detect circuit. 0 Tamper source disabled. 1 Set the time invalid flag if the clock tamper flag is set.
1 VTE	Voltage Tamper Enable When set, this bit enables the analog voltage tamper detect circuit. 0 Tamper source disabled. 1 Set the time invalid flag if the voltage tamper flag is set.
0 Reserved	This read-only field is reserved and always has the value zero.

44.2.14 RTC Tamper Detect Register (RTC_TDR)

Address: RTC_TDR is 4003_D000h base + 34h offset = 4003_D034h

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										TMF	FSF	TTF	CTF	VTF	1
W											w1c	w1c	w1c	w1c	w1c	
Reset	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	0*	1*

* Notes:

- A software reset will clear all tamper flags, but flags can set again once software reset is negated.

RTC_TDR field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5 TMF	Test Mode Flag This flag is set whenever any test mode is entered. To clear, write logic one to this flag after exiting from all test modes. 0 Tamper not detected. 1 Test mode tamper detected.
4 FSF	Flash Security Flag This flag is set whenever flash security is disabled. To clear, write logic one to this flag after flash security is enabled. 0 Tamper not detected. 1 Flash security tamper detected.
3 TTF	Temperature Tamper Flag This flag is set when the junction temperature is outside of specification. To clear, write logic one to this flag after the junction temperature has returned to be within the valid range. 0 Tamper not detected. 1 Temperature tampering detected.
2 CTF	Clock Tamper Flag This flag is set when the 32.768 kHz clock source is outside the valid range. To clear, write logic one to this flag after the 32.768 kHz clock source has returned to be within the valid range.

Table continues on the next page...

RTC_TDR field descriptions (continued)

Field	Description
	0 Tamper not detected. 1 Clock tampering detected.
1 VTF	Voltage Tamper Flag This flag is set when the VBAT voltage is outside the valid range. To clear, write logic one to this flag after the VBAT voltage has returned to be within the valid range. 0 Tamper not detected. 1 Voltage tampering detected.
0 Reserved	This read-only field is reserved and always has the value one.

44.2.15 RTC Tamper Trim Register (RTC_TTR)

Address: RTC_TTR is 4003_D000h base + 38h offset = 4003_D038h

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			0											TDTH			TDTL			CDTH			CDTL			VDTH			VDTL		
W	0																															
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

RTC_TTR field descriptions

Field	Description
31–29 Reserved	This field is reserved. It must always be written to 000.
28–18 Reserved	This read-only field is reserved and always has the value zero.
17–15 TDTH	Temperature Detect Trim High Trims the analog tamper high temperature detect trip point. This field is loaded with factory programmed IFR data during each chip reset unless the tamper trim register is locked.
14–12 TDTL	Temperature Detect Trim Low Trims the analog tamper low temperature detect trip point. This field is loaded with factory programmed IFR data during each chip reset unless the tamper trim register is locked.
11–9 CDTH	Clock Detect Trim High Trims the analog tamper high clock frequency detect trip point. This field is loaded with factory programmed IFR data during each chip reset unless the tamper trim register is locked.
8–6 CDTL	Clock Detect Trim Low Trims the analog tamper low clock frequency detect trip point. This field is loaded with factory programmed IFR data during each chip reset unless the tamper trim register is locked.

Table continues on the next page...

RTC_TTR field descriptions (continued)

Field	Description
5–3 VDTH	Voltage Detect Trim High Trims the analog tamper high voltage detect trip point. This field is loaded with factory programmed IFR data during each chip reset unless the tamper trim register is locked.
2–0 VDTL	Voltage Detect Trim Low Trims the analog tamper low voltage detect trip point. This field is loaded with factory programmed IFR data during each chip reset unless the tamper trim register is locked.

44.2.16 RTC Tamper Interrupt Register (RTC_TIR)

Address: RTC_TIR is 4003_D000h base + 3Ch offset = 4003_D03Ch

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																										TMIE	FSIE	TTIE	CTIE	VTIE	0
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

RTC_TIR field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value zero.
5 TMIE	Test Mode Interrupt Enable 0 Interupt disabled. 1 An interrupt is generated when the test mode flag is set.
4 FSIE	Flash Security Interrupt Enable 0 Interupt disabled. 1 An interrupt is generated when the flash security flag is set.
3 TTIE	Temperature Tamper Interrupt Enable When set, this bit enables the analog temperature tamper detect circuit. 0 Interupt disabled. 1 An interrupt is generated when the temperature tamper flag is set.
2 CTIE	Clock Tamper Interrupt Enable When set, this bit enables the analog clock tamper detect circuit. 0 Interupt disabled. 1 An interrupt is generated when the clock tamper flag is set.
1 VTIE	Voltage Tamper Interrupt Enable

Table continues on the next page...

RTC_TIR field descriptions (continued)

Field	Description
	When set, this bit enables the analog voltage tamper detect circuit. 0 Interupt disabled. 1 An interrupt is generated when the voltage tamper flag is set.
0 Reserved	This read-only field is reserved and always has the value zero.

44.2.17 RTC Write Access Register (RTC_WAR)

Address: RTC_WAR is 4003_D000h base + 800h offset = 4003_D800h

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																TIRW	TTRW	TDRW	TERW	MCHW	MCLW	MERW	TTSW	IERW	LRW	SRW	CRW	TCRW	TARW	TPRW	TSRW
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

RTC_WAR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15 TIRW	Tamper Interrupt Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the tamper interrupt register are ignored. 1 Writes to the tamper interrupt register complete as normal.
14 TTRW	Tamper Trim Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the tamper trim register are ignored. 1 Writes to the tamper trim register complete as normal.
13 TDRW	Tamper Detect Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the tamper detect register are ignored. 1 Writes to the tamper detect register complete as normal.
12 TERW	Tamper Enable Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the tamper enable register are ignored. 1 Writes to the tamper enable register complete as normal.

Table continues on the next page...

RTC_WAR field descriptions (continued)

Field	Description
11 MCHW	Monotonic Counter High Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the monotonic counter high register are ignored. 1 Writes to the monotonic counter high register complete as normal.
10 MCLW	Monotonic Counter Low Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the monotonic counter low register are ignored. 1 Writes to the monotonic counter low register complete as normal.
9 MERW	Monotonic Enable Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the monotonic enable register are ignored. 1 Writes to the monotonic enable register complete as normal.
8 TTSW	Tamper Time Seconds Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the tamper time seconds register are ignored. 1 Writes to the tamper time seconds register complete as normal.
7 IERW	Interrupt Enable Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the interupt enable register are ignored. 1 Writes to the interrupt enable register complete as normal.
6 LRW	Lock Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the lock register are ignored. 1 Writes to the lock register complete as normal.
5 SRW	Status Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the status register are ignored. 1 Writes to the status register complete as normal.
4 CRW	Control Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the control register are ignored. 1 Writes to the control register complete as normal.
3 TCRW	Time Compensation Register Write

Table continues on the next page...

RTC_WAR field descriptions (continued)

Field	Description
	Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the time compensation register are ignored. 1 Writes to the time compensation register complete as normal.
2 TARW	Time Alarm Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the time alarm register are ignored. 1 Writes to the time alarm register complete as normal.
1 TPRW	Time Prescaler Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the time prescaler register are ignored. 1 Writes to the time prescaler register complete as normal.
0 TSRW	Time Seconds Register Write Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Writes to the time seconds register are ignored. 1 Writes to the time seconds register complete as normal.

44.2.18 RTC Read Access Register (RTC_RAR)

Address: RTC_RAR is 4003_D000h base + 804h offset = 4003_D804h

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																TIRR	TTRR	TDRR	TERR	MCHR	MCLR	MERR	TTSR	IERR	LRR	SRR	CRR	TCRR	TARR	TPRR	TSRR
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

RTC_RAR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15 TIRR	Tamper Interrupt Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the tamper interrupt register are ignored. 1 Reads to the tamper interrupt register complete as normal.
14 TTRR	Tamper Trim Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset.

Table continues on the next page...

RTC_RAR field descriptions (continued)

Field	Description
	0 Reads to the tamper trim register are ignored. 1 Reads to the tamper trim register complete as normal.
13 TDRR	Tamper Detect Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the tamper detect register are ignored. 1 Reads to the tamper detect register complete as normal.
12 TERR	Tamper Enable Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the tamper enable register are ignored. 1 Reads to the tamper enable register complete as normal.
11 MCHR	Monotonic Counter High Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the monotonic counter high register are ignored. 1 Reads to the monotonic counter high register complete as normal.
10 MCLR	Monotonic Counter Low Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the monotonic counter low register are ignored. 1 Reads to the monotonic counter low register complete as normal.
9 MERR	Monotonic Enable Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the monotonic enable register are ignored. 1 Reads to the monotonic enable register complete as normal.
8 TTSR	Tamper Time Seconds Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the tamper time seconds register are ignored. 1 Reads to the tamper time seconds register complete as normal.
7 IERR	Interrupt Enable Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the interrupt enable register are ignored. 1 Reads to the interrupt enable register complete as normal.
6 LRR	Lock Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the lock register are ignored. 1 Reads to the lock register complete as normal.

Table continues on the next page...

RTC_RAR field descriptions (continued)

Field	Description
5 SRR	Status Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the status register are ignored. 1 Reads to the status register complete as normal.
4 CRR	Control Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the control register are ignored. 1 Reads to the control register complete as normal.
3 TCRR	Time Compensation Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset 0 Reads to the time compensation register are ignored. 1 Reads to the time compensation register complete as normal.
2 TARR	Time Alarm Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the time alarm register are ignored. 1 Reads to the time alarm register complete as normal.
1 TPRR	Time Prescaler Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the time prescaler register are ignored. 1 Reads to the time prescaler register complete as normal.
0 TSRR	Time Seconds Register Read Once cleared, this bit is only set by system reset. It is not affected by VBAT POR or software reset. 0 Reads to the time seconds register are ignored. 1 Reads to the time seconds register complete as normal.

44.3 Functional description

44.3.1 Power, clocking and reset

The RTC is an always powered block that is powered by the battery power supply (VBAT). The battery power supply ensures that the RTC registers retain their state during chip power-down and that the RTC time counter remains operational.

The time counter within the RTC is clocked by a 32.768 kHz clock and can supply this clock to other peripherals. The 32.768 kHz clock can only be sourced from an external crystal using the oscillator that is part of the RTC module.

The RTC includes its own analog POR block, which generates a power-on-reset signal whenever the RTC module is powered up and initializes all RTC registers to their default state. A software reset bit can also initialize all RTC registers. The RTC also monitors the chip power supply and electrically isolates itself when the rest of the chip is powered down.

Any attempt to access an RTC register (except the access control registers) when VBAT is powered down, when the RTC is electrically isolated, or when VBAT POR is asserted, will result in a bus error.

44.3.1.1 Oscillator control

The 32.768 kHz crystal oscillator is disabled at VBAT POR and must be enabled by software. After enabling the cystal oscillator, wait the oscillator startup time before setting the SR[TCE] bit or using the oscillator clock external to the RTC.

The crystal oscillator includes tunable capacitors that can be configured by software. Do not change the capacitance unless the oscillator is disabled.

44.3.1.2 Software reset

Writing one to the CR[SWR] forces the equivalent of a VBAT POR to the rest of the RTC module. The CR[SWR] is not affected by the software reset and must be cleared by software. The access control registers are not affected by either VBAT POR or the software reset; they are reset by the chip reset.

44.3.1.3 Supervisor access

When the supervisor access control bit is clear, only supervisor mode software can write to the RTC registers or read the RTC tamper and monotonic registers, non-supervisor mode software will generate a bus error. Both supervisor and non-supervisor mode software can always read the other RTC registers.

44.3.2 Time counter

The time counter consists of a 32-bit seconds counter that increments once every second and a 16-bit prescaler register that increments once every 32.768 kHz clock cycle.

The time seconds register and time prescaler register can only be written when the SR[TCE] bit is clear. Always write to the prescaler register before writing to the seconds register, since the seconds register increments on the falling edge of bit 14 of the prescaler register.

The time prescaler register increments provided the SR[TCE] bit is set, the SR[TIF] is clear, the SR[TOF] is clear and the 32.768 kHz clock source is present. After enabling the oscillator, wait the oscillator startup time before setting the SR[TCE] bit to allow time for the oscillator clock output to stabilize.

If the time seconds register overflows then the SR[TOF] will set and the time prescaler register will stop incrementing. Clear the SR[TOF] by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever the SR[TOF] is set.

The SR[TIF] is set on VBAT POR, software reset and when an enabled tamper source is detected. It is cleared by clearing the enabled tamper detect flag and then initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever the SR[TIF] is set.

44.3.3 Compensation

The compensation logic provides an accurate and wide compensation range and can correct errors as high as 3906 ppm and as low as 0.12 ppm. Note that the compensation factor must be calculated externally to the RTC and supplied by software to the compensation register. The RTC itself does not calculate the amount of compensation that is required, although the 1 Hz clock is output to an external pin in support of external calibration logic.

Crystal compensation can be supported by using firmware and crystal characteristics to determine the compensation amount. Temperature compensation can be supported by firmware that periodically measures the external temperature (via ADC) and updates the compensation register based on a look-up table that specifies the change in crystal frequency over temperature.

The compensation logic alters the number of 32.768 kHz clock cycles it takes for the prescaler register to overflow and increment the time seconds counter. The time compensation value is used to adjust the number of clock cycles between -127 and +128. Cycles are added or subtracted from the prescaler register when the prescaler register equals 0x3FFF and then increments. The compensation interval is used to adjust the frequency at which the time compensation value is used (from once a second to once every 256 seconds).

Updates to the time compensation register will not take effect until the next time the time seconds register increments and provided the previous compensation interval has expired. When the compensation interval is set to other than once a second then the compensation is applied in the first second interval and the remaining second intervals receive no compensation.

Compensation is disabled by configuring the time compensation register to zero.

44.3.4 Time alarm

The time alarm register, SR[TAF] and IER[TAIE] allow the RTC to generate an interrupt at a predefined time. The 32-bit time alarm register is compared with the 32-bit time seconds register each time it increments. The SR[TAF] will set when the time alarm register equals the time seconds register and the time seconds register increments.

The time alarm flag is cleared by writing the time alarm register. This will usually be the next alarm value, although writing a value that is less than the time seconds register (such as zero) will prevent the time alarm flag from setting again. The time alarm flag cannot otherwise be disabled, although the interrupt it generates is enabled or disabled by IER[TAIE].

44.3.5 Update mode

The update mode bit (CR[UM]) in the control register configures software write access to the time counter enable (SR[TCE]) bit. When CR[UM] is clear, SR[TCE] can only be written when the LR[SRL] bit is set. When CR[UM] is set, the SR[TCE] can also be written when SR[TCE] is clear or when SR[TIF] or SR[TOF] are set. This allows the time seconds and prescaler registers to be initialized whenever time is invalidated, while preventing the time seconds and prescaler registers from being changed on the fly. When LR[SRL] is set, the CR[UM] bit has no effect on SR[TCE].

The CR[UM] also configures software write access to the monotonic counter enable (MER[MCE]) bit. When CR[UM] is clear, MER[MCE] can only be written when the LR[MEL] bit is set. When CR[UM] is set, the MER[MCE] can also be written when MER[MCE] is clear or when SR[TIF] or SR[MOF] are set. This allows the monotonic counter register to be initialized whenever the monotonic counter is invalid, while preventing the monotonic counter from being changed on the fly. When LR[MEL] is set, the CR[UM] bit has no effect on MCR[MCE].

The CR[UM] also configures software write access to the tamper detect register. When CR[UM] is clear, the tamper detect register can only be written when the LR[TDL] bit is set. When CR[UM] is set, the tamper detect register can also be written when SR[TIF] is clear. This allows tamper interrupts to be cleared provided the time has not been invalidated. When LR[TDL] bit is set, the CR[UM] bit has no effect on the tamper detect register.

44.3.6 Monotonic counter

The 64-bit Monotonic Counter is a counter that cannot be exhausted or return to any previous value, once it has been initialized. If the monotonic overflow flag is set, the monotonic counter returns zero and does not increment.

Depending on the value of the monotonic counter enable bit, writing to the monotonic counter either initializes the register with the value written, or increments the register by one (and the value written is ignored).

When the monotonic counter is enabled, the monotonic counter high increments on either a write to the monotonic counter high register or if the monotonic counter low register overflows (due to a write to the monotonic counter low register). The monotonic overflow flag sets when the monotonic counter high register overflows and is cleared by writing the monotonic counter high register when the monotonic counter is disabled.

The monotonic counter is held in reset whenever the time invalid flag is set. Always clear the time invalid flag before initializing the monotonic counter.

44.3.7 Tamper detect

Each tamper source can be enabled to invalidate the time when the tamper is detected. Each flag in the tamper detect register will set when the appropriate tamper source asserts. If the corresponding bit in the tamper enable register is set then the time is invalidated (and the time invalid flag is set).

The flags in the tamper detect register are cleared by software writing a logic one to the appropriate flag provided the tamper source has negated. An enabled tamper detect register flag must be cleared before attempting to clear the time invalid flag.

When the time invalid flag is set, the tamper time seconds register records the contents of the time seconds register at the time the tamper event was detected.

Writing the tamper time seconds register at any time will set the time invalid flag. Since this is a software initiated tamper, there is no status flag to indicate this tamper source. To disable the software initiated tamper, lock the tamper time seconds register to prevent write accesses to that register.

44.3.8 Register lock

The lock register can be used to block write accesses to certain registers until the next VBAT POR or software reset. Locking the control register will disable the software reset. Locking the lock register will block future updates to the lock register.

Write accesses to a locked register are ignored and do not generate a bus error.

44.3.9 Access control

The read access and write access registers are implemented in the chip power domain and reset on the chip reset (they are not affected by the VBAT POR or the software reset). They are used to block read or write accesses to each register until the next chip system reset. When accesses are blocked the bus access is not seen in the VBAT power supply and does not generate a bus error.

44.3.10 Interrupt

The RTC Interrupt is asserted whenever a status flag and the corresponding interrupt enable bit are both set. It is always asserted on VBAT POR, software reset and when the VBAT power supply is powered down. The RTC interrupt is enabled at the chip level by enabling the chip-specific RTC clock gate control bit. The RTC Interrupt can be used to wakeup the chip from any low power mode.

The optional RTC seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector that is generated once a second and requires no software overhead (there is no corresponding status flag to clear). It is enabled in the RTC by the time seconds interrupt enable bit and enabled at the chip level by setting the chip-specific RTC clock gate control bit. The RTC seconds interrupt does not cause the RTC wakeup pin to assert. This interrupt is optional and may not be implemented on all devices.

Chapter 45
Universal Serial Bus OTG Controller (USBOTG)

45.1 Introduction

NOTE

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

This section describes the USB. The OTG implementation in this module provides limited host functionality as well as device solutions for implementing a USB 2.0 full-speed/low-speed compliant peripheral. The OTG implementation supports the On-The-Go (OTG) addendum to the USB 2.0 Specification. Only one protocol can be active at any time. A negotiation protocol must be used to switch to a USB host functionality from a USB device. This is known as the Master Negotiation Protocol (MNP).

45.1.1 USB

The USB is a cable bus that supports data exchange between a host computer and a wide range of simultaneously accessible peripherals. The attached peripherals share USB bandwidth through a host-scheduled, token-based protocol. The bus allows peripherals to be attached, configured, used, and detached while the host and other peripherals are in operation.

USB software provides a uniform view of the system for all application software, hiding implementation details making application software more portable. It manages the dynamic attach and detach of peripherals.

There is only one host in any USB system. The USB interface to the host computer system is referred to as the Host Controller.

There may be multiple USB devices in any system such as joysticks, speakers, printers, etc. USB devices present a standard USB interface in terms of comprehension, response, and standard capability.

The host initiates transactions to specific peripherals, while the device responds to control transactions. The device sends and receives data to and from the host using a standard USB data format. USB 2.0 full-speed /low-speed peripherals operate at 12Mb/s or 1.5 Mb/s.

For additional information, refer to the USB 2.0 specification.

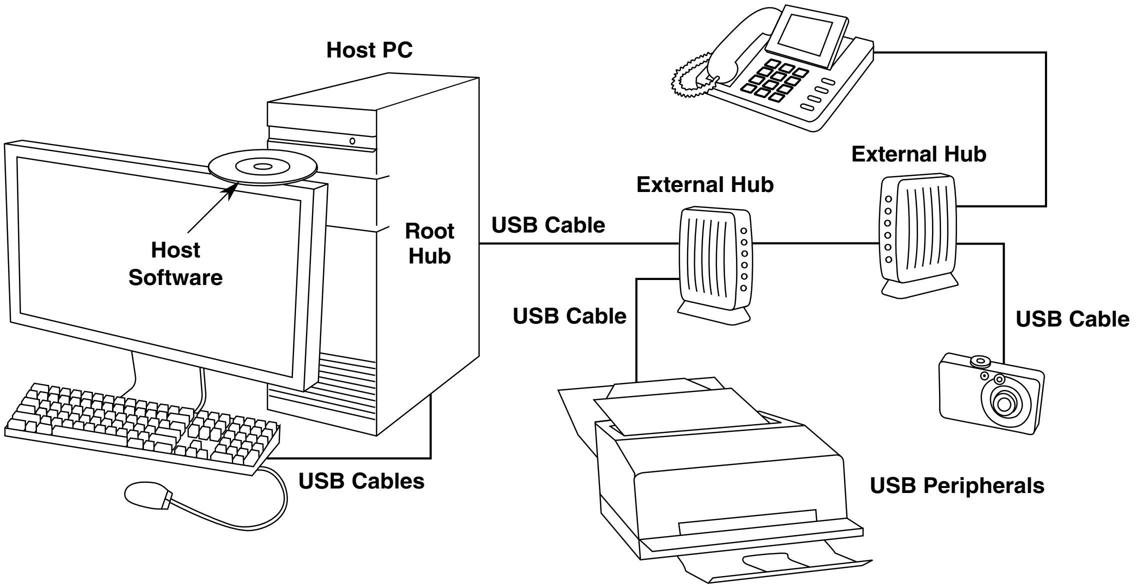


Figure 45-1. Example USB 2.0 System Configuration

45.1.2 USB On-The-Go

USB (Universal Serial Bus) is a popular standard for connecting peripherals and portable consumer electronic devices such as digital cameras and hand-held computers to host PCs. The On-The-Go (OTG) Supplement to the USB Specification extends USB to peer-to-peer application. Using USB OTG technology consumer electronics, peripherals and portable devices can connect to each other (for example, a digital camera can connect directly to a printer, or a keyboard can connect to a Personal Digital Assistant) to exchange data.

With the USB On-The-Go product, you can develop a fully USB-compliant peripheral device that can also assume the role of a USB host. Software determines the role of the device based on hardware signals, and then initializes the device in the appropriate mode of operation (host or peripheral) based on how it is connected. After connecting the devices can negotiate using the OTG protocols to assume the role of host or peripheral based on the task to be accomplished.

For additional information, refer to the *On-The-Go Supplement to the USB 2.0 Specification*.

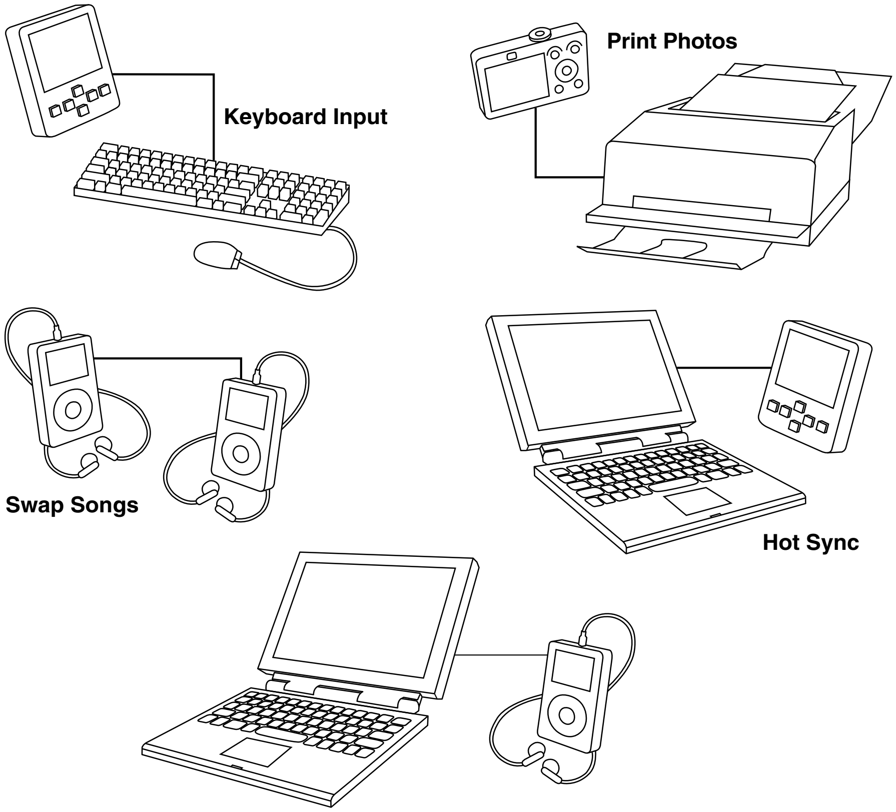


Figure 45-2. Example USB 2.0 On-The-Go Configurations

45.1.3 USB-FS Features

- USB 1.1 and 2.0 compliant full-speed device controller
- 16-Bidirectional end points
- DMA or FIFO data stream interfaces
- Low-power consumption
- On-The-Go protocol logic

45.2 Functional Description

The USB-FS 2.0 full-speed/low-speed module communicates with the processor core through status registers, control registers, and data structures in memory.

45.2.1 Data Structures

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus. To efficiently manage USB endpoint communications the USB-FS implements a Buffer Descriptor Table (BDT) in system memory. See Figure 45-3.

45.3 Programmers Interface

This section discusses the major components of the programming model for the USB module.

45.3.1 Buffer Descriptor Table

To efficiently manage USB endpoint communications the USB-FS implements a Buffer Descriptor Table (BDT) in system memory. The BDT resides on a 512 byte boundary in system memory and is pointed to by the BDT Page Registers. Every endpoint direction requires two eight-byte Buffer Descriptor entries. Therefore, a system with 16 fully bidirectional endpoints would require 512 bytes of system memory to implement the BDT. The two Buffer Descriptor (BD) entries allows for an EVEN BD and ODD BD entry for each endpoint direction. This allows the microprocessor to process one BD while the USB-FS is processing the other BD. Double buffering BDs in this way allows the USB-FS to easily transfer data at the maximum throughput provided by USB.

The software API intelligently manages buffers for the USB-FS by updating the BDT when needed. This allows the USB-FS to efficiently manage data transmission and reception, while the microprocessor performs communication overhead processing and other function dependent applications. Because the buffers are shared between the microprocessor and the USB-FS a simple semaphore mechanism is used to distinguish who is allowed to update the BDT and buffers in system memory. A semaphore bit, the OWN bit, is cleared to 0 when the BD entry is owned by the microprocessor. The microprocessor is allowed read and write access to the BD entry and the buffer in system memory when the OWN bit is 0. When the OWN bit is set to 1, the BD entry and the buffer in system memory are owned by the USB-FS. The USB-FS now has full read and write access and the microprocessor should not modify the BD or its corresponding data buffer. The BD also contains indirect address pointers to where the actual buffer resides in system memory. This indirect address mechanism is shown in the following diagram.

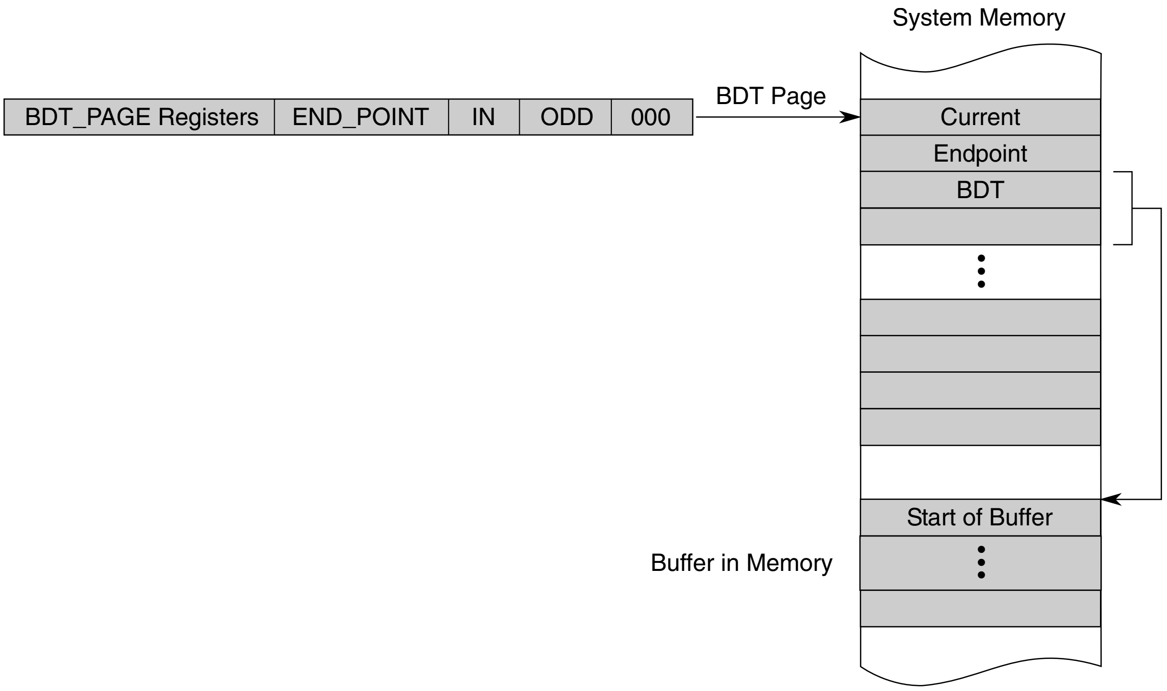


Figure 45-3. Buffer Descriptor Table

45.3.2 Rx vs. Tx as a USB Target Device or USB Host

The USB-FS core uses software control to switch between two modes of operation:

- USB target device
- USB hosts

In either mode, USB host or USB target device, the same data paths and buffer descriptors are used for the transmission and reception of data. For this reason, a USB-FS core centric nomenclature is used to describe the direction of the data transfer between the USB-FS core and the USB:

Rx (or receive)

describes transfers that move data from the USB to memory.

Tx (or transmit)

describes transfers that move data from memory to the USB.

The following table shows how the data direction corresponds to the USB token type in host and target device applications.

Table 45-1. Data Direction for USB Host or USB Target

	Rx	Tx
Device	OUT or Setup	IN

Table continues on the next page...

Table 45-1. Data Direction for USB Host or USB Target (continued)

	Rx	Tx
Host	IN	Out or Setup

45.3.3 Addressing Buffer Descriptor Table Entries

An understanding of the addressing mechanism of the Buffer Descriptor Table is useful when accessing endpoint data via the USB-FS or microprocessor. Some points of interest are:

- The Buffer Descriptor Table occupies up to 512 bytes of system memory.
- 16 bidirectional endpoints can be supported with a full BDT of 512 bytes.
- 16 bytes are needed for each USB endpoint direction.
- Applications with less than 16 endpoints require less RAM to implement the BDT.
- The BDT Page Registers point to the starting location of the BDT.
- The BDT must be located on a 512-byte boundary in system memory.
- All enabled TX and RX endpoint BD entries are indexed into the BDT to allow easy access via the USB-FS or MCU core.

When a USB token on an enabled endpoint is received, the USB-FS uses its integrated DMA controller to interrogate the BDT. The USB-FS reads the corresponding endpoint BD entry to determine if it owns the BD and corresponding buffer in system memory.

To compute the entry point in to the BDT, the BDT_PAGE registers is concatenated with the current endpoint and the TX and ODD fields to form a 32-bit address. This address mechanism is shown in the following diagrams:

Table 45-2. BDT Address Calculation Fields

Field	Description
BDT_PAGE	BDT_PAGE registers in the Control Register Block
END_POINT	END POINT field from the USB TOKEN
TX	1 for an TX transmit transfers and 0 for an RX receive transfers
ODD	This bit is maintained within the USB-FS SIE. It corresponds to the buffer currently in use. The buffers are used in a ping-pong fashion.

45.3.4 Buffer Descriptor Formats

The Buffer Descriptors (BD) provide endpoint buffer control information for the USB-FS and microprocessor. The Buffer Descriptors have different meaning based on whether it is the USB-FS or microprocessor reading the BD in memory.

The USB-FS Controller uses the data stored in the BDs to determine:

- Who owns the buffer in system memory
- Data0 or Data1 PID
- Release Own upon packet completion
- No address increment (FIFO Mode)
- Data toggle synchronization enable
- How much data is to be transmitted or received
- Where the buffer resides in system memory

While the microprocessor uses the data stored in the BDs to determine:

- Who owns the buffer in system memory
- Data0 or Data1 PID
- The received TOKEN PID
- How much data was transmitted or received
- Where the buffer resides in system memory

The format for the BD is shown in the following figure.

Table 45-3. Buffer Descriptor Byte Format

31:26	25:16	15:8	7	6	5	4	3	2	1	0
RSVD	BC (10 bits)	RSVD	OWN	DATA0/1	KEEP/ TOK_PID[3]	NINC/ TOK_PID[2]	DTS/ TOK_PID[1]	BDT_STALL/ TOK_PID[0]	0	0
Buffer Address (32-Bits)										

Table 45-4. Buffer Descriptor Byte Fields

Field	Description
31 –26 RSVD	Reserved
25 –16 BC[9:0]	The Byte Count bits represent the 10-bit Byte Count. The USB-FS SIE changes this field upon the completion of a RX transfer with the byte count of the data received.
15 –8 RSVD	Reserved

Table continues on the next page...

Table 45-4. Buffer Descriptor Byte Fields (continued)

Field	Description
7 OWN	The OWN bit determines whether the microprocessor or the USB-FS currently owns the buffer. Except when KEEP=1, the SIE writes a 0 to this bit when it has completed a token. This byte of the BD should always be the last byte the microprocessor updates when it initializes a BD. **0** The microprocessor has exclusive access to the BD. The USB-FS ignores all other fields in the BD. **1** USB-FS has exclusive access to the BD. After the BD has been assigned to the USB-FS, the microprocessor should not change it in any way.
6 DATA0/1	This bit defines if a DATA0 field (DATA0/1=0) or a DATA1 (DATA0/1=1) field was transmitted or received. It is unchanged by the USB-FS.
5 KEEP/ TOK_PID[3]	Typically this bit is set (that is, 1) with ISO endpoints feeding a FIFO. The microprocessor is not informed that a token has been processed, the data is simply transferred to or from the FIFO. If KEEP is set, normally the NINC bit is also set to prevent address increment. **0** Bit 3 of the current token PID is written back in to the BD by the USB-FS. Allows the USB-FS to release the BD when a token has been processed. **1** This bit is unchanged by the USB-FS. If the OWN bit also is set, the BD remains owned by the USB-FS forever.
4 NINC/ TOK_PID[2]	The No Increment (NINC) bit disables the DMA engine address increment. This forces the DMA engine to read or write from the same address. This is useful for endpoints when data needs to be read from or written to a single location such as a FIFO. Typically this bit is set with the KEEP bit for ISO endpoints that are interfacing to a FIFO. **0** the USB-FS writes bit 2 of the current token PID to the BD. **1** This bit is unchanged by the USB-FS.
3 DTS/ TOK_PID[1]	Setting this bit enables the USB-FS to perform Data Toggle Synchronization. • If KEEP=0, bit 1 of the current token PID is written back to the BD. • If KEEP=1, this bit is unchanged by the USB-FS. **0** Data Toggle Synchronization is disabled. **1** Enables the USB-FS to perform Data Toggle Synchronization.

Table continues on the next page...

Table 45-4. Buffer Descriptor Byte Fields (continued)

Field	Description
2 BDT_STALL TOK_PID[0]	Setting this bit causes the USB-FS to issue a STALL handshake if a token is received by the SIE that would use the BDT in this location. The BDT is not consumed by the SIE (the owns bit remains set and the rest of the BDT is unchanged) when a BDT-STALL bit is set. • If KEEP=0, bit 0 of the current token PID is written back to the BD. • If KEEP=1, this bit is unchanged by the USB-FS. **0** No stall issued. **1** The BDT is not consumed by the SIE (the OWN bit remains set and the rest of the BDT is unchanged).
TOK_PID[n]	Bits [5:2] can also represent the current token PID. The current token PID is written back in to the BD by the USB-FS when a transfer completes. The values written back are the token PID values from the USB specification: • 0x1 for an OUT token. • 0x9 for an IN token. • 0xd for a SETUP token. In host mode, this field is used to report the last returned PID or a transfer status indication. The possible values returned are: • 0x3 DATA0 • 0xb DATA1 • 0x2 ACK • 0xe STALL • 0xa NAK • 0x0 Bus Timeout • 0xf Data Error
1–0 Reserved	Reserved, should read as zeroes.
ADDR[31:0]	The Address bits represent the 32 -bit buffer address in system memory. These bits are unchanged by the USB-FS.

45.3.5 USB Transaction

When the USB-FS transmits or receives data, it computes the BDT address using the address generation shown in "Addressing Buffer Descriptor Entries" table.

If OWN =1, the following process occurs:

1. The USB-FS reads the BDT.
2. The SIE transfers the data via the DMA to or from the buffer pointed to by the ADDR field of the BD.
3. When the TOKEN is complete, the USB-FS updates the BDT and, if KEEP=0, changes the OWN bit to 0.
4. The STAT register is updated and the TOK_DNE interrupt is set.

5. When the microprocessor processes the TOK_DNE interrupt, it reads from the status register all the information needed to process the endpoint.
6. At this point, the microprocessor allocates a new BD so additional USB data can be transmitted or received for that endpoint, and then processes the last BD.

The following figure shows a timeline of how a typical USB token is processed after the BDT is read and OWN=1.

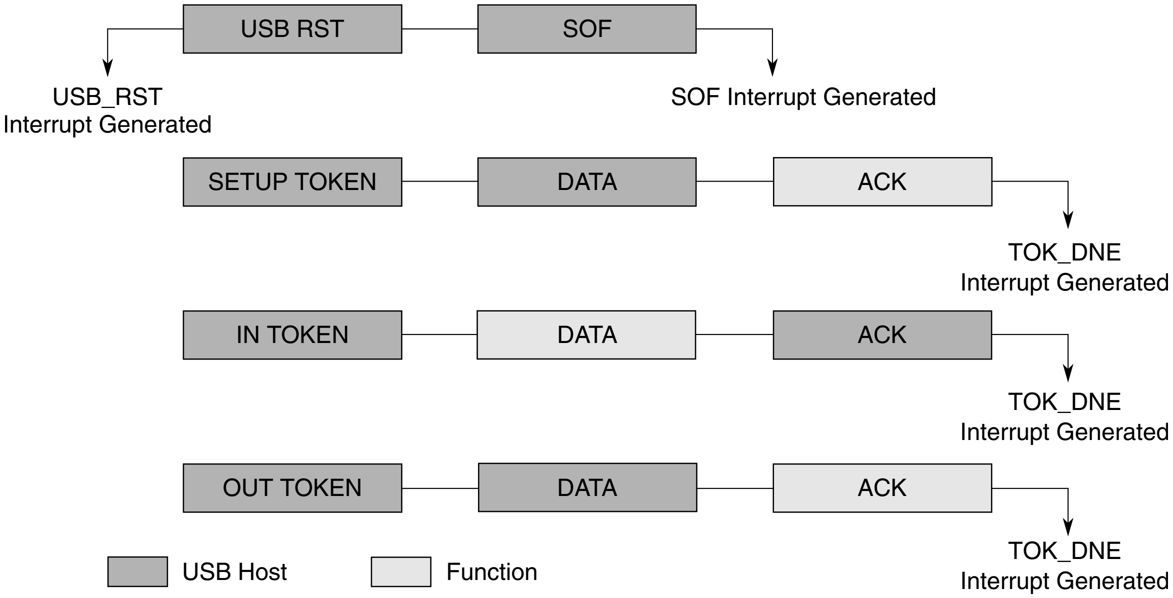


Figure 45-4. USB Token Transaction

The USB has two sources for the DMA overrun error:

Memory Latency

The memory latency on the BVCI initiator interface may be too high and cause the receive FIFO to overflow. This is predominantly a hardware performance issue, usually caused by transient memory access issues.

Oversized Packets

The packet received may be larger than the negotiated *MaxPacket* size. Typically, this is caused by a software bug. For DMA overrun errors due to oversized data packets, the USB specification is ambiguous. It assumes correct software drivers on both sides. NAKing the packet can result in retransmission of the already oversized packet data. Therefore, in response to oversized packets, the USB core continues ACKing the packet for non-isochronous transfers.

Table 45-5. USB Responses to DMA Overrun Errors

Errors due to Memory Latency	Errors due to Oversized Packets
Non-Acknowledgment (NAK) or Bus Timeout (BTO) — See bit 4 in "Error Interrupt Status Register (ERR_STAT)" as appropriate for the class of transaction.	Continues acknowledging (ACKing) the packet for non-isochronous transfers.

Table continues on the next page...

Table 45-5. USB Responses to DMA Overrun Errors (continued)

Errors due to Memory Latency	Errors due to Oversized Packets
—	The data written to memory is clipped to the MaxPacket size so as not to corrupt system memory.
The DMA_ERR bit is set in the ERR_STAT register for host and device modes of operation. Depending on the values of the INT_ENB and ERR_ENB register, the core may assert an interrupt to notify the processor of the DMA error.	Asserts the DMA_ERR bit of the ERR_STAT register (which could trigger an interrupt) and a TOK_DNE interrupt fires. (Note: The TOK_PID field of the BDT is not 1111 because the DMA_ERR is not due to latency).
• For host mode, the TOK_DNE interrupt fires and the TOK_PID field of the BDT is 1111 to indicate the DMA latency error. Host mode software can decide to retry or move to next scheduled item. • In device mode, the BDT is not written back nor is the TOK_DNE interrupt triggered because it is assumed that a second attempt is queued and will succeed in the future.	The packet length field written back to the BDT is the MaxPacket value that represents the length of the clipped data actually written to memory.
From here, the software can decide an appropriate course of action for future transactions such as stalling the endpoint, canceling the transfer, disabling the endpoint, etc.	

45.4 Memory Map/Register Definitions

This section provides the memory map and detailed descriptions of all USB interface registers.

USB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_2000	Peripheral ID Register (USB0_PERID)	8	R	04h	45.4.1/ 1270
4007_2004	Peripheral ID Complement Register (USB0_IDCOMP)	8	R	FBh	45.4.2/ 1270
4007_2008	Peripheral Revision Register (USB0_REV)	8	R	33h	45.4.3/ 1271
4007_200C	Peripheral Additional Info Register (USB0_ADDINFO)	8	R	01h	45.4.4/ 1271
4007_2010	OTG Interrupt Status Register (USB0_OTGISTAT)	8	R/W	00h	45.4.5/ 1272
4007_2014	OTG Interrupt Control Register (USB0_OTGICR)	8	R/W	00h	45.4.6/ 1273
4007_2018	OTG Status Register (USB0_OTGSTAT)	8	R/W	00h	45.4.7/ 1274

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_201C	OTG Control Register (USB0_OTGCTL)	8	R/W	00h	45.4.8/ 1275
4007_2080	Interrupt Status Register (USB0_ISTAT)	8	R/W	00h	45.4.9/ 1276
4007_2084	Interrupt Enable Register (USB0_INTEN)	8	R/W	00h	45.4.10/ 1277
4007_2088	Error Interrupt Status Register (USB0_ERRSTAT)	8	R/W	00h	45.4.11/ 1278
4007_208C	Error Interrupt Enable Register (USB0_ERREN)	8	R/W	00h	45.4.12/ 1279
4007_2090	Status Register (USB0_STAT)	8	R	00h	45.4.13/ 1280
4007_2094	Control Register (USB0_CTL)	8	R/W	00h	45.4.14/ 1281
4007_2098	Address Register (USB0_ADDR)	8	R/W	00h	45.4.15/ 1282
4007_209C	BDT Page Register 1 (USB0_BDTPAGE1)	8	R/W	00h	45.4.16/ 1283
4007_20A0	Frame Number Register Low (USB0_FRMNUML)	8	R/W	00h	45.4.17/ 1283
4007_20A4	Frame Number Register High (USB0_FRMNUMH)	8	R/W	00h	45.4.18/ 1284
4007_20A8	Token Register (USB0_TOKEN)	8	R/W	00h	45.4.19/ 1285
4007_20AC	SOF Threshold Register (USB0_SOFTHLD)	8	R/W	00h	45.4.20/ 1286
4007_20B0	BDT Page Register 2 (USB0_BDTPAGE2)	8	R/W	00h	45.4.21/ 1286
4007_20B4	BDT Page Register 3 (USB0_BDTPAGE3)	8	R/W	00h	45.4.22/ 1287
4007_20C0	Endpoint Control Register (USB0_ENDPT0)	8	R/W	00h	45.4.23/ 1287
4007_20C4	Endpoint Control Register (USB0_ENDPT1)	8	R/W	00h	45.4.23/ 1287
4007_20C8	Endpoint Control Register (USB0_ENDPT2)	8	R/W	00h	45.4.23/ 1287
4007_20CC	Endpoint Control Register (USB0_ENDPT3)	8	R/W	00h	45.4.23/ 1287
4007_20D0	Endpoint Control Register (USB0_ENDPT4)	8	R/W	00h	45.4.23/ 1287

Table continues on the next page...

USB memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4007_20D4	Endpoint Control Register (USB0_ENDPT5)	8	R/W	00h	45.4.23/ 1287
4007_20D8	Endpoint Control Register (USB0_ENDPT6)	8	R/W	00h	45.4.23/ 1287
4007_20DC	Endpoint Control Register (USB0_ENDPT7)	8	R/W	00h	45.4.23/ 1287
4007_20E0	Endpoint Control Register (USB0_ENDPT8)	8	R/W	00h	45.4.23/ 1287
4007_20E4	Endpoint Control Register (USB0_ENDPT9)	8	R/W	00h	45.4.23/ 1287
4007_20E8	Endpoint Control Register (USB0_ENDPT10)	8	R/W	00h	45.4.23/ 1287
4007_20EC	Endpoint Control Register (USB0_ENDPT11)	8	R/W	00h	45.4.23/ 1287
4007_20F0	Endpoint Control Register (USB0_ENDPT12)	8	R/W	00h	45.4.23/ 1287
4007_20F4	Endpoint Control Register (USB0_ENDPT13)	8	R/W	00h	45.4.23/ 1287
4007_20F8	Endpoint Control Register (USB0_ENDPT14)	8	R/W	00h	45.4.23/ 1287
4007_20FC	Endpoint Control Register (USB0_ENDPT15)	8	R/W	00h	45.4.23/ 1287
4007_2100	USB Control Register (USB0_USBCTRL)	8	R/W	C0h	45.4.24/ 1288
4007_2104	USB OTG Observe Register (USB0_OBSERVE)	8	R	50h	45.4.25/ 1289
4007_2108	USB OTG Control Register (USB0_CONTROL)	8	R/W	00h	45.4.26/ 1290
4007_210C	USB Transceiver Control Register 0 (USB0_USBTRC0)	8	R/W	00h	45.4.27/ 1290
4007_2114	Frame Adjust Register (USB0_USBFRMADJUST)	8	R/W	00h	45.4.28/ 1291

45.4.1 Peripheral ID Register (USBx_PERID)

The Peripheral ID Register reads back the value of 0x04. This value is defined for the USB Peripheral.

Addresses: USB0_PERID is 4007_2000h base + 0h offset = 4007_2000h

Bit	7	6	5	4	3	2	1	0
Read	0		ID					
Write	▨		▨					
Reset	0	0	0	0	0	1	0	0

USBx_PERID field descriptions

Field	Description
7–6 Reserved	This read-only field is reserved and always has the value zero.
5–0 ID	Peripheral identification bits These bits always read 0x04 (00_0100)

45.4.2 Peripheral ID Complement Register (USBx_IDCOMP)

The Peripheral ID Complement Register reads back the complement of the Peripheral ID Register. For the USB Peripheral, this is the value 0xFB.

Addresses: USB0_IDCOMP is 4007_2000h base + 4h offset = 4007_2004h

Bit	7	6	5	4	3	2	1	0
Read	1		NID					
Write	▨		▨					
Reset	1	1	1	1	1	0	1	1

USBx_IDCOMP field descriptions

Field	Description
7–6 Reserved	This read-only field is reserved and always has the value one. These bits always read ones
5–0 NID	Ones complement of peripheral identification bits.

45.4.3 Peripheral Revision Register (USBx_REV)

This register contains the revision number of the USB Module.

Addresses: USB0_REV is 4007_2000h base + 8h offset = 4007_2008h

Bit	7	6	5	4	3	2	1	0
Read	REV							
Write								
Reset	0	0	1	1	0	0	1	1

USBx_REV field descriptions

Field	Description
7–0 REV	Revision Indicate the revision number of the USB Core.

45.4.4 Peripheral Additional Info Register (USBx_ADDINFO)

The Peripheral Additional info Register reads back the value of the fixed Interrupt Request Level (IRQNUM) along with the Host Enable bit. If set to 1, the Host Enable bit indicates the USB peripheral is operating in host mode.

Addresses: USB0_ADDINFO is 4007_2000h base + Ch offset = 4007_200Ch

Bit	7	6	5	4	3	2	1	0
Read	IRQNUM					0		IEHOST
Write								
Reset	0	0	0	0	0	0	0	1

USBx_ADDINFO field descriptions

Field	Description
7–3 IRQNUM	Assigned Interrupt Request Number
2–1 Reserved	This read-only field is reserved and always has the value zero.
0 IEHOST	This bit is set if host mode is enabled.

45.4.5 OTG Interrupt Status Register (USBx_OTGISTAT)

The OTG Interrupt Status Register records changes of the ID sense and VBUS signals. Software can read this register to determine which event has caused an interrupt. Only bits that have changed since the last software read are set. Writing a one to a bit clears the associated interrupt.

Addresses: USB0_OTGISTAT is 4007_2000h base + 10h offset = 4007_2010h

Bit	7	6	5	4	3	2	1	0
Read	IDCHG	ONEMSEC	LINE_ STATE_ CHG	0	SESSVLDCHG	B_SESS_CHG	0	AVBUSCHG
Write				▨			▨	
Reset	0	0	0	0	0	0	0	0

USBx_OTGISTAT field descriptions

Field	Description
7 IDCHG	This bit is set when a change in the ID Signal from the USB connector is sensed.
6 ONEMSEC	This bit is set when the 1 millisecond timer expires. This bit stays asserted until cleared by software. The interrupt must be serviced every millisecond to avoid losing 1msec counts.
5 LINE_STATE_ CHG	This bit is set when the USB line state changes. The interrupt associated with this bit can be used to detect Reset, Resume, Connect, and Data Line Pulse signals.
4 Reserved	This read-only field is reserved and always has the value zero.
3 SESSVLDCHG	This bit is set when a change in VBUS is detected indicating a session valid or a session no longer valid.
2 B_SESS_CHG	This bit is set when a change in VBUS is detected on a B device.
1 Reserved	This read-only field is reserved and always has the value zero.
0 AVBUSCHG	This bit is set when a change in VBUS is detected on an A device.

45.4.6 OTG Interrupt Control Register (USBx_OTGICR)

The OTG Interrupt Control Register enables the corresponding interrupt status bits defined in the OTG Interrupt Status Register.

Addresses: USB0_OTGICR is 4007_2000h base + 14h offset = 4007_2014h

Bit	7	6	5	4	3	2	1	0
Read	IDEN	ONEMSECEN	LINESTATEEN	0	SESSVLDEN	BSESSEN	0	AVBUSEN
Write				▨			▨	
Reset	0	0	0	0	0	0	0	0

USBx_OTGICR field descriptions

Field	Description
7 IDEN	ID interrupt enable 0 The ID interrupt is disabled 1 The ID interrupt is enabled
6 ONEMSECEN	1 millisecond interrupt enable 0 The 1msec timer interrupt is disabled. 1 The 1msec timer interrupt is enabled.
5 LINESTATEEN	Line State change interrupt enable 0 The LINE_STAT_CHG interrupt is disabled. 1 The LINE_STAT_CHG interrupt is enabled
4 Reserved	This read-only field is reserved and always has the value zero.
3 SESSVLDEN	Session valid interrupt enable 0 The SESSVLDCHG interrupt is disabled. 1 The SESSVLDCHG interrupt is enabled.
2 BSESSEN	B Session END interrupt enable 0 The B_SESS_CHG interrupt is disabled 1 The B_SESS_CHG interrupt is enabled
1 Reserved	This read-only field is reserved and always has the value zero.
0 AVBUSEN	A VBUS Valid interrupt enable 0 The AVBUSCHG interrupt is disabled 1 The AVBUSCHG interrupt is enabled

45.4.7 OTG Status Register (USBx_OTGSTAT)

The OTG Status Register displays the actual value from the external comparator outputs of the ID pin and VBUS.

Addresses: USB0_OTGSTAT is 4007_2000h base + 18h offset = 4007_2018h

Bit	7	6	5	4	3	2	1	0
Read	ID	ONEMSECEN	LINESTATESTABLE	0	SESS_VLD	BSESSEND	0	AVBUSVLD
Write				▨			▨	
Reset	0	0	0	0	0	0	0	0

USBx_OTGSTAT field descriptions

Field	Description
7 ID	Indicates the current state of the ID pin on the USB connector 0 Indicates a Type A cable has been plugged into the USB connector 1 Indicates no cable is attached or a Type B cable has been plugged into the USB connector
6 ONEMSECEN	This bit is reserved for the 1msec count, but it is not useful to software.
5 LINESTATESTABLE	This bit indicates that the internal signals that control the LINE_STATE_CHG bit (bit 5) of the OTGISTAT register have been stable for at least 1 millisecond. First read the LINE_STATE_CHG bit, and then read this bit. If this bit reads as 1, then the value of LINE_STATE_CHG can be considered stable. 0 The LINE_STAT_CHG bit is not yet stable. 1 The LINE_STAT_CHG bit has been debounced and is stable.
4 Reserved	This read-only field is reserved and always has the value zero.
3 SESS_VLD	Session valid 0 The VBUS voltage is below the B session Valid threshold 1 The VBUS voltage is above the B session Valid threshold.
2 BSESSEND	B Session END 0 The VBUS voltage is above the B session End threshold. 1 The VBUS voltage is below the B session End threshold.
1 Reserved	This read-only field is reserved and always has the value zero.
0 AVBUSVLD	A VBUS Valid 0 The VBUS voltage is below the A VBUS Valid threshold. 1 The VBUS voltage is above the A VBUS Valid threshold.

45.4.8 OTG Control Register (USBx_OTGCTL)

The OTG Control Register controls the operation of VBUS and Data Line termination resistors.

Addresses: USB0_OTGCTL is 4007_2000h base + 1Ch offset = 4007_201Ch

Bit	7	6	5	4	3	2	1	0
Read	DPHIGH	0	DPLOW	DMLOW	0	OTGEN	0	
Write		▨			▨		▨	
Reset	0	0	0	0	0	0	0	0

USBx_OTGCTL field descriptions

Field	Description
7 DPHIGH	D+ Data Line pullup resistor enable 0 D+ pullup resistor is not enabled 1 D+ pullup resistor is enabled
6 Reserved	This read-only field is reserved and always has the value zero.
5 DPLOW	D+ Data Line pull-down resistor enable This bit should always be enabled together with bit 4 (DMLOW) 0 D+ pulldown resistor is not enabled. 1 D+ pulldown resistor is enabled.
4 DMLOW	D- Data Line pull-down resistor enable 0 D- pulldown resistor is not enabled. 1 D- pulldown resistor is enabled.
3 Reserved	This read-only field is reserved and always has the value zero.
2 OTGEN	On-The-Go pullup/pulldown resistor enable 0 If USB_EN is set and HOST_MODE is clear in the Control Register (CTL), then the D+ Data Line pull-up resistors are enabled. If HOST_MODE is set the D+ and D- Data Line pull-down resistors are engaged. 1 The pull-up and pull-down controls in this register are used.
1–0 Reserved	This read-only field is reserved and always has the value zero.

45.4.9 Interrupt Status Register (USBx_ISTAT)

The Interrupt Status Register contains bits for each of the interrupt sources within the USB Module. Each of these bits are qualified with their respective interrupt enable bits. All bits of this register are logically OR'd together along with the OTG Interrupt Status Register (OTGSTAT) to form a single interrupt source for the processor's interrupt controller. After an interrupt bit has been set it may only be cleared by writing a one to the respective interrupt bit. This register contains the value of 0x00 after a reset.

Addresses: USB0_ISTAT is 4007_2000h base + 80h offset = 4007_2080h

Bit	7	6	5	4	3	2	1	0
Read	STALL	ATTACH	RESUME	SLEEP	TOKDNE	SOFTOK	ERROR	USBRST
Write	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

USBx_ISTAT field descriptions

Field	Description
7 STALL	Stall Interrupt In Target mode this bit is asserted when a STALL handshake is sent by the SIE. In Host mode this bit is set when the USB Module detects a STALL acknowledge during the handshake phase of a USB transaction. This interrupt can be use to determine is the last USB transaction was completed successfully or if it stalled.
6 ATTACH	Attach Interrupt This bit is set when the USB Module detects an attach of a USB device. This signal is only valid if HOSTMODEEN is true. This interrupt signifies that a peripheral is now present and must be configured.
5 RESUME	This bit is set depending upon the DP/DM signals, and can be used to signal remote wake-up signaling on the USB bus. When not in suspend mode this interrupt should be disabled.
4 SLEEP	This bit is set when the USB Module detects a constant idle on the USB bus for 3 milliseconds. The sleep timer is reset by activity on the USB bus.
3 TOKDNE	This bit is set when the current token being processed has completed. The processor should immediately read the STAT register to determine the EndPoint and BD used for this token. Clearing this bit (by writing a one) causes the STAT register to be cleared or the STAT holding register to be loaded into the STAT register.
2 SOFTOK	This bit is set when the USB Module receives a Start Of Frame (SOF) token. In Host mode this bit is set when the SOF threshold is reached, so that software can prepare for the next SOF.
1 ERROR	This bit is set when any of the error conditions within the ERRSTAT register occur. The processor must then read the ERRSTAT register to determine the source of the error.
0 USBRST	This bit is set when the USB Module has decoded a valid USB reset. This informs the Microprocessor that it should write 0x00 into the address register and enable endpoint 0. USBRST is set after a USB reset has been detected for 2.5 microseconds. It is not asserted again until the USB reset condition has been removed and then reasserted.

45.4.10 Interrupt Enable Register (USBx_INTEN)

The Interrupt Enable Register contains enable bits for each of the interrupt sources within the USB Module. Setting any of these bits enables the respective interrupt source in the ISTAT register. This register contains the value of 0x00 after a reset.

Addresses: USB0_INTEN is 4007_2000h base + 84h offset = 4007_2084h

Bit	7	6	5	4	3	2	1	0
Read Write	STALLEN	ATTACHEN	RESUMEEN	SLEEPEN	TOKDNEEN	SOFTOKEN	ERROREN	USBRSTEN
Reset	0	0	0	0	0	0	0	0

USBx_INTEN field descriptions

Field	Description
7 STALLEN	STALL Interrupt Enable 0 The STALL interrupt is not enabled. 1 The STALL interrupt is enabled.
6 ATTACHEN	ATTACH Interrupt Enable 0 The ATTACH interrupt is not enabled. 1 The ATTACH interrupt is enabled.
5 RESUMEEN	RESUME Interrupt Enable 0 The RESUME interrupt is not enabled. 1 The RESUME interrupt is enabled.
4 SLEEPEN	SLEEP Interrupt Enable 0 The SLEEP interrupt is not enabled. 1 The SLEEP interrupt is enabled.
3 TOKDNEEN	TOKDNE Interrupt Enable 0 The TOKDNE interrupt is not enabled. 1 The TOKDNE interrupt is enabled.
2 SOFTOKEN	SOFTOK Interrupt Enable 0 The SOFTOK interrupt is not enabled. 1 The SOFTOK interrupt is enabled.
1 ERROREN	ERROR Interrupt Enable 0 The ERROR interrupt is not enabled. 1 The ERROR interrupt is enabled.
0 USBRSTEN	USBRST Interrupt Enable 0 The USBRST interrupt is not enabled. 1 The USBRST interrupt is enabled.

45.4.11 Error Interrupt Status Register (USBx_ERRSTAT)

The Error Interrupt Status Register contains enable bits for each of the error sources within the USB Module. Each of these bits are qualified with their respective error enable bits. All bits of this Register are logically OR'd together and the result placed in the ERROR bit of the ISTAT register. After an interrupt bit has been set it may only be cleared by writing a one to the respective interrupt bit. Each bit is set as soon as the error conditions is detected. Therefore, the interrupt does not typically correspond with the end of a token being processed. This register contains the value of 0x00 after a reset.

Addresses: USB0_ERRSTAT is 4007_2000h base + 88h offset = 4007_2088h

Bit	7	6	5	4	3	2	1	0
Read	BTSERR	0	DMAERR	BTOERR	DFN8	CRC16	CRC5EOF	PIDERR
Write	w1c		w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

USBx_ERRSTAT field descriptions

Field	Description
7 BTSERR	This bit is set when a bit stuff error is detected. If set, the corresponding packet is rejected due to the error.
6 Reserved	This read-only field is reserved and always has the value zero.
5 DMAERR	This bit is set if the USB Module has requested a DMA access to read a new BDT but has not been given the bus before it needs to receive or transmit data. If processing a TX transfer this would cause a transmit data underflow condition. If processing a RX transfer this would cause a receive data overflow condition. This interrupt is useful when developing device arbitration hardware for the microprocessor and the USB Module to minimize bus request and bus grant latency. This bit is also set if a data packet to or from the host is larger than the buffer size allocated in the BDT. In this case the data packet is truncated as it is put into buffer memory.
4 BTOERR	This bit is set when a bus turnaround timeout error occurs. The USB Module contains a bus turnaround timer that keeps track of the amount of time elapsed between the token and data phases of a SETUP or OUT TOKEN or the data and handshake phases of a IN TOKEN. If more than 16 bit times are counted from the previous EOP before a transition from IDLE, a bus turnaround timeout error occurs.
3 DFN8	This bit is set if the data field received was not 8 bits in length. USB Specification 1.0 requires that data fields be an integral number of bytes. If the data field was not an integral number of bytes, this bit is set.
2 CRC16	This bit is set when a data packet is rejected due to a CRC16 error.
1 CRC5EOF	This error interrupt has two functions. When the USB Module is operating in peripheral mode (HOSTMODEEN=0), this interrupt detects CRC5 errors in the token packets generated by the host. If set the token packet was rejected due to a CRC5 error. When the USB Module is operating in host mode (HOSTMODEEN=1), this interrupt detects End Of Frame (EOF) error conditions. This occurs when the USB Module is transmitting or receiving data and the SOF counter reaches zero. This interrupt is useful when developing USB packet scheduling software to ensure that no USB transactions cross the start of the next frame.

Table continues on the next page...

USBx_ERRSTAT field descriptions (continued)

Field	Description
0 PIDERR	This bit is set when the PID check field fails.

45.4.12 Error Interrupt Enable Register (USBx_ERREN)

The Error Interrupt Enable Register contains enable bits for each of the error interrupt sources within the USB Module. Setting any of these bits enables the respective interrupt source in the ERRSTAT register. Each bit is set as soon as the error conditions is detected. Therefore, the interrupt does not typically correspond with the end of a token being processed. This register contains the value of 0x00 after a reset.

Addresses: USB0_ERREN is 4007_2000h base + 8Ch offset = 4007_208Ch

Bit	7	6	5	4	3	2	1	0
Read	BTSERREN	0	DMAERREN	BTOERREN	DFN8EN	CRC16EN	CRC5EOFEN	PIDERREN
Write								
Reset	0	0	0	0	0	0	0	0

USBx_ERREN field descriptions

Field	Description
7 BTSERREN	BTSERR Interrupt Enable 0 The BTSERR interrupt is not enabled. 1 The BTSERR interrupt is enabled.
6 Reserved	This read-only field is reserved and always has the value zero.
5 DMAERREN	DMAERR Interrupt Enable 0 The DMAERR interrupt is not enabled. 1 The DMAERR interrupt is enabled.
4 BTOERREN	BTOERR Interrupt Enable 0 The BTOERR interrupt is not enabled. 1 The BTOERR interrupt is enabled.
3 DFN8EN	DFN8 Interrupt Enable 0 The DFN8 interrupt is not enabled. 1 The DFN8 interrupt is enabled.
2 CRC16EN	CRC16 Interrupt Enable 0 The CRC16 interrupt is not enabled. 1 The CRC16 interrupt is enabled.

Table continues on the next page...

USBx_ERREN field descriptions (continued)

Field	Description
1 CRC5EOFEN	CRC5/EOF Interrupt Enable 0 The CRC5/EOF interrupt is not enabled. 1 The CRC5/EOF interrupt is enabled.
0 PIDERREN	PIDERR Interrupt Enable 0 The PIDERR interrupt is not enabled. 1 The PIDERR interrupt is enabled.

45.4.13 Status Register (USBx_STAT)

The Status Register reports the transaction status within the USB Module. When the processor's interrupt controller has received a TOKDNE interrupt the Status Register should be read to determine the status of the previous endpoint communication. The data in the status register is valid when the TOKDNE interrupt bit is asserted. The STAT register is actually a read window into a status FIFO maintained by the USB Module. When the USB Module uses a BD, it updates the Status Register. If another USB transaction is performed before the TOKDNE interrupt is serviced, the USB Module stores the status of the next transaction in the STAT FIFO. Thus the STAT register is actually a four byte FIFO that allows the processor core to process one transaction while the SIE is processing the next transaction. Clearing the TOKDNE bit in the ISTAT register causes the SIE to update the STAT register with the contents of the next STAT value. If the data in the STAT holding register is valid, the SIE immediately reasserts to TOKDNE interrupt.

Addresses: USB0_STAT is 4007_2000h base + 90h offset = 4007_2090h

Bit	7	6	5	4	3	2	1	0
Read	ENDP				TX	ODD	0	
Write								
Reset	0	0	0	0	0	0	0	0

USBx_STAT field descriptions

Field	Description
7–4 ENDP	This four-bit field encodes the endpoint address that received or transmitted the previous token. This allows the processor core to determine which BDT entry was updated by the last USB transaction.
3 TX	Transmit Indicator 0 The most recent transaction was a Receive operation. 1 The most recent transaction was a Transmit operation.

Table continues on the next page...

USBx_STAT field descriptions (continued)

Field	Description
2 ODD	this bit is set if the last Buffer Descriptor updated was in the odd bank of the BDT.
1–0 Reserved	This read-only field is reserved and always has the value zero.

45.4.14 Control Register (USBx_CTL)

The Control Register provides various control and configuration information for the USB Module.

Addresses: USB0_CTL is 4007_2000h base + 94h offset = 4007_2094h

Bit	7	6	5	4	3	2	1	0
Read Write	JSTATE	SE0	TXSUSPENDTOKENBUSY	RESET	HOSTMODEEN	RESUME	ODDRST	USBENSOFEN
Reset	0	0	0	0	0	0	0	0

USBx_CTL field descriptions

Field	Description
7 JSTATE	Live USB differential receiver JSTATE signal The polarity of this signal is affected by the current state of LSEN .
6 SE0	Live USB Single Ended Zero signal
5 TXSUSPENDTOKENBUSY	When the USB Module is in Host mode TOKEN_BUSY is set when the USB Module is busy executing a USB token and no more token commands should be written to the Token Register. Software should check this bit before writing any tokens to the Token Register to ensure that token commands are not lost. In Target mode TXD_SUSPEND is set when the SIE has disabled packet transmission and reception. Clearing this bit allows the SIE to continue token processing. This bit is set by the SIE when a Setup Token is received allowing software to dequeue any pending packet transactions in the BDT before resuming token processing.
4 RESET	Setting this bit enables the USB Module to generate USB reset signaling. This allows the USB Module to reset USB peripherals. This control signal is only valid in Host mode (HOSTMODEEN=1). Software must set RESET to 1 for the required amount of time and then clear it to 0 to end reset signaling. For more information on RESET signaling see Section 7.1.4.3 of the USB specification version 1.0.
3 HOSTMODEEN	When set to 1, this bit enables the USB Module to operate in Host mode. In host mode, the USB module performs USB transactions under the programmed control of the host processor.

Table continues on the next page...

USBx_CTL field descriptions (continued)

Field	Description
2 RESUME	When set to 1 this bit enables the USB Module to execute resume signaling. This allows the USB Module to perform remote wake-up. Software must set RESUME to 1 for the required amount of time and then clear it to 0. If the HOSTMODEEN bit is set, the USB module appends a Low Speed End of Packet to the Resume signaling when the RESUME bit is cleared. For more information on RESUME signaling see Section 7.1.4.5 of the USB specification version 1.0.
1 ODDRST	Setting this bit to 1 resets all the BDT ODD ping/pong bits to 0, which then specifies the EVEN BDT bank.
0 USBENSOFEN	USB Enable Setting this bit causes the SIE to reset all of its ODD bits to the BDTs. Therefore, setting this bit resets much of the logic in the SIE. When host mode is enabled, clearing this bit causes the SIE to stop sending SOF tokens. 0 The USB Module is disabled. 1 The USB Module is enabled.

45.4.15 Address Register (USBx_ADDR)

The Address Register holds the unique USB address that the USB Module decodes when in Peripheral mode (HOSTMODEEN=0). When operating in Host mode (HOSTMODEEN=1) the USB Module transmits this address with a TOKEN packet. This enables the USB Module to uniquely address an USB peripheral. In either mode, the USB_EN bit within the control register must be set. The Address Register is reset to 0x00 after the reset input becomes active or the USB Module decodes a USB reset signal. This action initializes the Address Register to decode address 0x00 as required by the USB specification.

Addresses: USB0_ADDR is 4007_2000h base + 98h offset = 4007_2098h

Bit	7	6	5	4	3	2	1	0
Read Write	LSEN	ADDR						
Reset	0	0	0	0	0	0	0	0

USBx_ADDR field descriptions

Field	Description
7 LSEN	Low Speed Enable bit This bit informs the USB Module that the next token command written to the token register must be performed at low speed. This enables the USB Module to perform the necessary preamble required for low-speed data transmissions.

Table continues on the next page...

USBx_ADDR field descriptions (continued)

Field	Description
6–0 ADDR	USB address This 7-bit value defines the USB address that the USB Module decodes in peripheral mode, or transmit when in host mode.

45.4.16 BDT Page Register 1 (USBx_BDTPAGE1)

The Buffer Descriptor Table Page Register 1 provides address bits 15 through 9 of the base address where the current Buffer Descriptor Table (BDT) resides in system memory. The 32-bit BDT Base Address is always aligned on 512-byte boundaries, so bits 8 through 0 of the base address are always taken as zero.

Addresses: USB0_BDTPAGE1 is 4007_2000h base + 9Ch offset = 4007_209Ch

Bit	7	6	5	4	3	2	1	0
Read	BDTBA							0
Write								
Reset	0	0	0	0	0	0	0	0

USBx_BDTPAGE1 field descriptions

Field	Description
7–1 BDTBA	This field provides address bits 15 through 9 of the BDT base address.
0 Reserved	This read-only field is reserved and always has the value zero.

45.4.17 Frame Number Register Low (USBx_FRMNUML)

The Frame Number Register (Low and High) contains an 11-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

Addresses: USB0_FRMNUML is 4007_2000h base + A0h offset = 4007_20A0h

Bit	7	6	5	4	3	2	1	0
Read	FRM[7:0]							
Write								
Reset	0	0	0	0	0	0	0	0

USBx_FRMNUML field descriptions

Field	Description
7–0 FRM[7:0]	This 8-bit field and the 3-bit field in the Frame Number Register High are used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

45.4.18 Frame Number Register High (USBx_FRMNUMH)

The Frame Number Register (Low and High) contains an 11-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

Addresses: USB0_FRMNUMH is 4007_2000h base + A4h offset = 4007_20A4h

Bit	7	6	5	4	3	2	1	0
Read	0					FRM[10:8]		
Write								
Reset	0	0	0	0	0	0	0	0

USBx_FRMNUMH field descriptions

Field	Description
7–3 Reserved	This read-only field is reserved and always has the value zero.
2–0 FRM[10:8]	This 3-bit field and the 8-bit field in the Frame Number Register Low are used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

45.4.19 Token Register (USBx_TOKEN)

The Token Register is used to perform USB transactions when in host mode (HOSTMODEEN=1). When the processor core wishes to execute a USB transaction to a peripheral, it writes the TOKEN type and endpoint to this register. After this register has been written, the USB module begins the specified USB transaction to the address contained in the address register. The processor core should always check that the TOKEN_BUSY bit in the control register is not set before performing a write to the Token Register. This ensures token commands are not overwritten before they can be executed. The address register and endpoint control register 0 are also used when performing a token command and therefore must also be written before the Token Register. The address register is used to correctly select the USB peripheral address transmitted by the token command. The endpoint control register determines the handshake and retry policies used during the transfer.

Addresses: USB0_TOKEN is 4007_2000h base + A8h offset = 4007_20A8h

Bit	7	6	5	4	3	2	1	0
Read Write	TOKENPID				TOKENENDPT			
Reset	0	0	0	0	0	0	0	0

USBx_TOKEN field descriptions

Field	Description
7–4 TOKENPID	This 4-bit field contains the token type executed by the USB Module. 0001 OUT Token. USB Module performs an OUT (TX) transaction. 1001 IN Token. USB Module performs an In (RX) transaction. 1101 SETUP Token. USB Module performs a SETUP (TX) transaction
3–0 TOKENENDPT	This 4 bit field holds the Endpoint address for the token command. The four bit value written must be a valid endpoint.

45.4.20 SOF Threshold Register (USBx_SOFTHLD)

The SOF Threshold Register is used only in Hosts mode (HOSTMODEEN=1). When in Host mode, the 14-bit SOF counter counts the interval between SOF frames. The SOF must be transmitted every 1msec so the SOF counter is loaded with a value of 12000. When the SOF counter reaches zero, a Start Of Frame (SOF) token is transmitted. The SOF threshold register is used to program the number of USB byte times before the SOF to stop initiating token packet transactions. This register must be set to a value that ensures that other packets are not actively being transmitted when the SOF time counts to zero. When the SOF counter reaches the threshold value, no more tokens are transmitted until after the SOF ha been transmitted. The value programmed into the threshold register must reserve enough time to ensure the worst case transaction completes. In general the worst case transaction is a IN token followed by a data packet from the target followed by the response from the host. The actual time required is a function of the maximum packet size on the bus. Typical values for the SOF threshold are: 64-byte packets=74; 32-byte packets=42; 16-byte packets=26; 8-byte packets=18.

Addresses: USB0_SOFTHLD is 4007_2000h base + ACh offset = 4007_20ACh

Bit	7	6	5	4	3	2	1	0
Read Write	CNT							
Reset	0	0	0	0	0	0	0	0

USBx_SOFTHLD field descriptions

Field	Description
7–0 CNT	This 8-bit field represents the SOF count threshold in byte times.

45.4.21 BDT Page Register 2 (USBx_BDTPAGE2)

The Buffer Descriptor Table Page Register 2 contains an 8-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

Addresses: USB0_BDTPAGE2 is 4007_2000h base + B0h offset = 4007_20B0h

Bit	7	6	5	4	3	2	1	0
Read Write	BDTBA							
Reset	0	0	0	0	0	0	0	0

USBx_BDTPAGE2 field descriptions

Field	Description
7–0 BDTBA	This 8 bit field provides address bits 23 through 16 of the BDT base address, which defines where the Buffer Descriptor Table resides in system memory.

45.4.22 BDT Page Register 3 (USBx_BDTPAGE3)

The Buffer Descriptor Table Page Register 3 contains an 8-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

Addresses: USB0_BDTPAGE3 is 4007_2000h base + B4h offset = 4007_20B4h

Bit	7	6	5	4	3	2	1	0
Read Write	BDTBA							
Reset	0	0	0	0	0	0	0	0

USBx_BDTPAGE3 field descriptions

Field	Description
7–0 BDTBA	This 8 bit field provides address bits 31 through 24 of the BDT base address, which defines where the Buffer Descriptor Table resides in system memory.

45.4.23 Endpoint Control Register (USBx_ENDPT*n*)

The Endpoint Control Registers contain the endpoint control bits for each of the 16 endpoints available within the USB Module for a decoded address. The format for these registers is shown in the following figure. Endpoint 0 (ENDPT0) is associated with control pipe 0, which is required for all USB functions. Therefore, after a USBRST interrupt occurs the processor core should set the ENDPT0 register to contain 0x0D.

In Host mode ENDPT0 is used to determine the handshake, retry and low speed characteristics of the host transfer. For Host mode control, bulk and interrupt transfers the EPHSHK bit should be set to 1. For Isochronous transfers it should be set to 0. Common values to use for ENDPT0 in host mode are 0x4D for Control, Bulk, and Interrupt transfers, and 0x4C for Isochronous transfers.

Addresses: 4007_2000h base + C0h offset + (4d × *n*), where *n* = 0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	HOSTWOHUB	RETRYDIS	0	EPCTLDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK
Write			▨					
Reset	0	0	0	0	0	0	0	0

USBx_ENDPT*n* field descriptions

Field	Description
7 HOSTWOHUB	This is a Host mode only bit and is only present in the control register for endpoint 0 (ENDPT0). When set this bit allows the host to communicate to a directly connected low speed device. When cleared, the host produces the PRE_PID then switch to low speed signaling when sending a token to a low speed device as required to communicate with a low speed device through a hub.
6 RETRYDIS	This is a Host mode only bit and is only present in the control register for endpoint 0 (ENDPT0). When set this bit causes the host to not retry NAK'ed (Negative Acknowledgement) transactions. When a transaction is NAKed, the BDT PID field is updated with the NAK PID, and the TOKEN_DNE interrupt is set. When this bit is cleared NAKed transactions is retried in hardware. This bit must be set when the host is attempting to poll an interrupt endpoint.
5 Reserved	This read-only field is reserved and always has the value zero.
4 EPCTLDIS	This bit, when set, disables control (SETUP) transfers. When cleared, control transfers are enabled. This applies if and only if the EPRXEN and EPTXEN bits are also set.
3 EPRXEN	This bit, when set, enables the endpoint for RX transfers.
2 EPTXEN	This bit, when set, enables the endpoint for TX transfers.
1 EPSTALL	When set this bit indicates that the endpoint is called. This bit has priority over all other control bits in the EndPoint Enable Register, but it is only valid if EPTXEN=1 or EPRXEN=1. Any access to this endpoint causes the USB Module to return a STALL handshake. After an endpoint is stalled it requires intervention from the Host Controller.
0 EPHSHK	When set this bet enables an endpoint to perform handshaking during a transaction to this endpoint. This bit is generally set unless the endpoint is Isochronous.

45.4.24 USB Control Register (USBx_USBCTRL)

Addresses: USB0_USBCTRL is 4007_2000h base + 100h offset = 4007_2100h

Bit	7	6	5	4	3	2	1	0
Read	SUSP	PDE	0					
Write								
Reset	1	1	0	0	0	0	0	0

USBx_USBCTRL field descriptions

Field	Description
7 SUSP	Places the USB transceiver into the suspend state. 0 USB transceiver is not in suspend state. 1 USB transceiver is in suspend state.
6 PDE	Enables the weak pulldowns on the USB transceiver. 0 Weak pulldowns are disabled on D+ and D- 1 Weak pulldowns are enabled on D+ and D-.

Table continues on the next page...

USBx_USBCTRL field descriptions (continued)

Field	Description
5–0 Reserved	This read-only field is reserved and always has the value zero.

45.4.25 USB OTG Observe Register (USBx_OBSERVE)

Provides visibility on the state of the pull-ups and pull-downs at the transceiver. Useful when interfacing to an external OTG control module via a serial interface.

Addresses: USB0_OBSERVE is 4007_2000h base + 104h offset = 4007_2104h

Bit	7	6	5	4	3	2	1	0
Read	DPPU	DPPD	0	DMPD	0			0
Write								
Reset	0	1	0	1	0	0	0	0

USBx_OBSERVE field descriptions

Field	Description
7 DPPU	Provides observability of the D+ Pull Up enable at the USB transceiver. 0 D+ pullup disabled. 1 D+ pullup enabled.
6 DPPD	Provides observability of the D+ Pull Down enable at the USB transceiver. 0 D+ pulldown disabled. 1 D+ pulldown enabled.
5 Reserved	This read-only field is reserved and always has the value zero.
4 DMPD	Provides observability of the D- Pull Down enable at the USB transceiver. 0 D- pulldown disabled. 1 D- pulldown enabled.
3–1 Reserved	This read-only field is reserved and always has the value zero.
0 Reserved	This read-only field is reserved and always has the value zero.

45.4.26 USB OTG Control Register (USBx_CONTROL)

Addresses: USB0_CONTROL is 4007_2000h base + 108h offset = 4007_2108h

Bit	7	6	5	4	3	2	1	0
Read	0			DPPULLUPNONOTG	0			
Write								
Reset	0	0	0	0	0	0	0	0

USBx_CONTROL field descriptions

Field	Description
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 DPPULLUPNONOTG	Provides control of the DP PULLUP in the USB OTG module, if USB is configured in non-OTG device mode. 0 DP Pull up in non-OTG device mode is not enabled. 1 DP Pull up in non-OTG device mode is enabled.
3–0 Reserved	This read-only field is reserved and always has the value zero.

45.4.27 USB Transceiver Control Register 0 (USBx_USBTRC0)

Addresses: USB0_USBTRC0 is 4007_2000h base + 10Ch offset = 4007_210Ch

Bit	7	6	5	4	3	2	1	0
Read		0	USBRESMEN	0			SYNC_DET	USB_ RESUME_ INT
Write	USBRESET							
Reset	0	0	0	0	0	0	0	0

USBx_USBTRC0 field descriptions

Field	Description
7 USBRESET	USB reset Generates a hard reset to the USB_OTG module. After this bit is set and the reset occurs, this bit is automatically cleared. **NOTE: It is always read as zero.**

Table continues on the next page...

USBx_USBTRC0 field descriptions (continued)

Field	Description
	0 Normal USB module operation. 1 Returns the USB module to its reset state.
6 Reserved	This read-only field is reserved and always has the value zero.
5 USBRESMEN	Asynchronous Resume Interrupt Enable This bit, when set, allows the USB module to send an asynchronous wakeup event to the MCU upon detection of resume signaling on the USB bus. The MCU then re-enables clocks to the USB module. It is used for low-power suspend mode when USB module clocks are stopped or the USB transceiver is in Suspend mode. Async wakeup only works in device mode. 0 USB asynchronous wakeup from suspend mode disabled. 1 USB asynchronous wakeup from suspend mode enabled. The asynchronous resume interrupt differs from the synchronous resume interrupt in that it asynchronously detects K-state using the unfiltered state of the D+ and D- pins. This interupt should only be enabled when the Transceiver is suspended.
4–2 Reserved	This read-only field is reserved and always has the value zero.
1 SYNC_DET	Synchronous USB Interrupt Detect 0 Synchronous interrupt has not been detected. 1 Synchronous interrupt has been detected.
0 USB_RESUME_ INT	USB Asynchronous Interrupt 0 No interrupt was generated. 1 Interrupt was generated because of the USB asynchronous interrupt.

45.4.28 Frame Adjust Register (USBx_USBFRMADJUST)

Addresses: USB0_USBFRMADJUST is 4007_2000h base + 114h offset = 4007_2114h

Bit	7	6	5	4	3	2	1	0
Read Write	ADJ							
Reset	0	0	0	0	0	0	0	0

USBx_USBFRMADJUST field descriptions

Field	Description
7–0 ADJ	Frame Adjustment In Host mode, the frame adjustment is a twos complement number that adjusts the period of each USB frame in 12-MHz clock periods. A SOF is normally generated every 12,000 12-MHz clock cycles. The Frame Adjust Register can adjust this by -128 to +127 to compensate for inaccuracies in the USB 48-MHz clock. Changes to the ADJ bit take effect at the next start of the next frame.

45.5 OTG and Host Mode Operation

The Host Mode logic allows devices such as digital cameras and palmtop computers to function as a USB Host Controller. The OTG logic adds an interface to allow the OTG Host Negotiation and Session Request Protocols (HNP and SRP) to be implemented in software. Host Mode allows a peripheral such as a digital camera to be connected directly to a USB compliant printer. Digital photos can then be easily printed without having to upload them to a PC. In the palmtop computer application, a USB compliant keyboard/ mouse can be connected to the palmtop computer with the obvious advantages of easier interaction.

Host mode is intended for use in handheld-portable devices to allow easy connection to simple HID class devices such as printers and keyboards. It is NOT intended to perform the functions of a full OHCI or UHCI compatible host controller found on PC motherboards. The USB-FS is not supported by Windows 98 as a USB host controller. Host mode allows bulk, Isochronous, interrupt and control transfers. Bulk data transfers are performed at nearly the full USB bus bandwidth. Support is provided for ISO transfers, but the number of ISO streams that can be practically supported is affected by the interrupt latency of the processor servicing the token during interrupts from the SIE. Custom drivers must be written to support Host mode operation.

Setting the HOST_MODE_EN bit in the CTL register enables host Mode. The USB-FS core can only operate as a peripheral device or in Host Mode. It cannot operate in both modes simultaneously. When HOST_MODE is enabled, only endpoint zero is used. All other endpoints should be disabled by software.

45.6 Host Mode Operation Examples

The following sections illustrate the steps required to perform USB host functions using the USB-FS core. While it is useful to understand the interaction of the hardware and the software at a detailed level, an understanding of the interactions at this level is not required to write host applications using the API software.

To enable host mode and discover a connected device:

1. Enable Host Mode (CTL[HOST_MODE_EN]=1). The pull-down resistors are enabled, and pull-up disabled. Start of Frame (SOF) generation begins. SOF counter loaded with 12,000. Disable SOF packet generation to eliminate noise on the USB by writing the USB enable bit to 0 (CTL[USB_EN]=0).

2. Enable the ATTACH interrupt (INT_ENB[ATTACH]=1).
3. Wait for ATTACH interrupt (INT_STAT[ATTACH]). Signaled by USB Target pull-up resistor changing the state of DPLUS or DMINUS from 0 to 1 (SE0 to J or K state).
4. Check the state of the JSTATE and SE0 bits in the control register. If the connecting device is low speed (JSTATE bit is 0), set the low-speed bit in the address registers (ADDR[LS_EN]=1) and the host without hub bit in endpoint 0 register control (EP_CTL0[HOST_WO_HUB]=1).
5. Enable RESET (CTL[RESET]=1) for 10 ms.
6. Enable SOF packet to keep the connected device from going to suspend (CTL[USB_EN=1]).
7. Start enumeration by sending a sequence of device framework commands, device frame work packets to the default control pipe of the connected device. Refer to the *Universal Serial Bus Revision 2.0 specification,* "Chapter 9 USB Device Framework" (http://www.usb.org/developers/docs).

To complete a control transaction to a connected device:

1. Complete all steps discover a connected device
2. Set up the endpoint control register for bidirectional control transfers EP_CTL0[4:0] = 0x0d.
3. Place a copy of the device framework setup command in a memory buffer. Refer to the *Universal Serial Bus Revision 2.0 specification,* "Chapter 9 USB Device Framework" (http://www.usb.org/developers/docs).
4. Initialize current (even or odd) TX EP0 BDT to transfer the 8 bytes of command data for a device framework command (for example, a GET DEVICE DESCRIPTOR).
   - Set the BDT command word to 0x00080080 –Byte count to 8, own bit to 1.
   - Set the BDT buffer address field to the start address of the 8 byte command buffer.
5. Set the USB device address of the target device in the address register (ADDR[6:0]). After the USB bus reset, the device USB address is zero. It is set to some other value (usually 1) by the Set Address device framework command.
6. Write the token register with a SETUP to Endpoint 0 the target device default control pipe (TOKEN=0xD0). This initiates a setup token on the bus followed by a data packet. The device handshake is returned in the BDT PID field after the packets

complete. When the BDT is written a token done (INT_STAT[TOK_DNE]) interrupt is asserted. This completes the setup phase of the setup transaction. Refer to the *Universal Serial Bus Revision 2.0 specification,* "Chapter 9 USB Device Framework" (http://www.usb.org/developers/docs).

7. To initiate the data phase of the setup transaction (for example, get the data for the GET DEVICE descriptor command) set up a buffer in memory for the data to be transferred.
8. Initialize the current (even or odd) TX EP0 BDT to transfer the data.
    - Set the BDT command word to 0x004000C0 –Byte count to the length of the data buffer in this case 64, own bit to 1, Data toggle to Data1.
    - Set the BDT buffer address field to the start address of the data buffer
9. Write the token register with a IN or OUT token to Endpoint 0 the target device default control pipe, an IN token for a GET DEVICE DESCRIPTOR command (TOKEN=0x90). This initiates an IN token on the bus followed by a data packet from the device to the host. When the data packet completes the BDT is written and a token done (INT_STAT[TOK_DNE]) interrupt is asserted. For control transfers with a single packet data phase this completes the data phase of the setup transaction. Refer to the *Universal Serial Bus Revision 2.0 specification,* "Chapter 9 USB Device Framework" (http://www.usb.org/developers/docs).
10. To initiate the Status phase of the setup transaction set up a buffer in memory to receive or send the zero length status phase data packet.
11. Initialize the current (even or odd) TX EP0 BDT to transfer the status data.
    - Set the BDT command word to 0x00000080 –Byte count to the length of the data buffer in this case 0, own bit to 1, Data toggle to Data0.
    - Set the BDT buffer address field to the start address of the data buffer
12. Write the token register with a IN or OUT token to Endpoint 0 the target device default control pipe, an OUT token for a GET DEVICE DESCRIPTOR command (TOKEN=0x10). This initiates an OUT token on the bus followed by a zero length data packet from the host to the device. When the data packet completes the BDT is written with the handshake form the device and a token done (INT_STAT[TOK_DNE]) interrupt is asserted. This completes the data phase of the setup transaction. Refer to the *Universal Serial Bus Revision 2.0 specification,* "Chapter 9 USB Device Framework" (http://www.usb.org/developers/docs).

To send a Full speed bulk data transfer to a target device:

1. Complete all steps discover a connected device and to configure a connected device. Write the ADDR register with the address of the target device. Typically, there is only one other device on the USB bus in host mode so it is expected that the address is 0x01 and should remain constant.
2. Write the ENDPT0 to 0x1D register to enable transmit and receive transfers with handshaking enabled.
3. Setup the Even TX EP0 BDT to transfer up to 64 bytes.
4. Set the USB device address of the target device in the address register (ADDR[6:0]).
5. Write the TOKEN register with an OUT token to the desired endpoint. The write to this register triggers the USB-FS transmit state machines to begin transmitting the TOKEN and the data.
6. Setup the Odd TX EP0 BDT to transfer up to 64 bytes.
7. Write the TOKEN register with an OUT token as in step 4. Two Tokens can be queued at a time to allow the packets to be double buffered to achieve maximum throughput.
8. Wait for the TOK_DNE interrupt. This indicates one of the BDTs has been released back to the microprocessor and that the transfer has completed. If the target device asserts NAKs, the USB-FS continues to retry the transfer indefinitely without processor intervention unless the RETRY_DIS retry disable bit is set in the EP0 control register. If the retry disable bit is set, the handshake (ACK, NAK, STALL, or ERROR (0xf)) is returned in the BDT PID field. If a stall interrupt occurs, the pending packet must be dequeued and the error condition in the target device cleared. If a RESET interrupt occurs (SE0 for more than 2.5us), the target has detached.
9. After the TOK_DNE interrupt occurs, the BDTs can be examined and the next data packet queued by returning to step 2.

45.7 On-The-Go Operation

The USB-OTG core provides sensors and controls to enable On-The-Go (OTG) operation. These sensors are used by the OTG API software to implement the Host Negotiation Protocol (HNP) and Session Request Protocol (SRP). API calls are provided to give access the OTG protocol control signals, and include the OTG capabilities in the device application. The following state machines show the OTG operations involved with HNP and SRP protocols from either end of the USB cable.

45.7.1 OTG Dual Role A Device Operation

A device is considered the A device because of the type of cable attached. If the USB Type A connector or the USB Type Mini A connector is plugged into the device, he is considered the A device.

A dual role A device operates as the following flow diagram and state description table illustrates.

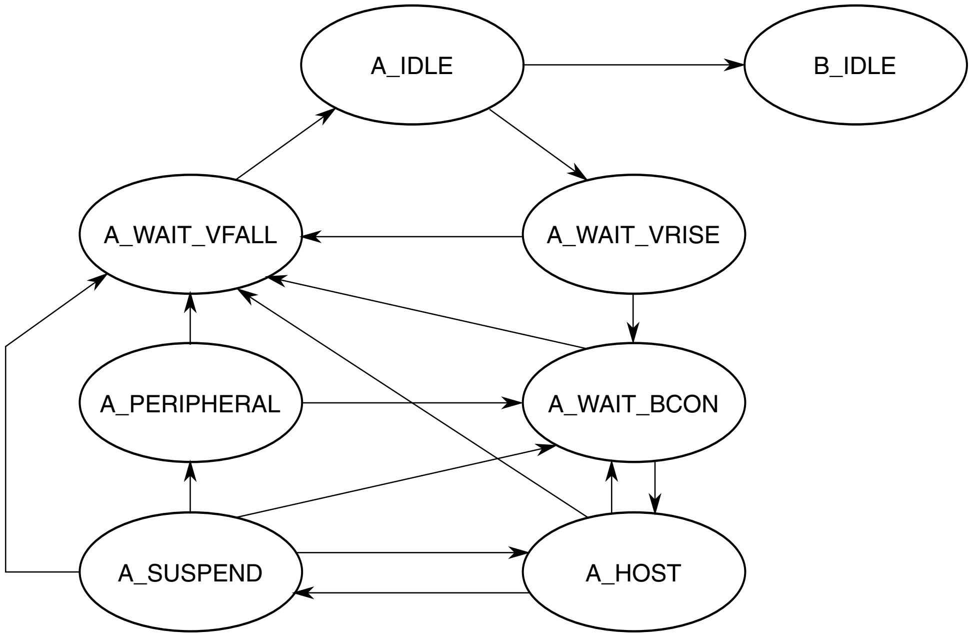


Figure 45-93. Dual Role A Device Flow Diagram

Table 45-96. State Descriptions for the Dual Role A Device Flow

State	Action	Response
A_IDLE	If ID Interrupt. The cable has been un-plugged or a Type B cable has been attached. The device now acts as a Type B device.	Go to B_IDLE
	If the A application wants to use the bus or if the B device is doing an SRP as indicated by an A_SESS_VLD Interrupt or Attach or Port Status Change Interrupt check data line for 5 –10 msec pulsing.	Go to A_WAIT_VRISE Turn on DRV_VBUS
A_WAIT_VRISE	If ID Interrupt or if A_VBUS_VLD is false after 100 msec The cable has been changed or the A device cannot support the current required from the B device.	Go to A_WAIT_VFALL Turn off DRV_VBUS
	If A_VBUS_VLD interrupt	Go to A_WAIT_BCON

Table continues on the next page...

Table 45-96. State Descriptions for the Dual Role A Device Flow (continued)

State	Action	Response
A_WAIT_BCON	After 200 msec without Attach or ID Interrupt. (This could wait forever if desired.)	Go to A_WAIT_FALL Turn off DRV_VBUS
	A_VBUS_VLD Interrupt and B device attaches	Go to A_HOST Turn on Host Mode
A_HOST	Enumerate Device determine OTG Support.	
	If A_VBUS_VLD/ Interrupt or A device is done and doesn't think he wants to do something soon or the B device disconnects	Go to A_WAIT_VFALL Turn off Host Mode Turn off DRV_VBUS
	If the A device is finished with session or if the A device wants to allow the B device to take bus.	Go to A_SUSPEND
	ID Interrupt or the B device disconnects	Go to A_WAIT_BCON
A_SUSPEND	If ID Interrupt, or if 150 msec B disconnect timeout (This timeout value could be longer) or if A_VBUS_VLD\ Interrupt	Go to A_WAIT_VFALL Turn off DRV_VBUS
	If HNP enabled, and B disconnects in 150 msec then B device is becoming the host.	Go to A_PERIPHERAL Turn off Host Mode
	If A wants to start another session	Go to A_HOST
A_PERIPHERAL	If ID Interrupt or if A_VBUS_VLD interrupt	Go to A_WAIT_VFALL Turn off DRV_VBUS.
	If 3 –200 msec of Bus Idle	Go to A_WAIT_BCON Turn on Host Mode
A_WAIT_VFALL	If ID Interrupt or (A_SESS_VLD/ & b_conn/)	Go to A_IDLE

45.7.2 OTG Dual Role B Device Operation

A device is considered a B device if it connected to the bus with a USB Type B cable or a USB Type Mini B cable.

A dual role B device operates as the following flow diagram and state description table illustrates.

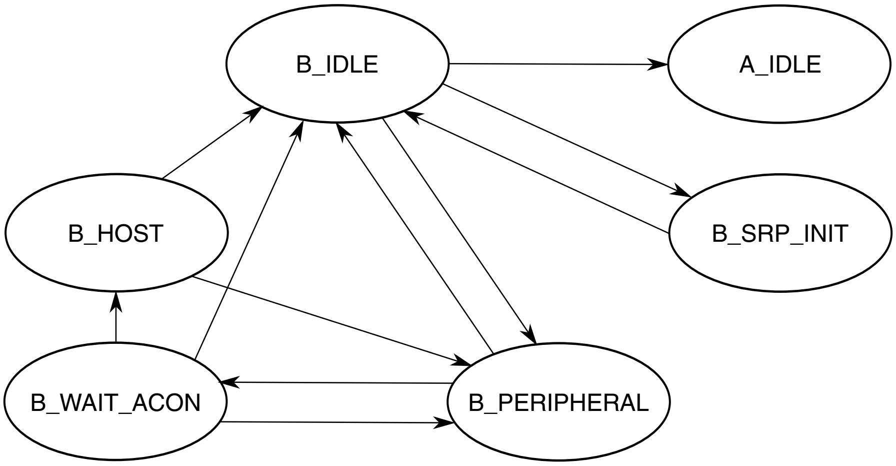


Figure 45-94. Dual Role B Device Flow Diagram

Table 45-97. State Descriptions for the Dual Role B Device Flow

State	Action	Response
B_IDLE	If ID\ Interrupt. A Type A cable has been plugged in and the device should now respond as a Type A device.	Go to A_IDLE
	If B_SESS_VLD Interrupt. The A device has turned on VBUS and begins a session.	Go to B_PERIPHERAL Turn on DP_HIGH
	If B application wants the bus and Bus is Idle for 2 ms and the B_SESS_END bit is set, the B device can perform an SRP.	Go to B_SRP_INIT Pulse CHRG_VBUS Pulse DP_HIGH 5-10 ms
B_SRP_INIT	If ID\ Interrupt or SRP Done (SRP must be done in less than 100 msecs.)	Go to B_IDLE
B_PERIPHERAL	If HNP enabled and the bus is suspended and B wants the bus, the B device can become the host.	Go to B_WAIT_ACON Turn off DP_HIGH
B_WAIT_ACON	If A connects, an attach interrupt is received	Go to B_HOST Turn on Host Mode
	If ID\ Interrupt or B_SESS_VLD/ Interrupt If the cable changes or if VBUS goes away, the host doesn't support us. Go to B_IDLE	Go to B_IDLE
	If 3.125 ms expires or if a Resume occurs	Go to B_PERIPHERAL
B_HOST	If ID\ Interrupt or B_SESS_VLD\ Interrupt If the cable changes or if VBUS goes away, the host doesn't support us.	Go to B_IDLE
	If B application is done or A disconnects	Go to B_PERIPHERAL

Chapter 46
USB Device Charger Detection Module (USBDCD)

46.1 Preface

46.1.1 References

The following publications are referenced in this document. For updates to these specifications, see http://www.usb.org.

- *USB Battery Charging Specification Revision 1.1, USB Implementers Forum*
- *Universal Serial Bus Specification Revision 2.0, USB Implementers Forum*

46.1.2 Acronyms and Abbreviations

The following table contains acronyms and abbreviations used in this document.

Table 46-1. Acronyms and Abbreviated Terms

Term	Meaning
FS	Full Speed (12 Mbps)
HS	High Speed (480 Mbps)
I_{DEV_DCHG}	Current drawn when the USB device is connected to a dedicated charging port
$I_{DEV\_HCHG\_LFS}$	Current drawn when the USB device is connected to an FS charging host port
I_{DM_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 Mbps)
N/A	Not applicable

Table continues on the next page...

Table 46-1. Acronyms and Abbreviated Terms (continued)

Term	Meaning
OTG	On-The-Go
R_{DM_DWN}	D- pulldown resistance for data pin contact detect
V_{DAT_REF}	Data detect reference voltage for the voltage comparator
V_{DP_SRC}	Voltage source for the D+ line
V_{LGC}	Threshold voltage for logic high

46.1.3 Glossary

The following table shows a glossary of terms used in this document.

Table 46-2. 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.

46.2 Introduction

NOTE

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

The USBDCD module works with the USB transceiver to detect if the USB device is attached to a charging port (either a dedicated charging port or a charging host). System software coordinates the detection activites of the module and controls an off-chip integrated circuit that performs the battery charging.

46.2.1 Block Diagram

The following figure is a high level block diagram of the module.

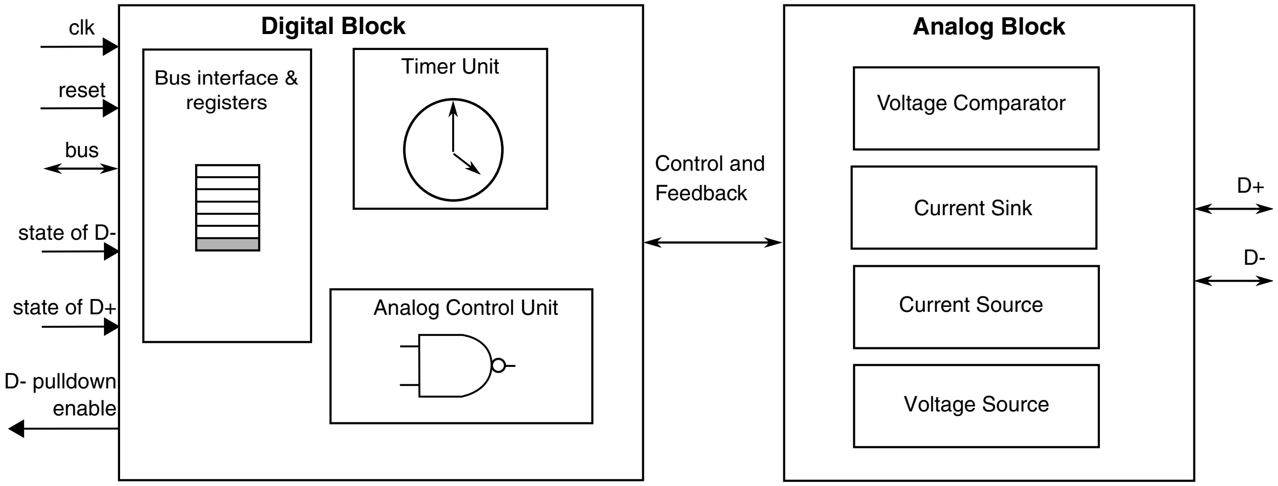


Figure 46-1. Block Diagram

The USBDCD module consists of 2 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 voltage source, current source, current sink, and voltage comparator circuitry.

46.2.2 Features

The USBDCD module offers the following features:

- Compliant with the latest industry standard specification: *USB Battery Charging Specification, Revision 1.1*
- Programmable timing parameters default to values required by the industry standards:
  - Having standard default values allows for minimal configuration: Set the clock frequency before enabling the module.
  - Programmability allows for flexibility to meet future udpates to the standards.

46.2.3 Modes of Operation

The USBDCD module operating modes are shown in the following table.

Table 46-3. 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. Optionally, the analog supply voltage avdd33 also may be reduced to as low as 1.7v without causing excess leakage.	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 46-4. Entering and Exiting Module Modes

Module Mode	Entering	Exiting	Mode after Exiting
Enabled	Set the CONTROL[START] bit.	Set the CONTROL[SR] bit.[1]	Disabled
Disabled	Take *either* of the following actions: • Set the CONTROL[SR] bit.[1] • Reset the module. (The module is disabled out of reset by default.)	Set the CONTROL[START] bit.	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.

46.3 Module Signal Description

This section describes the module signals.

46.3.1 USB Signal Descriptions

The following table shows a summary of module signals that interface with the device's pins.

Table 46-5. USB 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]	3.3v regulated analog supply	I
avss	Analog ground	I
dvss	Digital ground	I
dvdd	1.2 V digital supply	I

[1] Voltage must be 3.3v +/- 10% for full functionality of the module. That is, the charger detection function does not work when this voltage is below 3.0v, and the CONTROL[START] bit should not be set.

NOTE

The transceiver module also interfaces to usb_dm and usb_dp. Both modules and the USB host/hub use these signal as bi-directional, tri-state 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.

46.4 Memory Map/Register Definition

This section describes the memory map and registers for the USBDCD module.

USBDCD memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_5000	USBDCD_CONTROL	32	R/W	0001_0000h	46.4.1/ 1304
4003_5004	Clock Register (USBDCD_CLOCK)	32	R/W	0000_00C1h	46.4.2/ 1305

Table continues on the next page...

USBDCD memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_5008	Status Register (USBDCD_STATUS)	32	R	0000_0000h	46.4.3/ 1306
4003_5010	TIMER0 Register (USBDCD_TIMER0)	32	R/W	0010_0000h	46.4.4/ 1308
4003_5014	USBDCD_TIMER1	32	R/W	000A_0028h	46.4.5/ 1309
4003_5018	USBDCD_TIMER2	32	R/W	0028_0001h	46.4.6/ 1309

46.4.1 Control Register (USBDCD_CONTROL)

Contains the control and interrupt bit fields.

Address: USBDCD_CONTROL is 4003_5000h base + 0h offset = 4003_5000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						0	0	0							IE
W							SR	START								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							IF	0							0
W																IACK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBDCD_CONTROL field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value zero.
25 SR	Software Reset Determines whether a software reset is performed. 0 Do not perform a software reset. 1 Perform a software reset.
24 START	Start Change Detection Sequence

Table continues on the next page...

USBDCD_CONTROL field descriptions (continued)

Field	Description
	Determines whether the charger detection sequence is initiated. 0b0 Do not start the sequence. Writes of this value have no effect. 0b1 Initiate the charger detection sequence. If the sequence is already running, writes of this value have no effect.
23–17 Reserved	This read-only field is reserved and always has the value zero.
16 IE	Interrupt Enable Enables/disables interrupts to the system. 0b0 Disable interrupts to the system. 0b1 Enable interrupts to the system.
15–9 Reserved	This field is reserved.
8 IF	Interrupt Flag Determines whether an interrupt is pending 0b0 No interrupt is pending. 0b1 An interrupt is pending.
7–1 Reserved	This read-only field is reserved and always has the value zero.
0 IACK	Interrupt Acknowledge Determines whether the interrupt is cleared. 0b0 Do not clear the interrupt. 0b1 Clear the IF bit (interrupt flag).

46.4.2 Clock Register (USBDCD_CLOCK)

Address: USBDCD_CLOCK is 4003_5000h base + 4h offset = 4003_5004h

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				CLOCK_SPEED										0	CLOCK_UNIT
W	▨														▨	
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1

USBDCD_CLOCK field descriptions

Field	Description
31–12 Reserved	This read-only field is reserved and always has the value zero.
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	This read-only field is reserved and always has the value zero.
0 CLOCK_UNIT	Unit of measurement encoding for Clock Speed Specifies the unit of measure for the clock speed. 0b0 kHz Speed (between 1 kHz and 1023 kHz) 0b1 MHz Speed (between 1 MHz and 1023 MHz)

46.4.3 Status Register (USBDCD_STATUS)

The status register provides the current state of the module for system software monitoring.

Address: USBDCD_STATUS is 4003_5000h base + 8h offset = 4003_5008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									ACTIVE	TO	ERR	SEQ_STAT		SEQ_RES	
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	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBDCD_STATUS field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value zero.
22 ACTIVE	Active Status Indicator Indicates whether the sequence is running. 0b0 The sequence is not running. 0b1 The sequence is running.
21 TO	Timeout Flag Indicates whether the detection sequence has passed the timeout threshhold. 0b0 The detection sequence has not been running for over 1 s. 0b1 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. 0b0 No sequence errors. 0b1 Error in the detection sequence. See the SEQ_STAT field to determine the phase in which the error occurred.
19–18 SEQ_STAT	Charger Detection Sequence Status Indicates the status of the charger detection sequence. 0b00 The module is either not enabled, or the module is enabled but the data pins have not yet been detected. 0b01 Data pin contact detection is complete. 0b10 Charger detection is complete. 0b11 Charger type detection is complete.
17–16 SEQ_RES	Charger Detection Sequence Results Reports how charger detection is attached. 0b00 No results to report. 0b01 Attached to a standard host. Must comply with USB Spec 2.0 by drawing only 2.5mA (max) until connected. 0b10 Attached to a charging port. The exact meaning depends on bit 18: 0: Attached to either a charging host or a dedicated charger (The charger type detection has not completed.) 1: Attached to a charging host (The charger type detection has completed.) 0b11 Attached to a dedicated charger.
15–0 Reserved	This field is reserved. **NOTE:** Bits do not always read as 0.

46.4.4 TIMER0 Register (USBDCD_TIMER0)

TIMER0 has an TSEQ_INIT field that 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, however the USB Battery Charging Specification requires the entire sequence, including TSEQ_INIT, to be completed in 1s or less.

Address: USBDCD_TIMER0 is 4003_5000h base + 10h offset = 4003_5010h

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="6">0</td><td colspan="10" rowspan="2">TSEQ_INIT</td><td colspan="4">0</td><td colspan="12">TUNITCON</td></tr>
<tr><td>W</td><td colspan="6"></td><td colspan="4"></td><td colspan="12"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

USBDCD_TIMER0 field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value zero.
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	This read-only field is reserved and always has the value zero.
11–0 TUNITCON	Unit Connection Timer Elapse (in ms) Displays the current elapsed time since software set the CONTROL[START] bit plus the value of TSEQ_INIT. The timer is initially loaded with the value of TSEQ_INIT before starting to count. This timer enables compliance with the maximum time allowed to connect (TUNIT_CON) under the USB Battery Charging Specification, v1.1.If the timer reaches the TUNIT_CON 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.

46.4.5 USBDCD_TIMER1

Address: USBDCD_TIMER1 is 4003_5000h base + 14h offset = 4003_5014h

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						TDCD_DBNC										0						TVDPSRC_ON									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

USBDCD_TIMER1 field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value zero.
25–16 TDCD_DBNC	Time Period to Debounce D+ Signal Sets the amount of time (in ms) to debounce the D+ signal during the data pin contact detection phase (while IDP_SRC and RDM_DWN are enabled). Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 10 ms.
15–10 Reserved	This read-only field is reserved and always has the value zero.
9–0 TVDPSRC_ON	Time Period Comparator Enabled Sets the amount of time (in ms) that VDP_SRC, IDM_SINK, and the D-/VDAT_REF comparator are enabled and connected to the D+/D- lines during the charging port detection phase of the sequence. Valid values are 1-1023, but the USB Battery Charging Specification requires a minimum value of 40 ms.

46.4.6 USBDCD_TIMER2

TIMER2 contains timing parameters. Note that register values can be written that are not compliant with the USB Battery Charging Specification v1.1, so care should be taken when overwriting the default values.

Address: USBDCD_TIMER2 is 4003_5000h base + 18h offset = 4003_5018h

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						TVDPSRC_CON										0												CHECK_DM			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

USBDCD_TIMER2 field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

USBDCD_TIMER2 field descriptions (continued)

Field	Description
25–16 TVDPSRC_CON	Time Period Before Enabling D+ Pullup Sets the amount of time (in ms) that the module waits after charging port detection before system software should 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	This read-only field is reserved and always has the value zero.
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 (software enables the D+ pullup) until checking the state of the D- line to determine the type of charging port. Valid values are 1-15ms.

46.5 Functional Description

The sequence of detecting the presence of 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.

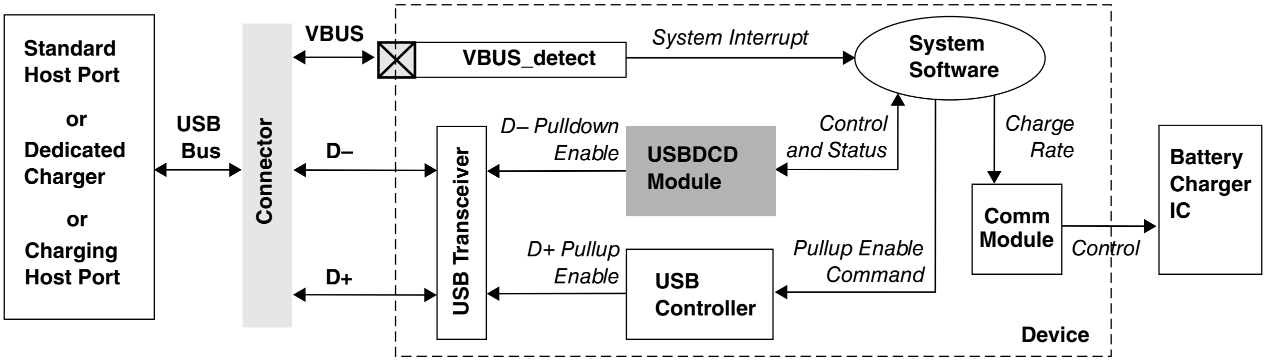


Figure 46-8. The USB Battery Charging Subsystem

Table 46-13. USB Battery Charger Subsystem Components

<table>
<tr><th>Component</th><th>Description</th></tr>
<tr><td>Battery Charger IC</td><td>The external battery charger IC regulates the charge rate to the rechargable battery. System software is responsible for communicating the appropriate charge rates.
<table>
<tr><th>Charger</th><th>Maximum Current Drawn[1]</th></tr>
<tr><td>Standard host port</td><td>up to 500 mA</td></tr>
<tr><td>Charging host port</td><td>up to 1500 mA</td></tr>
<tr><td>Dedicated charging port</td><td>up to 1800 mA</td></tr>
</table>
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.</td></tr>
<tr><td>Comm Module</td><td>A communications module on the device can be used to control the charge rate of the battery charger IC.</td></tr>
<tr><td>System software</td><td>Coordinates the detection activities of the subsystem.</td></tr>
<tr><td>USB Controller</td><td>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. Once this pullup is enabled, the device is considered to be connected to the USB bus. The host then attempts to enumerate it.
Note that the USB controller must be used only for USB <i>device</i> applications when using the USBDCD module. For USB <i>host</i> applications the USBDCD module must be disabled.</td></tr>
<tr><td>USB Transceiver</td><td>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. 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: The D+ pullup enable is physically output from the USB controller but is under software control. The USBDCD module controls the D- pulldown enable.</td></tr>
<tr><td>USBDCD Module</td><td>Detects if the device has been plugged into either a standard host port, a charging host port, or a dedicated charger.</td></tr>
<tr><td>VBUS_detect</td><td>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 upon 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 should be configured as an interrupt whenever the USB plug or unplug event is required to be detected.</td></tr>
</table>

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.

46.5.1 The Charger Detection Sequence

The following figure illustrates the charger detection sequence in a simplified timing diagram based on the USB Battery Charging Specification v1.1.

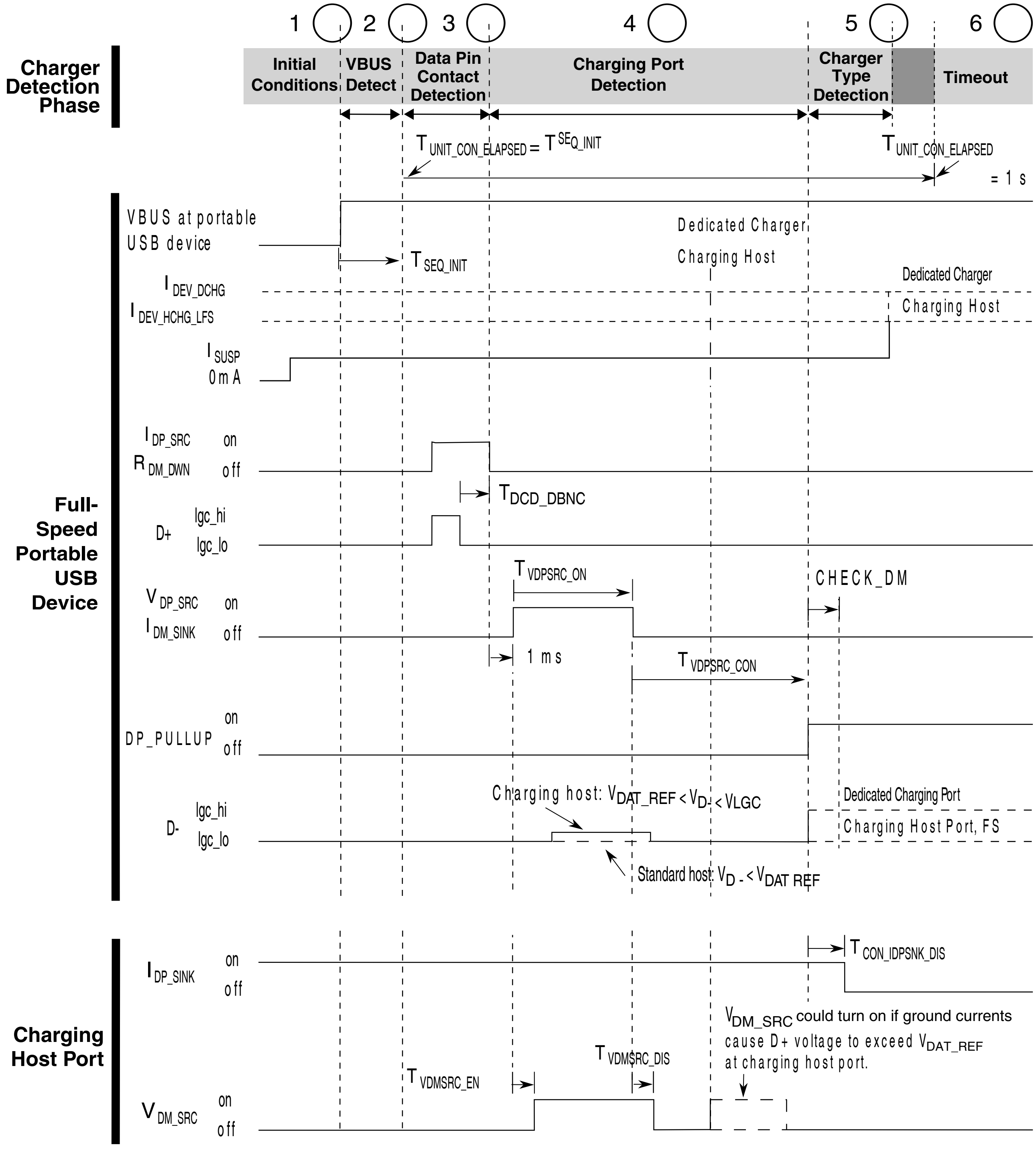


Figure 46-9. Full Speed Charger Detection Timing

The following table provides an overview description of the charger detection sequence shown in the preceding figure.

Table 46-14. Overview of the Charger Detection Sequence

Phase		Overview Description	Full Description
1	Initial Conditions	Initial system conditions that need to be met before initiating the detection sequence	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 a standard host or either type of charging port (charging host or dedicated charger).	Charging Port Detection
5	Charger Type Detection	If attached to a charging port, detect which type.	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

Timing parameter values used in this module are listed in the following table.

Table 46-15. Timing Parameters for the Charger Detection Sequence

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, v1.1*.

46.5.1.1 Initial System Conditions

Before starting the USBDCD module's charger detection sequence, the system must be:

- using a rechargable battery,
- for a FS USB *device* application (cannot be HS, LS, host, or OTG),
- powered-up and in run mode,

- recently plugged into a USB port, and
- drawing no more than 2.5 mA total system current from the USB bus.

There are many allowable precursors to this set of initial conditions. For example, the device could have been powered down and subsequently powered up upon being plugged into the USB bus. Alternatively, the device could have been in a low power state that was exited due to the plugin event. Or, the device could have been operating in normal run mode, powered by a separate supply or non-rechargable battery.

46.5.1.2 VBUS Contact Detection

Once the device is plugged into a USB port, the VBUS_detect system interrupt is triggered. System software should do the following to initialize the module and start the charger detection sequence:

1. Restore power if the module is powered-off.
2. Set the CONTROL[SR] bit to initiate a software reset.
3. Configure the USBDCD module: Program the CLOCK register and the timing parameters as needed.
4. Set the CONTROL[IE] bit to enable interrupts (by default), or clear the bit if using a software polling method.
5. Set the CONTROL[START] bit to start the charger detection sequence.

46.5.1.3 Data Pin Contact Detection

Because the detection sequence depends upon the state of the USB D+, the module must ensure that the data pins have made contact. 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.

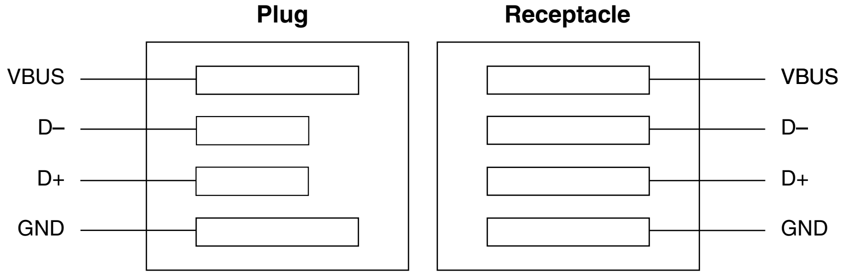


Figure 46-10. Relative Pin Positions in USB Plugs and Receptacles

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.

46.5.1.3.1 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. Once the data pins make contact, the D+ line goes low and debouncing begins.

Once the D+ line goes low, the module continuously samples the D+ line over the duration of the T_{DCD_DBNC} debounce time interval.T_{DCD_DBNC} defaults to 10 ms but 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:

- the data pin contact has been successfully debounced (see Success in Detecting Data Pin Contact (Phase Completion)), or
- a timeout occurs (see Charger Detection Sequence Timeout).

46.5.1.3.2 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 46-18 for field values.)
- directly proceeds to the next step in the sequence: detection of a charging port See Charging Port Detection.

46.5.1.4 Charging Port Detection

Once it is known that the data pins have made contact, the module waits for a fixed delay of 1 ms, and then attempts to detect if it has been plugged into a charging port. The module connects the following analog units to the USB D+ or D- lines during this phase (when the usbdcd_en and usbdcd_chg_det_en signals are asserted high):

- 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 46-16. Sampling D- in the Charging Port Detection Phase

If the voltage on D- is...	Then...	See...
Below V_{DAT_REF}	The port is a *standard host* that does not support the USB Battery Charging Specification v1.1.	Standard Host 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

46.5.1.4.1 Standard Host Port

As part of the charger detection handshake with a standard USB host, the module does the following (without waiting for the T_{VDPSRC_CON} interval to elapse):

- Updates the STATUS register to reflect that a standard host has been detected with SEQ_RES = 01. (See Table 46-18 for field values.)

- Sets the CONTROL[IF] bit.
- Generates an interrupt if enabled (the CONTROL[IE] bit is set).

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 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.
5. Communicate the appropriate charge rate to the external battery charger IC; see Table 46-13.

46.5.1.4.2 Charging Port

As part of the charger detection handshake with any type of USB host, the module waits until the T_{VDPSRC_CON} interval has elapsed before doing the following:

- Updates the STATUS register to reflect that a charging port has been detected with SEQ_RES = 10. (See Table 46-18 for field values.)
- Sets the CONTROL[IF] bit.
- Generates an interrupt if enabled (the CONTROL[IE] bit is set).

At this point, control has passed to system software via the interrupt. Software should:

1. Read the STATUS register.
2. Set the CONTROL[IACK] bit 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.

46.5.1.4.3 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 46-18 for field values.)
- Sets the CONTROL[IF] bit.
- Generates an interrupt if enabled (the CONTROL[IE] bit is set).

Note that in this case the module does not wait for the T_{VDPSRC_CON} interval 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 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.

46.5.1.5 Charger Type Detection

After software enables the D+ pullup resistor, the module is notified automatically (via internal signaling; the module waits until the ipp_pue_pullup_dp input goes high) to start the CHECK_DM timer counting down the time interval programmed into the TIMER2[CHECK_DM] field.

Once the CHECK_DM time has elapsed, the module samples the USB D- line to determine the type of charger. See the following table.

Table 46-17. Sampling D- in the Charger Type Detection Phase

If the voltage on D- is...	Then...	See...
High	The port is a *dedicated charging port*.[1]	Dedicated Charging Port
Low	The port is a *charging host port*.[2]	Charging Host Port

[1] In a dedicated charger, the D+ and D- lines are shorted together through a small resistor.
[2] In a charging host port, the D+ and D- lines are not shorted.

46.5.1.5.1 Dedicated Charging Port

For a dedicated charger, the module does the following:

- Updates the STATUS register to reflect that a dedicated charger has been detected with SEQ_RES = 11. (See Table 46-18 for field values.)
- Sets the CONTROL[IF] bit.
- Generates an interrupt if enabled (the CONTROL[IE] bit is set).

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 wake-up events to the system.
3. Set the CONTROL[IACK] bit to acknowledge the interrupt.
4. Set the CONTROL[SR] bit to issue a software reset to the module.
5. Disable the module.
6. Communicate the appropriate charge rate to the external battery charger IC; see Table 46-13.

46.5.1.5.2 Charging Host Port

For a charging host port, the module does the following:

- Updates the STATUS register to reflect that a charging host port has been detected with SEQ_RES = 10. (See Table 46-18 for field values.)
- Sets the CONTROL[IF] bit.
- Generates an interrupt if enabled (the CONTROL[IE] bit is set).

At this point, control has been passed to system software via the interrupt. 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.
5. Communicate the appropriate charge rate to the external battery charger IC; see Table 46-13.

46.5.1.6 Charger Detection Sequence Timeout

The maximum time to connect allowed under the *USB Battery Charging Specification, v1.1* is one second. If the Unit Connection Timer reaches the one second limit and the sequence is still running (indicated by the STATUS[ACTIVE] bit still being set), the module does the following:

- Updates the STATUS register to reflect that a timeout error has occured. (See Table 46-18 for field values.)
- Sets the CONTROL[IF] bit.
- Generates an interrupt if enabled (the CONTROL[IE] bit is set).
- 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 user unplugged the USB device from the 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 the $T_{UNITCON}$ register field (see the description of the TIMER0 Register) 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.

46.5.2 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 46-18. 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.	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 as 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.

46.5.2.1 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 the CONTROL[IE] bit is set. The CONTROL[IF] bit is always set under interrupt conditions, even if the IE bit is cleared. In this case, software can poll the IF flag to determine if an interrupt condition is pending.

Writes to the IF bit are ignored. To reset the IF bit, set the CONTROL[IACK] bit to acknowledge the interrupt.Writing to the IACK bit while the IF bit is cleared has no effect.

46.5.3 Resets

There are two ways to reset various register contents in this module: hardware resets and a software reset.

46.5.3.1 Hardware Resets

Hardware resets originate at the system or device level and propagate down to the individual module level. They include power-on 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.

46.5.3.2 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 the CONTROL[SR] bit initiates a software reset. The following table shows what register fields are reset to their default values by a software reset.

Table 46-19. 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] The CONTROL[SR, IACK] bits are self-clearing.

A software reset also returns all internal logic, timers, and counters to their reset states. State Machines return to IDLE. If the module is already active (STATUS[ACTIVE] = 1), a software reset stops the sequence.

Note

> Software should always initiate a software reset before starting the sequence (setting the CONTROL[START] bit) to ensure the module is in a known state.

46.6 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 other registers generally do not need to be modified because they default to values that comply with the USB Battery Charging Specification v1.1. 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 the CONTROL[START] bit result in undefined behavior.

46.7 Application Information

This section provides application information.

46.7.1 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.

46.7.2 Dead or Weak Battery

According to the USB Battery Charging Specification v1.1, a USB device with a dead, weak, or missing battery that is attached to a charging port can remain attached indefinitely drawing up to 1.5A until the battery is charged to the point that the USB device can connect.

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 D+ line high after the USBDCD module has determined that the device is attached to a charging port. The module is also compatible with systems that do 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 at 1.5A. Once the battery is charged to the good battery threshold, software can then connect to the USB host by pulling the D+ line high.

46.7.3 Handling Unplug Events

If the device is unplugged from the USB bus during the charger detection sequence, the contents of the STATUS register should be ignored and the USBDCD module should get a Sotware Reset, as described in Software Reset.

Chapter 47
USB Voltage Regulator

47.1 Introduction

NOTE

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

The USB Voltage Regulator module is a LDO linear voltage regulator to provide 3.3V power from an input power supply varying from 2.7 V to 5.5 V. It consists of one 3.3 V power channel. When the input power supply is below 3.6 V, the regulator goes to pass-through mode. The following figure shows the ideal relation between the regulator output and input power supply.

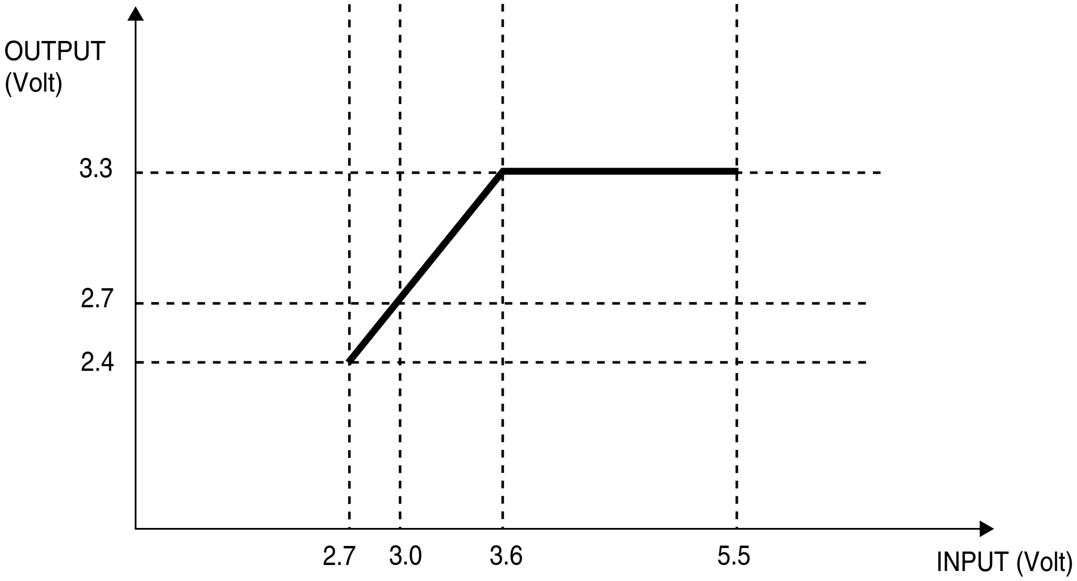


Figure 47-1. Ideal Relation Between the Regulator Output and Input Power Supply

47.1.1 Overview

A simplified block diagram for the USB Voltage Regulator module is shown below.

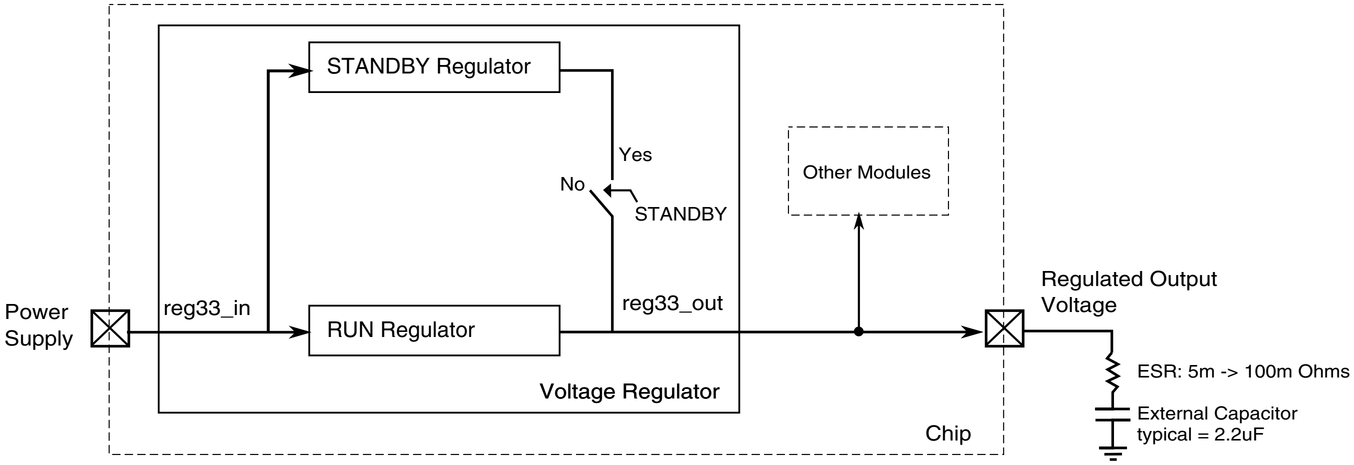


Figure 47-2. USB Voltage Regulator Block Diagram

This module uses 2 regulators in parallel. In run mode, the RUN regulator with the bandgap voltage reference is enabled and can provide up to 120 mA load current. In run mode, the STANDBY regulator and the low power reference are also enabled, but a switch disconnects its output from the external pin. In STANDBY mode, the RUN regulator is disabled and the STANDBY regulator output is connected to the external pin supplying up to 3 mA load current.

Internal power mode signals control whether the module is in RUN or STANDBY mode.

47.1.2 Features

- Low drop-out linear voltage regulator with one power channel (3.3V).
- Low drop-out voltage: 300 mV.
- Output current: 120 mA.
- Three different power modes: RUN, STANDBY and SHUTDOWN.
- Low quiescent current in RUN mode.
  - Typical value is around 120 uA (one thousand times smaller than the maximum load current).
- Very low quiescent current in STANDBY mode.
  - Typical value is around 1 uA.

- Automatic current limiting if the load current is greater than 290 mA.
- Automatic power-up once some voltage is applied to the regulator input.
- Pass-through mode for regulator input voltages less than 3.6 V
- Small output capacitor: 2.2 uF
- Stable with aluminum, tantalum or ceramic capacitors.

47.1.3 Modes of Operation

The regulator has these power modes:

- RUN—The regulating loop of the RUN regulator and the STANDBY regulator are active, but the switch connecting the STANDBY regulator output to the external pin is open.
- STANDBY—The regulating loop of the RUN regulator is disabled and the standby regulator is active. The switch connecting the STANDBY regulator output to the external pin is closed.
- SHUTDOWN—The module is disabled.

The regulator is enabled by default. This means that once the power supply is provided, the module power-up sequence to RUN mode starts.

47.2 USB Voltage Regulator Module Signal Descriptions

The following table shows the external signals for the regulator.

Table 47-1. USB Voltage Regulator Module Signal Descriptions

Signal	Description	I/O
reg33_in	Unregulated power supply	I
reg33_out	Regulator output voltage	O

Chapter 48
USB High Speed OTG Controller (USBHS)

48.1 Introduction

NOTE

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

This chapter describes the USB high speed OTG controller (USBHS), which implements many industry standards. However, it is beyond the scope of this document to document the intricacies of these standards. Instead, you should refer to the governing specifications. Readers of this chapter are assumed to be fluent in the operation and requirements of a USB network.

Visit the USB Implementers Forum web page at http://www.usb.org/developers/docs for:

- *Universal Serial Bus Specification, Revision 2.0*
- *On-The-Go Supplement to the USB 2.0 Specification, Revision 1.0a*

Visit the Intel USB specifications web page at http://www.intel.com/technology/usb/ehcispec for:

- *Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 1.0*

Visit the ULPI web page at http://www.ulpi.org for:

- *UTMI+ Specification, Revision 1.0*
- *UTMI Low Pin Interface (ULPI) Specification, Revision 1.1*

48.1.1 Overview

The USB high speed OTG controller (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) from Intel Corporation. The USBHS module can act as a host, a device, or an On-The-Go negotiable host/device on the USB bus.

The USBHS controller interfaces to the processor's core. The controller is programmable to support host or device operations under firmware control. The ULPI interface supports high-speed (HS) applications, as well as lower speeds.

48.1.2 Features

The USB On-The-Go module includes these features:

- Complies with USB specification rev 2.0
- USB host mode
    - Supports enhanced-host-controller interface (EHCI).
    - Supported by Linux and other commercially available operating systems.
- USB device mode
    - Supports full-speed/high-speed operation via an external ULPI transceiver.
    - Supports one upstream facing port.
    - Supports four programmable, bidirectional USB endpoints, including endpoint 0. See endpoint configurations:

Table 48-1. Endpoint Configurations

Endpoint	Type	FIFO Size	Data Transfer	Comments
0	Bidirectional	Variable	Control	Mandatory
1-3	IN or OUT	Variable	Ctrl, Int, Bulk, or Iso	Optional

- Suspend mode/low power
    - As host, firmware can suspend individual devices or the entire USB and disable chip clocks for low-power operation
    - Device supports low-power suspend

  - Remote wake-up supported for host and device
  - Integrated with the processor's low power modes
- Support for off-chip HS/FS/LS transceiver
  - External ULPI transceiver supports high speed (480 Mbps), full speed, and low speed operation in host mode, and high-speed and full-speed operation in device mode
  - Interface uses 8-bit single-data-rate ULPI data bus
  - ULPI PHY supplies a 60 MHz USB reference clock input to the processor

48.1.3 Modes of Operation

The USBHS module has two basic operating modes: host and device. Selection of operating mode is accomplished via the USBMODE[CM] bit field.

Speed selection is auto-detected at connect time using enumeration procedures in the USB network. The USBHS module provides these operation modes:

- USB disabled. In this mode, the USBHS's datapath does not accept transactions received on the USB interface.
- USB enabled. In this mode, the USB host's datapath is enabled to accept transactions received on the USB interface.
- USB enabled, low-power modes.

48.2 External Signal Description

This table describes the external signal functionality of the USBHS module.

Table 48-2. USBHS Signal Descriptions

Signal	I/O	Description
ULPI_CLK	I	60 MHz clock input from the ULPI transceiver

Table continues on the next page...

Table 48-2. USBHS Signal Descriptions (continued)

Signal	I/O	Description	
ULPI_DIR	I	Direction. ULPI_DIR controls data bus direction. When PHY has data to transfer to USB port, it drives ULPI_DIR high to take ownership of the bus. When the PHY has no data to transfer, it drives ULPI_DIR low and monitors the bus for link activity. The PHY pulls ULPI_DIR high when the interface cannot accept data from the link. For example, when PHY's PLL is not stable.	
		State Meaning	Asserted—PHY has data to transfer to the link. Negated—PHY has no data to transfer.
		Timing	Synchronous to ULPI_CLK.
ULPI_NXT	I	Next data. PHY asserts ULPI_NXT to throttle data. When USB port sends data to the PHY, ULPI_NXT indicates when PHY accepts the current byte. The USB port places the next byte on the data bus in the following clock cycle. When the PHY sends data to USB port, ULPI_NXT indicates when a new byte is available for USB port to consume.	
		State Meaning	Asserted—PHY is ready to transfer byte. Negated—PHY is not ready.
		Timing	Synchronous to ULPI_CLK.
ULPI_STP	O	Stop. ULPI_STP indicates the end of a transfer on the bus.	
		State Meaning	Asserted—USB asserts this signal for one clock cycle to stop the data stream currently on the bus. If the USB port sends data to the PHY, ULPI_STP indicates the last data byte was previously on the bus. If the PHY is sending data to the USB port, ULPI_STP forces the PHY to end its transfer, deassert ULPI_DIR, and relinquish control of the data bus to the USB port. Negated—Indicates normal operation.
		Timing	Synchronous to ULPI_CLK
ULPI_DATA[7:0]	I/O	Data bit n. ULPI_DATAT*n* is bit *n* of the 8-bit, bi-directional data bus used to carry USB, register, and interrupt data between the USB port controller and the PHY.	
		State Meaning	Asserted—Data bit *n* is 1. Negated—Data bit *n* is 0.
		Timing	Synchronous to ULPI_CLK

48.3 Memory Map/Register Definition

This section provides the memory map and detailed descriptions of the USBHS registers.

Table 48-3. USBHS Register Overview

Address	Register	EHCI[1]	H/D[2]
Module Identification Registers: Declare the slave interface presence and include a table of the hardware configuration parameters.			
0x000	Identification Register (ID)	N	H/D

Table continues on the next page...

Table 48-3. USBHS Register Overview (continued)

Address	Register	EHCI[1]	H/D[2]
0x004	General Hardware Parameters (HWGENERAL)	N	H/D
0x008	Host Hardware Parameters (HWHOST)	N	H/D
0x00C	Device Hardware Parameters (HWDEVICE)	N	D
0x010	TX Buffer Hardware Parameters (HWTXBUF)	N	H/D
0x014	RX Buffer Hardware Parameters (HWRXBUF)	N	H/D
Device/Host Timer Registers: Used by host/device controller drivers to measure time-related activities.			
0x080	General Purpose Timer 0 Load (GPTIMER0LD)	N	H/D
0x084	General Purpose Timer 0 Control (GPTIMER0CTL)	N	H/D
0x088	General Purpose Timer 1 Load (GPTIMER1LD)	N	H/D
0x08C	General Purpose Timer 1 Control (GPTIMER1CTL)	N	H/D
Capability Registers: Specifies software limits, restrictions, and capabilities of the host/device controller implementation.			
0x100	Host Interface Version Number (HCIVERSION)	Y	H
0x103	Capability Register Length (CAPLENGTH)	Y	H/D
0x104	Host Structural Parameters (HCSPARAMS)	Y	H
0x108	Host Capability Parameters (HCCPARAMS)	Y	H
0x122	Device Interface Version Number (DCIVERSION)	N	D
0x124	Device Capability Parameters (DCCPARAMS)	N	D
Operational Registers: Comprised of dynamic control or status registers.			
0x140	USB Command (USBCMD)	Y	H/D
0x144	USB Status (USBSTS)	Y	H/D
0x148	USB Interrupt Enable (USBINTR)	Y	H/D
0x14C	USB Frame Index (FRINDEX)	Y	H/D
0x154	Periodic Frame List Base Address (PERIODICLISTBASE)	Y	H
0x154	Device Address (DEVICEADDR)	N	D
0x158	Current Asynchronous List Address (ASYNCLISTADDR)	Y	H
0x158	Address at Endpoint List (EPLISTADDR)	N	D
0x15C	Host TT Asynchronous Buffer Control (TTCTRL)	N	H
0x160	Master Interface Data Burst Size (BURSTSIZE)	N	H/D
0x164	Host Transmit FIFO Tuning Control (TXFILLTUNING)	N	H
0x170	ULPI Register Access (ULPI_VIEWPORT)	N	H/D
0x180	Configure Flag Register (CONFIGFLAG)	Y	H/D
0x184	Port Status/Control (PORTSC1)	Y	H/D
0x1A4	On-The-Go Status and Control (OTGSC)	N	H/D
0x1A8	USB Mode Register (MODE)	N	H/D

Table continues on the next page...

Table 48-3. USBHS Register Overview (continued)

Address	Register	EHCI[1]	H/D[2]
0x1AC	Endpoint Setup Status Register (EPSETUPSR)	N	D
0x1B0	Endpoint Initialization (EPPRIME)	N	D
0x1B4	Endpoint De-initialize (EPFLUSH)	N	D
0x1B8	Endpoint Status Register (EPSR)	N	D
0x1BC	Endpoint Complete (EPCOMPLETE)	N	D
0x1C0	Endpoint Control Register 0 (EPCR0)	N	D
0x1C4	Endpoint Control Register 1 (EPCR1)	N	D
0x1C8	Endpoint Control Register 2 (EPCR2)	N	D
0x1CC	Endpoint Control Register 3 (EPCR3)	N	D

1. Indicates if the register is present in the EHCI specification.
2. Indicates if the register is available in host and/or device modes.

USBHS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_4000	Identification Register (USBHS_ID)	32	R	E461_FA05h	48.3.1/ 1337
4003_4004	General Hardware Parameters Register (USBHS_HWGENERAL)	32	R	0000_0085h	48.3.2/ 1338
4003_4008	Host Hardware Parameters Register (USBHS_HWHOST)	32	R	1002_0001h	48.3.3/ 1339
4003_400C	Device Hardware Parameters Register (USBHS_HWDEVICE)	32	R	0000_0009h	48.3.4/ 1339
4003_4010	Transmit Buffer Hardware Parameters Register (USBHS_HWTXBUF)	32	R	8007_0908h	48.3.5/ 1340
4003_4014	Receive Buffer Hardware Parameters Register (USBHS_HWRXBUF)	32	R (reads zero)	0000_0808h	48.3.6/ 1341
4003_4080	General Purpose Timer n Load Register (USBHS_GPTIMER0LD)	32	R/W	0000_0000h	48.3.7/ 1342
4003_4084	General Purpose Timer n Control Register (USBHS_GPTIMER0CTL)	32	R/W	0000_0000h	48.3.8/ 1343
4003_4088	General Purpose Timer n Load Register (USBHS_GPTIMER1LD)	32	R/W	0000_0000h	48.3.7/ 1342
4003_408C	General Purpose Timer n Control Register (USBHS_GPTIMER1CTL)	32	R/W	0000_0000h	48.3.8/ 1343

Table continues on the next page...

USBHS memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_4090	System Bus Interface Configuration Register (USBHS_USB_SBUSCFG)	32	R/W	0000_0000h	48.3.9/ 1344
4003_4100	Host Controller Interface Version and Capability Registers Length Register (USBHS_HCIVERSION)	32	R	0100_0040h	48.3.10/ 1345
4003_4104	Host Controller Structural Parameters Register (USBHS_HCSPARAMS)	32	R	0001_0011h	48.3.11/ 1346
4003_4108	Host Controller Capability Parameters Register (USBHS_HCCPARAMS)	32	R	0000_0006h	48.3.12/ 1348
4003_4122	Device Controller Interface Version (USBHS_DCIVERSION)	16	R	0001h	48.3.13/ 1349
4003_4124	Device Controller Capability Parameters (USBHS_DCCPARAMS)	32	R	0000_0184h	48.3.14/ 1350
4003_4140	USB Command Register (USBHS_USBCMD)	32	R/W	0008_0000h	48.3.15/ 1351
4003_4144	USB Status Register (USBHS_USBSTS)	32	R/W	0000_0000h	48.3.16/ 1354
4003_4148	USB Interrupt Enable Register (USBHS_USBINTR)	32	R/W	0000_0000h	48.3.17/ 1358
4003_414C	Frame Index Register (USBHS_FRINDEX)	32	R/W	0000_0000h	48.3.18/ 1360
4003_4154	Periodic Frame List Base Address Register (USBHS_PERIODICLISTBASE)	32	R/W	0000_0000h	48.3.19/ 1362
4003_4154	Device Address Register (USBHS_DEVICEADDR)	32	R/W	0000_0000h	48.3.20/ 1362
4003_4158	Current Asynchronous List Address Register (USBHS_ASYNCLISTADDR)	32	R/W	0000_0000h	48.3.21/ 1363
4003_4158	Endpoint List Address Register (USBHS_EPLISTADDR)	32	R/W	0000_0000h	48.3.22/ 1364
4003_415C	Host TT Asynchronous Buffer Control (USBHS_TTCTRL)	32	R/W	0000_0000h	48.3.23/ 1365
4003_4160	Master Interface Data Burst Size Register (USBHS_BURSTSIZE)	32	R/W	0000_8080h	48.3.24/ 1366
4003_4164	Transmit FIFO Tuning Control Register (USBHS_TXFILLTUNING)	32	R/W	0000_0000h	48.3.25/ 1366
4003_4170	ULPI Register Access (USBHS_ULPI_VIEWPORT)	32	R/W	0000_0000h	48.3.26/ 1368
4003_4178	Endpoint NAK Register (USBHS_ENDPTNAK)	32	R/W	0000_0000h	48.3.27/ 1370
4003_417C	Endpoint NAK Enable Register (USBHS_ENDPTNAKEN)	32	R/W	0000_0000h	48.3.28/ 1371

Table continues on the next page...

USBHS memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4003_4180	Configure Flag Register (USBHS_CONFIGFLAG)	32	R	0000_0001h	48.3.29/ 1372
4003_4184	Port Status and Control Registers (USBHS_PORTSC1)	32	R/W	8000_0000h	48.3.30/ 1372
4003_41A4	On-the-Go Status and Control Register (USBHS_OTGSC)	32	R/W	0000_1020h	48.3.31/ 1378
4003_41A8	USB Mode Register (USBHS_USBMODE)	32	R/W	0000_5000h	48.3.32/ 1382
4003_41AC	Endpoint Setup Status Register (USBHS_EPSETUPSR)	32	R/W	0000_0000h	48.3.33/ 1384
4003_41B0	Endpoint Initialization Register (USBHS_EPPRIME)	32	R/W	0000_0000h	48.3.34/ 1384
4003_41B4	Endpoint Flush Register (USBHS_EPFLUSH)	32	R/W	0000_0000h	48.3.35/ 1385
4003_41B8	Endpoint Status Register (USBHS_EPSR)	32	R	0000_0000h	48.3.36/ 1386
4003_41BC	Endpoint Complete Register (USBHS_EPCOMPLETE)	32	R/W	0000_0000h	48.3.37/ 1387
4003_41C0	Endpoint Control Register 0 (USBHS_EPCR0)	32	R/W	0080_0080h	48.3.38/ 1388
4003_41C4	Endpoint Control Register n (USBHS_EPCR1)	32	R/W	0000_0000h	48.3.39/ 1390
4003_41C8	Endpoint Control Register n (USBHS_EPCR2)	32	R/W	0000_0000h	48.3.39/ 1390
4003_41CC	Endpoint Control Register n (USBHS_EPCR3)	32	R/W	0000_0000h	48.3.39/ 1390
4003_4200	USB General Control Register (USBHS_USBGENCTRL)	32	R/W	0000_0000h	48.3.40/ 1392

48.3.1 Identification Register (USBHS_ID)

The ID register provides a simple way to determine if the module is provided in the system. The ID register identifies the module and its revision.

Address: USBHS_ID is 4003_4000h base + 0h offset = 4003_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VERSIONID			VERSION				REVISION				TAG				
W																
Reset	1	1	1	0	0	1	0	0	0	1	1	0	0	0	0	1

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

USBHS_ID field descriptions

Field	Description
31–29 VERSIONID	Version ID Internal version counter.
28–25 VERSION	Version Version of the module.
24–21 REVISION	Revision Revision number of the module.
20–16 TAG	Tag Tag of the module.
15–14 Reserved	Reserved This read-only field is reserved and always has the value one. Always reads as 0b11.
13–8 NID	Ones complement version of ID.
7–6 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared
5–0 ID	Configuration number This number is set to 0x05.

48.3.2 General Hardware Parameters Register (USBHS_HWGENERAL)

The HWGENERAL register contains parameters defining the particular implementation of the module.

Address: USBHS_HWGENERAL is 4003_4000h base + 4h offset = 4003_4004h

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																					SM		PHYM			0			1	0	1
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	1	0	0	0	0	1	0	1

USBHS_HWGENERAL field descriptions

Field	Description
31–11 Reserved	Reserved This read-only field is reserved and always has the value zero. Always reads as zeroes.
10–9 SM	Serial mode Indicates presence of serial interface. Always 00. 00 Serial engine not present
8–6 PHYM	PHY Mode Indicates USB transceiver interface used. Always reads 010. 010 ULPI only
5–3 Reserved	Reserved This read-only field is reserved and always has the value zero.
2 Reserved	Reserved This read-only field is reserved and always has the value one.
1 Reserved	Reserved This read-only field is reserved and always has the value zero.
0 Reserved	Reserved This read-only field is reserved and always has the value one.

48.3.3 Host Hardware Parameters Register (USBHS_HWHOST)

The HWHOST register provides host hardware parameters for this implementation of the module.

Address: USBHS_HWHOST is 4003_4000h base + 8h offset = 4003_4008h

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	TTPER								TTASY								0												NPORT			HC
W																																
Reset	0	0	0	1	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	1

USBHS_HWHOST field descriptions

Field	Description
31–24 TTPER	Transaction translator periodic contexts. The number of supported transaction translator periodic contexts. Always reads as 0x10 (16 contexts supported).
23–16 TTASY	Transaction translator contexts. The number of transaction translator contexts. Always reads as 0x02.
15–4 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared.
3–1 NPORT	Number of Ports Always 0, indicating the number of ports available (NPORT + 1) for this host implementation.
0 HC	Host Capable Always reads as 0b1, indicating the module is host capable.

48.3.4 Device Hardware Parameters Register (USBHS_HWDEVICE)

Provides device hardware parameters for this implementation of the USBHS module.

Address: USBHS_HWDEVICE is 4003_4000h base + Ch offset = 4003_400Ch

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																										DEVEP					DC
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	1	0	0	1

USBHS_HWDEVICE field descriptions

Field	Description
31–6 Reserved	Reserved This read-only field is reserved and always has the value zero. Always reads as zero
5–1 DEVEP	Device endpoints. The number of supported endpoints. Always reads as 0b00100.
0 DC	Device Capable Always reads as 0b1, indicating the USBHS module is device capable.

48.3.5 Transmit Buffer Hardware Parameters Register (USBHS_HWTXBUF)

The HWTXBUF register provides the transmit buffer parameters for this implementation of the module.

Address: USBHS_HWTXBUF is 4003_4000h base + 10h offset = 4003_4010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TXLC	0							TXCHANADD							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

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

USBHS_HWTXBUF field descriptions

Field	Description
31 TXLC	Transmit local Context Registers Indicates how the device transmit context registers implement. Always set. 0 Store device transmit contexts in the TX FIFO 1 Store device transmit contexts in a register file
30–24 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared

Table continues on the next page...

USBHS_HWTXBUF field descriptions (continued)

Field	Description
23–16 TXCHANADD	Transmit Channel Address The number of address bits required to address one channel's worth of TX data. Always 0x7.
15–8 TXADD	Transmit Address. The number of address bits for the entire TX buffer. Always 0x9.
7–0 TXBURST	Transmit Burst. Indicates the number of data beats in a burst for transmit DMA data transfers. Always 0x8.

48.3.6 Receive Buffer Hardware Parameters Register (USBHS_HWRXBUF)

The HWRXBUF register provide the receive buffer parameters for this implementation of the module.

Address: USBHS_HWRXBUF is 4003_4000h base + 14h offset = 4003_4014h

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																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

USBHS_HWRXBUF field descriptions

Field	Description
31–16 Reserved	Reserved This read-only field is reserved and always has the value zero.
15–8 RXADD	Receive Address. The number of address bits for the entire RX buffer. Always reads as 0x08.
7–0 RXBURST	Receive Burst. Indicates the number of data beats in a burst for receive DMA data transfers. Always reads as 0x08.

48.3.7 General Purpose Timer n Load Register (USBHS_GPTIMERLD)

The GPTIMERnLD register contains the timer duration or load value.

Addresses: USBHS_GPTIMER0LD is 4003_4000h base + 80h offset = 4003_4080h
USBHS_GPTIMER1LD is 4003_4000h base + 88h offset = 4003_4088h

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								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

USBHS_GPTIMER*n*LD field descriptions

Field	Description
31–24 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
23–0 GPTLD	Specifies the value to be loaded into the countdown timer on a reset. The value in this register represents the time in microseconds minus 1 for the timer duration. For example, for a one millisecond timer, load 1000 – 1 = 999 (0x00_03E7). **NOTE:** Maximum value is 0xFF_FFFF or 16.777215 seconds.

48.3.8 General Purpose Timer n Control Register (USBHS_GPTIMERCTL)

The GPTIMERnCTL register controls the various functions of the general purpose timers.

Addresses: USBHS_GPTIMER0CTL is 4003_4000h base + 84h offset = 4003_4084h
USBHS_GPTIMER1CTL is 4003_4000h base + 8Ch offset = 4003_408Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RUN		0					MODE	GPTCNT[8:16]							
W		RST														
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	GPTCNT[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBHS_GPTIMER*n*CTL field descriptions

Field	Description
31 RUN	Timer Run Enables the general purpose timer. Setting or clearing this bit does not have an effect on the GPTCNT field. 0 Timer stop 1 Timer run
30 RST	Timer Reset Setting this bit reloads GPTCNT with the value in GPTIMERnLD[GPTLD]. 0 No action 1 Load counter value
29–25 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
24 MODE	Timer Mode Selects between a single timer countdown and a looped countdown. In one-shot mode, the timer counts down to zero, generates an interrupt, and stops until the counter is reset by software. In repeat mode, the

Table continues on the next page...

USBHS_GPTIMER*n*CTL field descriptions (continued)

Field	Description
	timer counts down to zero, generates an interrupt, and automatically reloads the counter and begins another countdown. 0 One shot 1 Repeat
23–0 GPTCNT	Timer Count Indicates the current value of the running timer.

48.3.9 System Bus Interface Configuration Register (USBHS_USB_SBUSCFG)

The System Bus Interface Configuration (USB_SBUSCFG) register contains the control for the system bus interface.

Address: USBHS_USB_SBUSCFG is 4003_4000h base + 90h offset = 4003_4090h

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													BURSTMODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBHS_USB_SBUSCFG field descriptions

Field	Description
31–3 Reserved	Reserved This read-only field is reserved and always has the value zero.
2–0 BURSTMODE	Burst mode Selects the options for the burst signal of the Master Interface. In all cases where the unspecified length burst is allowed, single accesses may also occur; this is mostly true when the transaction is not 32-bit aligned. Two consecutive single accesses should not happen. When an INCRx burst size is selected and the transfer is not multiple of the INCRx burst, the burst is decomposed in the different ways. With BURSTMODE[2] = 1, the smaller bursts is unspecified length. with BURSTMODE[2] = 0, the smaller bursts are smaller INCRx or singles. For example, if it were required at a given time to transfer 22 words of information, for the following values of BURSTMODE the master sequences are:

Table continues on the next page...

USBHS_USB_SBUSCFG field descriptions (continued)

Field	Description
	101 INCR4 + INCR4 + INCR4 + INCR4 + INCR4 + INCR unspec. length. 110 INCR8 + INCR8 + INCR4 + INCR unspec. length. 111 INCR16 + INCR4 + INCR unspec. length. 001 INCR4 + INCR4 + INCR4 + INCR4 + INCR4 + SINGLE + SINGLE. 010 INCR8 + INCR8 + INCR4 + SINGLE + SINGLE. 011 INCR16 + INCR4 + SINGLE + SINGLE. When this field is different from zero, the values in the TXBURST/RXBURST bitfields in the USB_BURSTSIZE register are ignored by the controller. Internally the BURSTMODE is set to the value of the INCRx burst. Since this has a direct relation with the burst sizes you must be careful with AHB burst selected. Although the TXBURST/RXBURST are bypassed, this register can be written/read with no effect while the BURSTMODE field is non-zero. **NOTE:** Setting the BURSTMODE value to 000 might cause bus allocation during BULK or ISO transfers. **NOTE:** Changing this BURSTMODE field while a transaction is in progress yields undefined results. One possible way to prevent undefined results is to clear the Run/Stop (RS) bit in the USB_USBCMD register, after the HCHALTED is detected in USB_USBSTS. 000 INCR burst of unspecified length 001 INCR4, non-multiple transfers of INCR4 is decomposed into singles. 010 INCR8, non-multiple transfers of INCR8, is decomposed into INCR4 or singles. 011 INCR16, non-multiple transfers of INCR16, is decomposed into INCR8, INCR4 or singles. 100 Reserved, do not use. 101 INCR4, non-multiple transfers of INCR4 is decomposed into smaller unspecified length bursts. 110 INCR8, non-multiple transfers of INCR8 is decomposed into smaller unspecified length bursts. 111 INCR16, non-multiple transfers of INCR16 is decomposed into smaller unspecified length bursts.

48.3.10 Host Controller Interface Version and Capability Registers Length Register (USBHS_HCIVERSION)

Contains the CAPLENGTH field used as an offset to add to the register base address to find the beginning of the operational register space, the location of the USBCMD register. Also contains a BCD encoding of the EHCI revision number supported by this OTG controller. The most-significant byte of the HCIVERSION field represents a major revision; the least-significant byte is the minor revision.

Address: USBHS_HCIVERSION is 4003_4000h base + 100h offset = 4003_4100h

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	HCIVERSION																0								CAPLENGTH							
W																																
Reset	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	1	0	0	0	0	0	0

USBHS_HCIVERSION field descriptions

Field	Description
31–16 HCIVERSION	EHCI revision number Value is reads as 0x0100, indicating version 1.0.
15–8 Reserved	Reserved This read-only field is reserved and always has the value zero.
7–0 CAPLENGTH	Capability registers length Always reads as 0x40.

48.3.11 Host Controller Structural Parameters Register (USBHS_HCSPARAMS)

This register contains structural parameters such as the number of downstream ports.

Address: USBHS_HCSPARAMS is 4003_4000h base + 104h offset = 4003_4104h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				N_TT				N_PTT				0			PI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	N_CC				N_PCC				0			PPC	N_PORTS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

USBHS_HCSPARAMS field descriptions

Field	Description
31–28 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared.
27–24 N_TT	Number of Transaction Translators. Non-EHCI field. Indicates number of embedded transaction translators associated with host controller. This field always reads as 0x0. See section “Embedded Transaction Translator Function,” for more information on embedded transaction translators.
23–20 N_PTT	Ports per Transaction Translator Non-EHCI field.

Table continues on the next page...

USBHS_HCSPARAMS field descriptions (continued)

Field	Description
	Indicates number of ports assigned to each transaction translator within host controller.
19–17 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared.
16 PI	Port Indicators Indicates whether the ports support port indicator control. Always set. 0 No port indicator fields 1 The port status and control registers include a R/W field for controlling the state of the port indicator
15–12 N_CC	Number of Companion Controllers Indicates number of companion controllers associated with USBHS controller. Always cleared.
11–8 N_PCC	Number Ports per CC Indicates number of ports supported per internal companion controller. This field reads as 0x0 because no companion controllers are present.
7–5 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared.
4 PPC	Power Port Control Indicates whether host controller supports port power control. Always reads as 0b1. 1 Ports have power port switches
3–0 N_PORTS	Number of Ports Indicates number of physical downstream ports implemented for host applications. Field value determines how many addressable port registers in the operational register. Always reads as 0x1.

48.3.12 Host Controller Capability Parameters Register (USBHS_HCCPARAMS)

Identifies multiple mode control (time-base bit functionality) addressing capability.

Address: USBHS_HCCPARAMS is 4003_4000h base + 108h offset = 4003_4108h

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	EECP								IST				0	ASP	PFL	ADC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

USBHS_HCCPARAMS field descriptions

Field	Description
31–16 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared.
15–8 EECP	EHCI Extended Capabilities Pointer This optional field indicates the existence of a capabilities list. This field always reads as 0x000. 0x00 No extended capabilities are implemented
7–4 IST	Isochronous Scheduling Threshold Indicates where software can reliably update the isochronous schedule, relative to the current position of the executing host controller. This field always reads as 0x0. 0 The value of the least significant 3 bits indicates the number of microframes a host controller can hold a set of isochronous data structures (one or more) before flushing the state
3 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared.
2 ASP	Asynchronous Schedule Park capability Indicates if 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. This bit always reads as 0b1. 0 Park not supported. 1 Park supported.

Table continues on the next page...

USBHS_HCCPARAMS field descriptions (continued)

Field	Description
1 PFL	Programmable Frame List flag Indicates that system software can specify and use a frame list length less that 1024 elements. This bit always reads as 0b1. 1 Frame list size is configured 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.
0 ADC	64-bit addressing capability. This bit always reads as 0b0, indicating 64-bit addressing is not supported. 0 Data structures use 32-bit address memory pointers

48.3.13 Device Controller Interface Version (USBHS_DCIVERSION)

Not defined in the EHCI specification. DCIVERSION is a two-byte register containing a BCD encoding of the device controller interface. The most-significant byte of the register represents a major revision and the least-significant byte is the minor revision.

Address: USBHS_DCIVERSION is 4003_4000h base + 122h offset = 4003_4122h

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

USBHS_DCIVERSION field descriptions

Field	Description
15–0 DCIVERSION	Device interface revision number.

48.3.14 Device Controller Capability Parameters (USBHS_DCCPARAMS)

Not defined in the EHCI specification. Register describes the overall host/device capability of the USBHS module.

Address: USBHS_DCCPARAMS is 4003_4000h base + 124h offset = 4003_4124h

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																							HC	DC	0		DEN				
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	1	1	0	0	0	0	1	0	0

USBHS_DCCPARAMS field descriptions

Field	Description
31–9 Reserved	Reserved This read-only field is reserved and always has the value zero. always cleared.
8 HC	Host Capable Indicates the USBHS controller can operate as an EHCI compatible USB 2.0 host. Always reads as 0b1.
7 DC	Device Capable Indicates the USBHS controller can operate as an USB 2.0 device. Always set.
6–5 Reserved	Reserved This read-only field is reserved and always has the value zero. Always cleared
4–0 DEN	Device Endpoint Number This field indicates the number of endpoints built into the device controller. Always reads as 0b0100.

48.3.15 USB Command Register (USBHS_USBCMD)

The module executes the command indicated in this register.

Address: USBHS_USBCMD is 4003_4000h base + 140h offset = 4003_4140h

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								ITC								FS2	ATDTW	SUTW	0	ASPE	0	ASP		0	IAA	ASE	PSE	FS		RST	RS
W																																
Reset	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	0	0	0

USBHS_USBCMD field descriptions

Field	Description
31–24 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
23–16 ITC	Interrupt Threshold Control System software uses this field to set the maximum rate at which the module issues interrupts. ITC contains maximum interrupt interval measured in microframes. 0x00 Immediate (no threshold) 0x01 1 microframe 0x02 2 microframes 0x04 4 microframes 0x08 8 microframes 0x10 16 microframes 0x20 32 microframes 0x40 64 microframes Else Reserved
15 FS2	Frame list Size 2 See the FS bit description below. This is a non-EHCI bit.
14 ATDTW	Add dTD TripWire This is a non-EHCI bit. This bit is used as a semaphore when a dTD is added to an active (primed) endpoint. This bit is set and cleared by software. This bit is also cleared by hardware when the state machine is in a hazard region where adding a dTD to a primed endpoint may go unrecognized. More information appears in section “Executing a Transfer Descriptor.”
13 SUTW	Setup TripWire This is a non-EHCI bit. Used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by driver software without being corrupted. If the setup lockout mode is off (USBMODE[SLOM] = 1) then a hazard exists when new setup data arrives, and the software copies setup from the QH for a previous setup packet. This bit is set and cleared by software and is cleared by hardware when a hazard exists. More information appears in section “Control Endpoint Operation.”

Table continues on the next page...

USBHS_USBCMD field descriptions (continued)

Field	Description
12 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
11 ASPE	Asynchronous Schedule Park mode Enable Software uses this bit to enable or disable park mode. 0 Park mode disabled 1 Park mode enabled
10 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
9–8 ASP	Asynchronous Schedule Park mode count Contains a count of the successive transactions the host controller can execute from a high-speed queue head on the asynchronous schedule before continuing traversal of the asynchronous schedule. Valid values are 0b1 to 0b11. Software must not write a zero to this field when ASPE is set as this results in undefined behavior.
7 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
6 IAA	Interrupt on Async Advance doorbell Used as a doorbell by software to tell controller to issue an interrupt the next time it advances the asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When controller has evicted all appropriate cached schedule states, it sets USBSTS[AAI] register. If the USBINTR[AAE] bit is set, the host controller asserts an interrupt at the next interrupt threshold. The controller clears this bit after it has set the USBSTS[AAI] bit. Software must not write a 1 to this bit when the asynchronous schedule is inactive. Doing so yields undefined results. This bit used only in host mode. Writing a 1 to this bit when the USBHS module is in device mode has undefined results.
5 ASE	Asynchronous Schedule Enable Controls whether the controller skips processing the asynchronous schedule. Only used in host mode. 0 Do not process asynchronous schedule. 1 Use the ASYNCLISTADDR register to access asynchronous schedule.
4 PSE	Periodic Schedule Enable Controls whether the controller skips processing periodic schedule. Used only in host mode. 0 Do not process periodic schedule. 1 Use the PERIODICLISTBASE register to access the periodic schedule.
3–2 FS	Frame list Size With bit 15, these bits make the FS[2:0] field, which specifies the frame list size controlling which bits in the frame index register must be used for the frame list current index. Used only in host mode.

Table continues on the next page...

USBHS_USBCMD field descriptions (continued)

Field	Description
	NOTE: Values below 256 elements are not defined in the EHCI specification. 00 When FS2 = 0, the size is 1024 elements (4096 bytes). When FS2 = 1, the size is 64 elements (256 bytes). 01 When FS2 = 0, the size is 512 elements (2048 bytes). When FS2 = 1, the size is 32 elements (128 bytes). 10 When FS2 = 0, the size is 256 elements (1024 bytes). When FS2 = 1, the size is 16 elements (64 bytes). 11 When FS2 = 0, the size is 128 elements (512 bytes). When FS2 = 1, the size is 8 elements (32 bytes).
1 RST	Controller Reset Software uses this bit to reset controller. Controller clears this bit when reset process completes. Clearing this register does not allow software to terminate the reset process early. Host mode: When software sets this bit, the controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction in progress on the USB immediately terminates. A USB reset is not driven on downstream ports. Software must not set this bit when the USBSTS[HCH] bit is cleared. Attempting to reset an actively running host controller results in undefined behavior. Device mode: When software sets this bit, the controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Setting this bit with the device in the attached state is not recommended because it has an undefined effect on an attached host. To ensure the device is not in an attached state before initiating a device controller reset, all primed endpoints must be flushed and the USBCMD[RS] bit must be cleared.
0 RS	Run/Stop Host mode: When set, the controller proceeds with the execution of the schedule. The controller continues execution as long as this bit is set. When this bit is cleared, the controller completes the current transaction on the USB and then halts. The USBSTS[HCH] bit indicates when the host controller finishes the transaction and enters the stopped state. Software must not set this bit unless controller is in halted state (USBSTS[HCH] = 1). Device mode: Setting this bit causes the controller to enable a pull-up on DP and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up becomes disabled upon transitioning into high-speed mode. Software must use this bit to prevent an attach event before the USBHS controller has properly initialized. Clearing this bit causes a detach event.

48.3.16 USB Status Register (USBHS_USBSTS)

This register indicates various states of each module and any pending interrupts. This register does not indicate status resulting from a transaction on the serial bus. Software clears certain bits in this register by writing a 1 to them.

Address: USBHS_USBSTS is 4003_4000h base + 144h offset = 4003_4144h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						TI1	TI0	0				UPI	UAI	0	NAKI
W							w1c	w1c								
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	AS	PS	RCL	HCH	0	ULPII	0	SLI	SRI	URI	AAI	SEI	FRI	PCI	UEI	UI
W								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

USBHS_USBSTS field descriptions

Field	Description
31–26 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
25 TI1	General purpose Timer 1 Interrupt Set when the counter in the GPTIMER1CTRL register transitions to zero. Writing a one to this bit clears it. 0 No interrupt 1 Interrupt occurred
24 TI0	General purpose Timer 0 Interrupt Set when the counter in the GPTIMER0CTRL register transitions to zero. Writing a one to this bit clears it. 0 No interrupt 1 Interrupt occurred
23–20 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
19 UPI	USB host Periodic Interrupt

Table continues on the next page...

USBHS_USBSTS field descriptions (continued)

Field	Description
	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. **NOTE:** This bit is not used by the device controller and is always zero.
18 UAI	USB host Asynchronous Interrupt 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. **NOTE:** This bit is not used by the device controller and is always zero.
17 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
16 NAKI	NAK Interrupt Set by hardware for a particular endpoint when the TX/RX endpoint's NAK bit and the corresponding TX/RX endpoint's NAK enable bit are set. The hardware automatically clears this bit when all the enabled TX/RX endpoint NAK bits are cleared.
15 AS	Asynchronous schedule Status Reports the current real status of asynchronous schedule. Controller is not immediately required to disable or enable the asynchronous schedule when software transitions the USBCMD[ASE] bit. When this bit and the USBCMD[ASE] bit have the same value, the asynchronous schedule is enabled (1) or disabled (0). Used only in host mode. 0 Disabled 1 Enabled
14 PS	Periodic schedule Status Reports current real status of periodic schedule. Controller is not immediately required to disable or enable the periodic schedule when software transitions the USBCMD[PSE] bit. When this bit and the USBCMD[PSE] bit have the same value, the periodic schedule is enabled or disabled. Used only in host mode. 0 Disabled 1 Enabled
13 RCL	Reclamation Detects an empty asynchronous schedule. Used only by the host mode. 0 Non-empty asynchronous schedule 1 Empty asynchronous schedule

Table continues on the next page...

USBHS_USBSTS field descriptions (continued)

Field	Description
12 HCH	Host Controller Halted This bit is cleared when the USBCMD[RS] bit is set. The controller sets this bit after it stops executing because of the USBCMD[RS] bit being cleared, by software or the host controller hardware (for example, internal error). Used only in host mode. 0 Running 1 Halted
11 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
10 ULPII	ULPI Interrupt Set by event completion.
9 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
8 SLI	Device-controller suspend Non-EHCI bit. When a device controller enters a suspend state from an active state, this bit is set. The device controller clears the bit upon exiting from a suspend state. Used only by the device controller. 0 Active 1 Suspended
7 SRI	SOF Received This is a non-EHCI status bit. Software writes a 1 to this bit to clear it. Host mode: In host mode, this bit is set every 125 ms, provided PHY clock is present and running (for example, the port is NOT suspended) and can be used by the host-controller driver as a time base. Device mode: When controller detects a start of (micro) frame, bit is set. When a SOF is extremely late, controller automatically sets this bit to indicate an SOF was expected. Therefore, this bit is set roughly every 1 ms in device FS mode and every 125 us in HS mode, and it is synchronized to the actual SOF received. Because the controller is initialized to FS before connect, this bit is set at an interval of 1 ms during the prelude to the connect and chirp.
6 URI	USB Reset received A non-EHCI bit. When the controller detects a USB reset and enters the default state, this bit is set. Software can write a 1 to this bit to clear it. Used only by in device mode. 0 No reset received 1 Reset received
5 AAI	Interrupt on Async Advance By setting the USBCMD[IAA] bit, system software can force the controller to issue an interrupt the next time the controller advances the asynchronous schedule. This status bit indicates the assertion of that interrupt source. Used only by the host mode.

Table continues on the next page...

USBHS_USBSTS field descriptions (continued)

Field	Description
	0 No async advance interrupt 1 Async advance interrupt
4 SEI	System Error Set when an error is detected on the system bus. If the system error enable bit (USBINTR[SEE]) is set, interrupt generates. The interrupt and status bits remain set until cleared by writing a 1 to this bit. Additionally, when in host mode, the USBCMD[RS] bit is cleared, effectively disabling controller. An interrupt generates for the USBHS controller in device mode, but no other action is taken. 0 Normal operation 1 Error
3 FRI	Frame-list Rollover Controller sets this bit when the frame list index (FRINDEX) rolls over from its maximum value to 0. The exact value the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the USBCMD[FS] field) is 1024, the frame index register rolls over every time FRINDEX[13] toggles. Similarly, if the size is 512, the controller sets this bit each time FRINDEX[12] toggles. Used only in the host mode.
2 PCI	Port Change detect This bit is not EHCI compatible. Host mode: Controller sets this bit when a connect status occurs on any port, a port enable/disable change occurs, an over-current change occurs, or the force port resume (PORTSCn[FPR]) bit is set as the result of a J-K transition on the suspended port. Device mode: The controller sets this bit when it enters the full- or high-speed operational state. When it exits the full- or high-speed operation states due to reset or suspend events, the notification mechanisms are URI and SLI bits respectively. The device controller detects resume signaling only.
1 UEI	USB Error Interrupt When completion of USB transaction results in error condition, the controller sets this bit. If the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set, this bit is set along with the USBINT bit. See Section 4.15.1 in the EHCI specification for a complete list of host error interrupt conditions. See Table 24-62 for more information on device error matrix. 0 No error 1 Error detected
0 UI	USB Interrupt (USBINT) This bit is set by the controller when the cause of an interrupt is a completion of a USB transaction where the TD has an interrupt on complete (IOC) bit set. This bit is also set by the 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.

48.3.17 USB Interrupt Enable Register (USBHS_USBINTR)

The interrupts to software are enabled with this register. An interrupt generates when a bit is set and the corresponding interrupt is active. The USB status register (USBSTS) continues to show interrupt sources (even if the USBINTR register disables them), allowing polling of interrupt events by the software.

Address: USBHS_USBINTR is 4003_4000h base + 148h offset = 4003_4148h

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						TIE1	TIE0	0				UPIE	UAIE	0	NAKE	0					ULPIE	0	SLE	SRE	URE	AAE	SEE	FRE	PCE	UEE	UE
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

USBHS_USBINTR field descriptions

Field	Description
31–26 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
25 TIE1	General purpose Timer 1 Interrupt Enable When this bit and USBSTS[GPTINT1] are set, the USB controller issues an interrupt to the processor. The interrupt is acknowledged by clearing GPTINT1. 0 Disabled 1 Enabled
24 TIE0	General purpose Timer 0 Interrupt Enable When this bit and USBSTS[GPTINT0] are set, the USB controller issues an interrupt to the processor. The interrupt is acknowledged by clearing GPTINT0. 0 Disabled 1 Enabled
23–20 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
19 UPIE	USB host Periodic Interrupt Enable When this bit and USBSTS[USBHSTPERINT] are set, the host controller issues an interrupt at the next interrupt threshold. The interrupt is acknowledged by clearing USBHSTPERINT.
18 UAIE	USB host Asynchronous Interrupt Enable When this bit and USBSTS[USBHSTASYNCINT] are set, the host controller issues an interrupt at the next interrupt threshold. The interrupt is acknowledged by clearing USBHSTASYNCINT.

Table continues on the next page...

USBHS_USBINTR field descriptions (continued)

Field	Description
17 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
16 NAKE	NAK Interrupt Enable When this bit and the USBSTS[NAKI] bit are set, an interrupt generates. 0 Disabled 1 Enabled
15–11 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
10 ULPIE	ULPI Enable When this bit and USBSTS[ULPII] are set, controller issues an interrupt. The interrupt is acknowledged by writing a 1 to USBSTS[ULPII].
9 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
8 SLE	Sleep (DC suspend) Enable A non-EHCI bit. When this bit is set and the USBSTS[SLI] bit transitions, USBHS controller issues an interrupt. Software writing a 1 to the USBSTS[SLI] bit acknowledges the interrupt. Used only in device mode. 0 Disabled 1 Enabled
7 SRE	SOF-Received Enable This is a non-EHCI bit. When this bit and the USBSTS[SRI] bit are set, controller issues an interrupt. Software clearing the USBSTS[SRI] bit acknowledges the interrupt. 0 Disabled 1 Enabled
6 URE	USB-Reset Enable A non-EHCI bit. When this bit and the USBSTS[URI] bit are set, device controller issues an interrupt. Software clearing the USBSTS[URI] bit acknowledges the interrupt. Used only in device mode. 0 Disabled 1 Enabled
5 AAE	Interrupt on Async advance Enable When this bit and the USBSTS[AAI] bit are set, controller issues an interrupt at the next interrupt threshold. Software clearing the USBSTS[AAI] bit acknowledges the interrupt. 0 Disabled 1 Enabled

Table continues on the next page...

USBHS_USBINTR field descriptions (continued)

Field	Description
4 SEE	System Error Enable When this bit and the USBSTS[SEI] bit are set, controller issues an interrupt. Software clearing the USBSTS[SEI] bit acknowledges the interrupt. 0 Disabled 1 Enabled
3 FRE	Frame list Rollover Enable When this bit and the USBSTS[FRI] bit are set, controller issues an interrupt. Software clearing the USBSTS[FRI] bit acknowledges the interrupt. Used only in host mode. 0 Disabled 1 Enabled
2 PCE	Port Change detect Enable When this bit and the USBSTS[PCI] bit are set, controller issues an interrupt. Software clearing the USBSTS[PCI] bit acknowledges the interrupt. 0 Disabled 1 Enabled
1 UEE	USB Error interrupt Enable When this bit and the USBSTS[UEI] bit are set, controller issues an interrupt at the next interrupt threshold. Software clearing the USBSTS[UEI] bit acknowledges the interrupt. 0 Disabled 1 Enabled
0 UE	USB interrupt Enable When this bit is 1 and the USBSTS[UI] bit is set, the USBHS controller issues an interrupt at the next interrupt threshold. Software clearing the USBSTS[UI] bit acknowledges the interrupt. 0 Disabled 1 Enabled

48.3.18 Frame Index Register (USBHS_FRINDEX)

In host mode, the controller uses this register to index the periodic frame list. The register updates every 125 microseconds (once each microframe). Bits [N–3] 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 USBCMD[FS] field.

Only perform 32-bit accesses to this register. Byte writes produce undefined results. This register cannot be written unless the USBHS controller is in halted state as the USBSTS[HCH] bit indicates. A write to this register while the USBSTS[RS] bit is set produces undefined results. Writes to this register also affect the SOF value.

In device mode, this register is read-only, and the USBHS controller updates the FRINDEX[13–3] bits from the frame number the SOF marker indicates. When the USB bus receives a SOF, FRINDEX[13–3] checks against the SOF marker. If FRINDEX[13–3] is different from the SOF marker, FRINDEX[13–3] is set to the SOF value and FRINDEX[2–0] is cleared (SOF for 1 ms frame). If FRINDEX[13–3] equals the SOF value, FRINDEX[2–0] is incremented (SOF for 125 microsec microframe.)

Address: USBHS_FRINDEX is 4003_4000h base + 14Ch offset = 4003_414Ch

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																		FRINDEX													
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

USBHS_FRINDEX field descriptions

Field	Description
31–14 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
13–0 FRINDEX	Frame Index The value in this register increments at the end of each time frame (microframe). Bits [N– 3] are for the frame list current index. This means each location of the frame list is accessed 8 times per frame (once each microframe) before moving to the next index. In device mode, the value is the current frame number of the last frame transmitted and not used as an index. In either mode, bits 2–0 indicate current microframe. The table illustrates values of N based on the value of the USBCMD[FS] field when used in host mode.

Table 48-27. FRINDEX N Values

USBCMD[FS]	Frame List Size	FRINDEX N value
000	1024 elements (4096 bytes)	12
001	512 elements (2048 bytes)	11
010	256 elements (1024 bytes)	10
011	128 elements (512 bytes)	9
100	64 elements (256 bytes)	8
101	32 elements (128 bytes)	7
110	16 elements (64 bytes)	6
111	8 elements (32 bytes)	5

48.3.19 Periodic Frame List Base Address Register (USBHS_PERIODICLISTBASE)

This register contains the beginning address of the periodic frame list in the system memory. The host controller driver loads this register prior to starting the schedule execution by the controller. The memory structure referenced by this physical memory pointer assumes to be 4-Kbyte aligned. The contents combine with the FRINDEX register to enable the controller to step through the periodic frame list in sequence.

The host and device mode functions share this register. In host mode, it is the PERIODICLISTBASE register; in device mode, it is the DEVICEADDR register. See section “Device Address Register (DEVICEADDR),” for more information.

Address: USBHS_PERIODICLISTBASE is 4003_4000h base + 154h offset = 4003_4154h

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	PERBASE																				0											
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

USBHS_PERIODICLISTBASE field descriptions

Field	Description
31–12 PERBASE	Base address These bits correspond to memory address signal [31:12]. Used only in the host mode.
11–0 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.

48.3.20 Device Address Register (USBHS_DEVICEADDR)

This register is not defined in the EHCI specification. For device mode, the upper seven bits of this register represent the device address. After any controller or USB reset, the device address is set to the default address (0). The default address matches all incoming addresses. Software reprograms the address after receiving a SET_ADDRESS descriptor.

The host and device mode functions share this register. In device mode, it is the DEVICEADDR register; in host mode, it is the PERIODICLISTBASE register. See section “Periodic Frame List Base Address Register (PERIODICLISTBASE),” for more information.

Address: USBHS_DEVICEADDR is 4003_4000h base + 154h offset = 4003_4154h

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

USBHS_DEVICEADDR field descriptions

Field	Description
31–25 USBADR	Device Address This field corresponds to the USB device address.
24 USBADRA	Device Address Advance This field provides a mechanism for staging the device address in advance. After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism ensures this specification is met when the DCD cannot write to the device address within 2 ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA equaling 1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR is programmed instantly at the correct time and meets the 2 ms USB requirement. Hardware automatically clears this bit on the following conditions: • IN is ACKed to endpoint 0. (USBADR is updated from staging register.) • OUT/SETUP occur to endpoint 0. (USBADR is not updated.) • Device Reset occurs. (USBADR is reset to 0.) 0 Writes to USBADR are instantaneous. 1 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 is loaded from the holding register.
23–0 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.

48.3.21 Current Asynchronous List Address Register (USBHS_ASYNCLISTADDR)

The ASYNCLISTADDR register contains the address of the next asynchronous queue head to executed by the host.

The host and device mode functions share this register. In host mode, it is the ASYNCLISTADDR register; in device mode, it is the EPLISTADDR register. See section “Endpoint List Address Register (EPLISTADDR),” for more information.

Address: USBHS_ASYNCLISTADDR is 4003_4000h base + 158h offset = 4003_4158h

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	ASYBASE																											0				
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

USBHS_ASYNCLISTADDR field descriptions

Field	Description
31–5 ASYBASE	Link pointer low (LPL) These bits correspond to memory address signal [31:5]. This field may only reference a queue head (QH). Used only in host mode.
4–0 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.

48.3.22 Endpoint List Address Register (USBHS_EPLISTADDR)

This register is not defined in the EHCI specification. For device mode, this register contains the address of the endpoint list top in system memory. The memory structure referenced by this physical memory pointer assumes to be 64-bytes. The queue head is actually a 48-byte structure, but must be aligned on 64-byte boundary. However, the EPBASE field has a granularity of 2 Kbytes; in practice, the queue head should be 2-Kbyte aligned.

The host and device mode functions share this register. In device mode, it is the EPLISTADDR register; in host mode, it is the ASYNCLISTADDR register. See section “Current Asynchronous List Address Register (ASYNCLISTADDR),” for more information.

Address: USBHS_EPLISTADDR is 4003_4000h base + 158h offset = 4003_4158h

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	EPBASE																					0										
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

USBHS_EPLISTADDR field descriptions

Field	Description
31–11 EPBASE	Endpoint list address Correspond to memory address signals [31:11] References a list of up to 32 queue heads (i.e. one queue head per endpoint and direction). Address of the top of the endpoint list.
10–0 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.

48.3.23 Host TT Asynchronous Buffer Control (USBHS_TTCTRL)

Address: USBHS_TTCTRL is 4003_4000h base + 15Ch offset = 4003_415Ch

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	TTHA							0																							
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

USBHS_TTCTRL field descriptions

Field	Description
31 Reserved	Reserved This read-only field is reserved and always has the value zero. Must ne cleared.
30–24 TTHA	TT Hub Address This field is used to match against the Hub Address field in a QH or siTD to determine if the packet is routed to the internal TT for directly attached FS/LS devices. If the hub address in the QH or siTD does not match this address then the packet is broadcast on the high speed ports destined for a downstream HS hub with the address in the QH or siTD.
23–0 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.

48.3.24 Master Interface Data Burst Size Register (USBHS_BURSTSIZE)

This register is not defined in the EHCI specification. BURSTSIZE dynamically controls the burst size during data movement on the initiator (master) interface.

Address: USBHS_BURSTSIZE is 4003_4000h base + 160h offset = 4003_4160h

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																TXPBURST								RXPBURST							
W																																
Reset	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	0	0	0	0

USBHS_BURSTSIZE field descriptions

Field	Description
31–16 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
15–8 TXPBURST	Programable TX Burst length Represents the maximum length of a burst in 32-bit words while moving data from system memory to the USB bus. Must not be set to greater than 16. If the BURSTSIZE field of register SBUSCFG is non-zero, the TXPBURST field returns the value of the INCRx length.
7–0 RXPBURST	Programable RX Burst length This register represents the maximum length of a burst in 32-bit words while moving data from the USB bus to system memory. Must not be set to greater than 16. If the BURSTSIZE field of register SBUSCFG is non-zero, the RXPBURST field returns the value of the INCRx length.

48.3.25 Transmit FIFO Tuning Control Register (USBHS_TXFILLTUNING)

This register is not defined in the EHCI specification. The TXFILLTUNING register controls performance tuning associated with how the module posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include how much data to post into the FIFO and an estimate for how long that operation takes in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_s = Total packet flight time (send-only) packet ($T_s = T_0 + T_1$)

T_{ff} Time to fetch packet into TX FIFO up to specified level

T_p Total packet time (fetch and send) packet ($T_p = T_{ff} + T_s$)

Upon discovery of a transmit (OUT/SETUP) packet in the data structures, the host controller checks to ensure T_p remains before the end of the (micro)frame. If so, it pre-fills the TX FIFO. If at anytime during the pre-fill operation the time remaining the (micro)frame is less than T_s , packet attempt ceases and tries at a later time. Although this is not an error condition and the module eventually recovers, a mark is made in the scheduler health counter to mark 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 beginning after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and should be minimized (not necessarily eliminated). The TSCHHEALTH ($T_{ff)}$ parameter described below can minimize back-offs.

Address: USBHS_TXFILLTUNING is 4003_4000h base + 164h offset = 4003_4164h

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										TXFIFOTHRES						0			TXSCHHEALTH					0	TXSCHOH						
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

USBHS_TXFILLTUNING field descriptions

Field	Description
31–22 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
21–16 TXFIFOTHRES	FIFO burst Threshold FIFO burst threshold. Controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on the bus. The minimum value is 2 and this value should be as low as possible to maximize USB performance. Systems with unpredictable latency and/or insufficient bandwidth can use a higher value where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can replenish from system memory. This value is ignored if the USBMODE[SDIS] bit is set. When the USBMODE[SDIS] bit is set, the host controller behaves as if TXFIFOTHRES is set to its maximum value.
15–13 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.

Table continues on the next page...

USBHS_TXFILLTUNING field descriptions (continued)

Field	Description
12–8 TXSCHHEALTH	Scheduler Health counter These bits increment 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 SOF. This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register clears the counter and this counter stops counting after reaching the maximum of 31.
7 Reserved	Reserved This read-only field is reserved and always has the value zero.
6–0 TXSCHOH	Scheduler Overhead These bits add an additional fixed offset to the schedule time estimator described as T_{ff} . As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH field to less than 10 per second in a highly utilized bus. Choosing a value too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267 ms when a device connects in high-speed mode. The time unit represented in this register is 6.333 ms when a device connects in low-/full-speed mode. For most applications, TXSCHOH can be set to 4 or less. A good value to begin with is: (TXFIFOTHRES x BURSTSIZE x 4) / (40 x TimeUnit) Always rounded to the next higher integer. TimeUnit is 1.267 or 6.333 as noted earlier in this description. For example, if TXFIFOTHRES is 5 and BURSTSIZE is 8, set TXSCHOH to 5¥(8¥4)/(40¥1.267) equals 4 for a high-speed link. If this value of TXSCHOH results in a TXSCHHEALTH count of 0 per second, low the value by 1 if optimizing performance is desired. If TXSCHHEALTH exceeds 10 per second, raise the value by 1. If streaming mode is disabled via the USBMODE register, treat TXFIFOTHRES as the maximum value for purposes of the TXSCHOH calculation.

48.3.26 ULPI Register Access (USBHS_ULPI_VIEWPORT)

The register provides indirect access to the ULPI PHY register set. Although the controller modules perform access to the ULPI PHY register set, there may be circumstances where software may need direct access.

NOTE

Be advised that writes to the ULPI through the ULPI viewport can substantially harm standard USB operations. Currently no usage model has been defined where software should need to execute writes directly to the ULPI. Executing read operations though the ULPI viewport should have no harmful side effects to standard USB operations. Also, if the ULPI interface is not enabled, this register is always read cleared.

There are two operations that can be performed with the ULPI viewport, wake-up and read/write operations. The wake-up operation is used to put the ULPI interface into normal operation mode and re-enable the clock if necessary. A wake-up operation is required before accessing the registers when the ULPI interface is operating in low power mode, serial mode, or carkit mode. The ULPI state can be determined by reading the sync state bit (ULPI_SS). If this bit is set, then the ULPI interface is running in normal operating mode and can accept read/write operations. If ULPI_SS is cleared, then read/ write operations are not executed. Undefined behavior results if a read or write operation is performed when ULPI_SS is cleared. To execute a wake-up operation, write all 32-bits of the ULPI VIEWPORT where ULPI_PORT is constructed appropriately and the ULPI_WU bit is set and the ULPI_RUN bit is cleared. Poll the ULPI VIEWPORT until ULPI_WU is cleared for the operation to complete.

To execute a read or write operation, write all 32-bits of the ULPI VIEWPORT where ULPI_DATWR, ULPI_ADDR, ULPI_PORT, ULPI_RW are constructed appropriately and the ULPI_RUN bit is set. Poll the ULPI VIEWPORT until ULPI_RUN is cleared for the operation to complete. For read operations, ULPI_DATRD is valid after ULPI_RUN is cleared.

The polling method above can be replaced with interrupts using the ULPI interrupt defined in the USBSTS and USBINTR registers. When a wake-up or read/write operation completes, the ULPI interrupt is set.

Address: USBHS_ULPI_VIEWPORT is 4003_4000h base + 170h offset = 4003_4170h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ULPI_WU	ULPI_RUN	ULPI_RW	0	ULPI_SS	ULPI_PORT			ULPI_ADDR							
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	ULPI_DATRD								ULPI_DATWR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBHS_ULPI_VIEWPORT field descriptions

Field	Description
31 ULPI_WU	ULPI Wake-Up Setting this bit begins the wake-up operation. This bit automatically clears after the wake-up is complete. After this bit is set, it can not be cleared by software. **NOTE:** The driver must never execute a wake-up and a read/write operation at the same time.

Table continues on the next page...

USBHS_ULPI_VIEWPORT field descriptions (continued)

Field	Description
30 ULPI_RUN	ULPI Run Setting this bit begins a read/write operation. This bit automatically clears after the read/write is complete. After this bit is set, it can not be cleared by software. **NOTE:** The driver must never execute a wake-up and a read/write operation at the same time.
29 ULPI_RW	ULPI Read/Write Selects between running a read or write operation to the ULPI. 0 Read 1 Write
28 Reserved	Reserved This read-only field is reserved and always has the value zero. Should be cleared
27 ULPI_SS	ULPI Sync State Represents the state of the ULPI interface. Before reading this bit, the ULPI_PORT field should be set accordingly if used with the multi-port host. Otherwise, this field should always remain 0. 0 Any other state (that is, carkit, serial, low power) 1 Normal sync state
26–24 ULPI_PORT	ULPI Port number For wake-up or read/write operations this value selects the port number to which the ULPI PHY is attached. Valid values are 0 and 1.
23–16 ULPI_ADDR	ULPI data Address When a read or write operation is commanded, the address of the operation is written to this field.
15–8 ULPI_DATRD	ULPI Data Read After a read operation completes, the result is placed in this field.
7–0 ULPI_DATWR	ULPI Data Write When a write operation is commanded, the data to be sent is written to this field.

48.3.27 Endpoint NAK Register (USBHS_ENDPTNAK)

Address: USBHS_ENDPTNAK is 4003_4000h base + 178h offset = 4003_4178h

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												EPTN				0												EPRN			
W													w1c																w1c			
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

USBHS_ENDPTNAK field descriptions

Field	Description
31–20 Reserved	Reserved This read-only field is reserved and always has the value zero.
19–16 EPTN	TX Endpoint NAK 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. • EPTN[3]—Endpoint #3 • EPTN[2]—Endpoint #2 • EPTN[1]—Endpoint #1 • EPTN[0]—Endpoint #0
15–4 Reserved	Reserved This read-only field is reserved and always has the value zero.
3–0 EPRN	RX Endpoint NAK 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. • EPRN[3]—Endpoint #3 • EPRN[2]—Endpoint #2 • EPRN[1]—Endpoint #1 • EPRN[0]—Endpoint #0

48.3.28 Endpoint NAK Enable Register (USBHS_ENDPTNAKEN)

Address: USBHS_ENDPTNAKEN is 4003_4000h base + 17Ch offset = 4003_417Ch

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												EPTNE				0												EPRNE			
W													w1c																w1c			
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

USBHS_ENDPTNAKEN field descriptions

Field	Description
31–20 Reserved	Reserved This read-only field is reserved and always has the value zero.
19–16 EPTNE	TX Endpoint NAK 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. • EPTNE[3]—Endpoint #3 • EPTNE[2]—Endpoint #2 • EPTNE[1]—Endpoint #1 • EPTNE[0]—Endpoint #0

Table continues on the next page...

USBHS_ENDPTNAKEN field descriptions (continued)

Field	Description
15–4 Reserved	Reserved This read-only field is reserved and always has the value zero.
3–0 EPRNE	RX Endpoint NAK 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. • EPRNE[3]—Endpoint #3 • EPRNE[2]—Endpoint #2 • EPRNE[1]—Endpoint #1 • EPRNE[0]—Endpoint #0

48.3.29 Configure Flag Register (USBHS_CONFIGFLAG)

This EHCI register is not used in this implementation. A read from this register returns a constant of a 0x0000_0001 to indicate that all port routings default to this host controller.

Address: USBHS_CONFIGFLAG is 4003_4000h base + 180h offset = 4003_4180h

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																															1
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	1

USBHS_CONFIGFLAG field descriptions

Field	Description
31–1 Reserved	Reserved This read-only field is reserved and always has the value zero. (0x0000_0001, all port routings default to this host)
0 Reserved	Reserved This read-only field is reserved and always has the value one. (0x0000_0001, all port routings default to this host)

48.3.30 Port Status and Control Registers (USBHS_PORTSC1)

The USB module contains a single PORTSC register. This register only resets when power is initially applied or in response to a controller reset. Initial conditions of a port are:

- No device connected
- Port disabled

If the port has port power control, this state remains until software applies power to the port by setting port power to one.

For the USBHS module in device mode, the USBHS controller does not support power control. Port control in device mode is used only for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling, and allows software to place the PHY into low-power suspend mode and disable the PHY clock.

Address: USBHS_PORTSC1 is 4003_4000h base + 184h offset = 4003_4184h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PTS		0	0	PSPD		0	PFSC	PHCD	WKOC	WKDS	WKCN	PTC			
W																
Reset	1	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	PIC		PO	PP	LS		HSP	PR	SUSP	FPR	OCC	OCA	PEC	PE	CSC	CCS
W											w1c		w1c		w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBHS_PORTSC1 field descriptions

Field	Description
31–30 PTS	Port Transceiver Select Controls which parallel transceiver interface is selected. 10 ULPI parallel interface All other values are reserved. This field is not defined in the EHCI specification.
29 Reserved	Reserved This read-only field is reserved and always has the value zero.
28 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be Cleared
27–26 PSPD	Port Speed This read-only register field indicates the speed the port operates. This bit is not defined in the EHCI specification. 00 Full speed 01 Low speed

Table continues on the next page...

USBHS_PORTSC1 field descriptions (continued)

Field	Description
	10 High speed 11 Undefined
25 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
24 PFSC	Port force Full-Speed Connect Disables the chirp sequence that allows the port to identify itself as a HS port. useful for testing FS configurations with a HS host, hub, or device. Not defined in the EHCI specification. This bit is for debugging purposes. 0 Allow the port to identify itself as high speed 1 Force the port to only connect at full speed
23 PHCD	PHY low power suspend This bit is not defined in the EHCI specification. Host mode: The PHY can be placed into low-power suspend when downstream device is put into suspend mode or when no downstream device connects. Software completely controls low-power suspend. Device mode: For the USBHS module in device mode, the PHY can be put into low power suspend when the device is not running (USBCMD[RS] = 0) or suspend signaling is detected on the USB. The PHCD bit is cleared automatically when the resume signaling is detected or when forcing port resumes. Reading this bit indicates the status of the PHY.
22 WKOC	Wake on Over-Current enable Enables the port to be sensitive to over-current conditions as wake-up events. This field is 0 if the PP bit is cleared. In host mode, this bit can work with an external power control circuit.
21 WKDS	Wake on Disconnect enable Enables the port to be sensitive to device disconnects as wake-up events. This field is 0 if the PP bit is cleared or the module is in device mode. In host mode, this bit can work with an external power control circuit.
20 WKCN	Wake on Connect enable Enables the port to be sensitive to device connects as wake-up events. This field is 0 if the PP bit is cleared or the module is in device mode. In host mode, this can work with an external power control circuit.
19–16 PTC	Port Test Control Any value other than 0 indicates the port operates in test mode. Refer to Chapter 7 of the USB Specification Revision 2.0 for details on each test mode. **NOTE:** The FORCE_ENABLE_FS and FORCE ENABLE_LS settings are extensions to the test mode support in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE values forces the port into the connected and enabled state at the selected speed. Then clearing the PTC field allows the port state machines to progress normally from that point.

Table continues on the next page...

USBHS_PORTSC1 field descriptions (continued)

Field	Description
	0000 Not enabled 0001 J_STATE 0010 K_STATE 0011 SEQ_NAK 0100 Packet 0101 FORCE_ENABLE_HS 0110 FORCE_ENABLE_FS 0111 FORCE_ENABLE_LS Else Reserved
15–14 PIC	Port Indicator Control Not supported on this module.
13 PO	Port Owner Port owner handoff is not implemented in this device.
12 PP	Port Power Represents the current setting of the port power control switch (0 equals off, 1 equals on). When power is not available on a port (PP = 0), it is non-functional and does not report attaches, detaches, etc. When an over-current condition is detected on a powered port, the host controller driver from a 1to a 0 (removing power from the port) transitions the PP bit in each affected port.
11–10 LS	Line Status Reflects current logical levels of the USB DP (bit 11) and DM (bit 10) signal lines. In host mode, the line status by the host controller driver is not necessary (unlike EHCI) because hardware manages the connection of FS and LS. In device mode, LS by the device controller is not necessary. 00 SE0 01 J-state 10 K-state 11 Undefined
9 HSP	High Speed Port. Indicates if the host/device connected is in high speed mode. **NOTE:** This bit is redundant with the PSPD bit field. 0 FS or LS 1 HS
8 PR	Port Reset This field is cleared if the PP bit is cleared. Host mode: When software sets this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 starts. This bit automatically clears after the reset sequence completes. This behavior is different from EHCI where the host controller driver is required to clear this bit after the reset duration is timed in the driver. Device mode: This bit is a read-only status bit. Device reset from the USB bus is also indicated in the USBSTS register.

Table continues on the next page...

USBHS_PORTSC1 field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>0 Port is not in reset
1 Port is in reset</td></tr>
<tr><td>7
SUSP</td><td>Suspend
Host mode:
The PE and SUSP bits define the port state as follows:
<table>
<tr><th>PE</th><th>SUSP</th><th>Port State</th></tr>
<tr><td>0</td><td>x</td><td>Disable</td></tr>
<tr><td>1</td><td>0</td><td>Enable</td></tr>
<tr><td>1</td><td>1</td><td>Suspend</td></tr>
</table>
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 set. In the suspend state, the port is sensitive to resume detection. The bit status does not change until the port is suspended and there may be a delay in suspending a port if there is a transaction currently in progress on the USB.
The module unconditionally clears this bit when software clears the FPR bit. The host controller ignores clearing this bit. If host software sets this bit when the port is not enabled (PE = 0), the results are undefined.
This bit is cleared if the PP bit is cleared in host mode.
Device mode:
In device mode, this bit is a read-only status bit.
0 Port not in suspend state
1 Port in suspend state</td></tr>
<tr><td>6
FPR</td><td>Force Port Resume
This bit is not-EHCI compatible.
Host mode:
Software sets this bit to drive resume signaling. The controller sets this bit if a J-to-K transition is detected while the port is in suspend state (PE = SUSP = 1), which in turn sets the USBSTS[PCI] bit. This bit automatically clears after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to clear this bit after the resume duration is timed in the driver.
When the 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 set. This bit remains set until the port switches to the high-speed idle. Clearing this bit has no affect because the port controller times the resume operation to clear the bit the port control state switches to HS or FS idle.
This bit is cleared if the PP bit is cleared in host mode.
Device mode:
If remote wakeup is enabled, after the device is in suspend for 5 ms or more, software can set this bit to drive resume signaling before clearing. The device controller sets this bit if a J-to-K transition is detected while port is in suspend state, which in turn sets the USBSTS[PCI] bit. The bit is cleared when the device returns to normal operation.</td></tr>
</table>

Table continues on the next page...

USBHS_PORTSC1 field descriptions (continued)

Field	Description
	0 No resume (K-state) detected/driven on port 1 Resume detected/driven on port
5 OCC	Over-Current Change Indicates a change to the OCA bit. Software clears this bit by writing a 1. For host mode, the user can provide over-current detection to the USBn_PWRFAULT signal for this condition. For device-only implementations, this bit must always be cleared. 0 No over-current 1 Over-current detect
4 OCA	Over-current active This bit automatically transitions from 1 to 0 when the over-current condition is removed. For host/OTG implementations, the user can provide over-current detection to the USBn_PWRFAULT signal for this condition. For device-only implementations, this bit must always be cleared. 0 Port not in over-current condition 1 Port currently in over-current condition
3 PEC	Port Enable/disable Change For the root hub, this bit gets set 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 1 to it. In device mode, the device port is always enabled. (This bit is zero). This bit is cleared if the PP bit is cleared. 0 No change 1 Port disabled
2 PE	Port Enabled/disabled Host mode: Ports can only be enabled by the controller as a part of the reset and enable sequence. Software cannot enable a port by setting this bit. A fault condition (disconnect event or other fault condition) or host software can disable ports. 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 and bus events. When the port is disabled, downstream propagation of data is blocked except for reset. This field is cleared if the PP bit is cleared in host mode. Device mode: The device port is always enabled. (This bit is set).
1 CSC	Connect Change Status Host mode: This bit indicates a change occurred in the port's current connect status. The 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 is setting an already-set bit (i.e., the bit remains set). Software clears this bit by writing a 1 to it. This field is cleared if the PP bit is cleared. Device mode:

Table continues on the next page...

USBHS_PORTSC1 field descriptions (continued)

Field	Description
	In device mode, this bit is undefined. 0 No change 1 Connect status has changed
0 CCS	Current Connect Status Indicates that a device successfully attaches and operates in high speed or full speed as indicated by the PSPD bit. If clear, the device did not attach successfully or forcibly disconnects by the software clearing the USBCMD[RUN] bit. It does not state the device disconnected or suspended. This bit is cleared if the PP bit is cleared in host mode. 0 No device present (host mode) or attached (device mode) 1 Device is present (host mode) or attached (device mode)

48.3.31 On-the-Go Status and Control Register (USBHS_OTGSC)

This register is not defined in the EHCI specification. The host controller implements one OTGSC register corresponding to port 0 of the host controller.

The OTGSC register 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 de-bounce using a 1 ms time constant. Values on the status inputs that do not persist for more than 1 ms do not cause an update of the status inputs or an OTG interrupt.

Address: USBHS_OTGSC is 4003_4000h base + 1A4h offset = 4003_41A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DPIE	MSE	BSEIE	BSVIE	ASVIE	AVVIE	IDIE	0	DPIS	MSS	BSEIS	BSVIS	ASVIS	AVVIS	IDIS
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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DPS	MST	BSE	BSV	ASV	AVV	ID	HABA	0	IDPU	DP	OT	HAAR	VC	VD
W																
Reset	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0

USBHS_OTGSC field descriptions

Field	Description
31 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
30 DPIE	Data Pulse Interrupt Enable 0 Disable 1 Enable
29 MSE	1 Milli-Second timer interrupt Enable 0 Disable 1 Enable
28 BSEIE	B Session End Interrupt Enable 0 Disable 1 Enable
27 BSVIE	B Session Valid Interrupt Enable 0 Disable 1 Enable
26 ASVIE	A Session Valid Interrupt Enable 0 Disable 1 Enable
25 AVVIE	A VBUS Valid Interrupt Enable 0 Disable 1 Enable

Table continues on the next page...

USBHS_OTGSC field descriptions (continued)

Field	Description
24 IDIE	USB ID Interrupt Enable 0 Disable 1 Enable
23 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
22 DPIS	Data Pulse interrupt Status Indicates when data bus pulsing occurs on DP or DM. Data bus pulsing only detected when USBMODE[CM] equals 11 and PORTSC0[PP] is cleared. Software must write a 1 to clear this bit.
21 MSS	1 Milli-Second timer interrupt Status This bit is set once every millisecond. Software must write a 1 to clear this bit.
20 BSEIS	B Session End Interrupt Status Indicates when VBUS falls below the B session end threshold. Software must write a 1 to clear this bit.
19 BSVIS	B Session Valid Interrupt Status Indicates when VBUS rises above or falls below the B session valid threshold (0.8 VDC). Software must write a 1 to clear this bit.
18 ASVIS	A Session Valid Interrupt Status Indicates when VBUS rises above or falls below the A session valid threshold (0.8 VDC). Software must write a 1 to clear this bit.
17 AVVIS	A VBUS Valid Interrupt Status Indicates when VBUS rises above or falls below the VBUS valid threshold (4.4 VDC) on an A device. Software must write a 1 to clear this bit.
16 IDIS	USB ID Interrupt Status Indicates when a change on the ID input is detected. Software must write a 1 to clear this bit.
15 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
14 DPS	Data bus Pulsing Status 0 No pulsing on port 1 Pulsing detected on port
13 MST	1 Milli-Second timer Toggle This bit toggles once per millisecond.
12 BSE	B Session End 0 VBus is above B session end threshold 1 VBus is below B session end threshold
11 BSV	B Session Valid

Table continues on the next page...

USBHS_OTGSC field descriptions (continued)

Field	Description
	0 VBus is below B session valid threshold 1 VBus is above B session valid threshold
10 ASV	A Session Valid 0 VBus is below A session valid threshold 1 VBus is above A session valid threshold
9 AVV	A VBus Valid 0 VBus is below A VBus valid threshold 1 VBus is above A VBus valid threshold
8 ID	USB ID 0 A device 1 B device
7 HABA	Hardware Assist B-Disconnect to A-connect 0 Disabled. 1 Enable automatic B-disconnect to A-connect sequence.
6 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
5 IDPU	ID Pull-Up Provides control over the ID pull-up resistor. 0 Disable pull-up. ID input not sampled. 1 Enable pull-up
4 DP	Data Pulsing 0 The pull-up on DP is not asserted 1 The pull-up on DP is asserted for data pulsing during SRP
3 OT	OTG Termination This bit must be set with the OTG module in device mode. 0 Disable pull-down on DM 1 Enable pull-down on DM
2 HAAR	Hardware Assist Auto-Reset 0 Disabled. 1 Enable automatic reset after connect on host port.
1 VC	VBUS Charge Setting this bit causes the VBUS line to charge. This is used for VBus pulsing during SRP.
0 VD	VBUS Discharge Setting this bit causes VBUS to discharge through a resistor.

48.3.32 USB Mode Register (USBHS_USBMODE)

This register is not defined in the EHCI specification. It controls the operating mode of the module.

Address: USBHS_USBMODE is 4003_4000h base + 1A8h offset = 4003_41A8h

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																	TXHSD			0							SDIS	SLOM	ES	CM	
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0

USBHS_USBMODE field descriptions

Field	Description
31–15 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
14–12 TXHSD	Tx to Tx HS Delay Controls the value of TX to TX HS Interpacket Delay by changing the internal delay count. The value of the global TX to TX interpacket delay depends on this internal counter and on the intrinsic PHY TX End Delay and TX Start Delay values. The TX to TX interpacket gap must be within the interval [88,192] bit times. 88 ≤ Controller internal delay – Tx End Delay + Tx Start Delay ≤ 192 (HS bit times) Below are the values of the internal controller counter in terms of PHY clock cycles. 000 10 001 11 010 12 011 13 100 14 101 15 110 16 111 17
11–5 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
4 SDIS	Stream DISable Host mode: Setting this bit 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 fills to capacity before the packet launches onto the USB.

Table continues on the next page...

USBHS_USBMODE field descriptions (continued)

Field	Description
	Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TXFILLTUNING to characterize the adjustments needed for the scheduler when using this feature. Also, in systems with high system bus utilization, setting this bit ensures no overruns or underruns during operation at the expense of link utilization. SDIS can be left clear and the rules under the description of the TXFILLTUNING register can limit underruns/overruns for those who desire optimal link performance. Device mode: Setting this bit disables double priming on 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. In high-speed mode, all packets received are responded to with a NYET handshake when stream disable is active. 0 Inactive 1 Active
3 SLOM	Setup Lock-Out Mode For the module in device mode, this bit controls behavior of the setup lock mechanism. See section “Control Endpoint Operation.”
2 ES	Endian Select Controls the byte ordering of the transfer buffers to match the host microprocessor bus architecture. The bit fields in the register interface and the DMA data structures (including the setup buffer within the device QH) are unaffected by the value of this bit, because they are based upon 32-bit words. 0 Little endian. First byte referenced in least significant byte of 32-bit word. 1 Big endian. First byte referenced in most significant byte of 32-bit word.
1–0 CM	Controller Mode This register can be written only once after reset. If necessary to switch modes, software must reset the controller by writing to the USBCMD[RST] bit before reprogramming this register. **NOTE:** The USBHS module must be initialized to the desired operating mode after reset. 00 Idle (default for the USBHS module) 01 Reserved 10 Device controller 11 Host controller

48.3.33 Endpoint Setup Status Register (USBHS_EPSETUPSR)

This register is not defined in the EHCI specification. This register contains the endpoint setup status and is used only in device mode.

Address: USBHS_EPSETUPSR is 4003_4000h base + 1ACh offset = 4003_41ACh

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																												EPSETUPSTAT			
W																													w1c			
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

USBHS_EPSETUPSR field descriptions

Field	Description
31–4 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
3–0 EPSETUPSTAT	Setup Endpoint Status For every setup transaction received, a corresponding bit in this field is set. This register is used only in device mode. Software must clear or acknowledge the setup transfer by writing a 1 to a respective bit after it has read the setup data from the 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 lockout mechanism engages.

48.3.34 Endpoint Initialization Register (USBHS_EPPRIME)

This register is not defined in the EHCI specification. This register is used to initialize endpoints and is used only in device mode.

Address: USBHS_EPPRIME is 4003_4000h base + 1B0h offset = 4003_41B0h

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												PETB				0												PERB			
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

USBHS_EPPRIME field descriptions

Field	Description
31–20 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
19–16 PETB	Prime Endpoint tTansmit Buffer For each endpoint, a corresponding bit requests that a buffer be prepared for a transmit operation to respond to a USB IN/INTERRUPT transaction. Software must write a 1 to the corresponding bit when posting a new transfer descriptor to an endpoint. 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 associated endpoint(s) is (are) successfully primed. **NOTE:** These bits are momentarily set by hardware during hardware re-priming operations when a dTD retires, and the dQH updates.
15–4 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
3–0 PERB	Prime Endpoint Receive Buffer For each endpoint, a corresponding bit requests that a buffer be prepared for a receive operation to respond to a USB OUT transaction. Software must write a 1 to the corresponding bit when posting a new transfer descriptor to an endpoint. 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 associated endpoint(s) is (are) successfully primed. **NOTE:** These bits are momentarily set by hardware during hardware re-priming operations when a dTD retires, and the dQH updates.

48.3.35 Endpoint Flush Register (USBHS_EPFLUSH)

This register is not defined in the EHCI specification. This register used only in device mode.

Address: USBHS_EPFLUSH is 4003_4000h base + 1B4h offset = 4003_41B4h

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												FETB				0												FERB			
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

USBHS_EPFLUSH field descriptions

Field	Description
31–20 Reserved	Reserved This read-only field is reserved and always has the value zero.

Table continues on the next page...

USBHS_EPFLUSH field descriptions (continued)

Field	Description
	Must be cleared
19–16 FETB	Flush Endpoint Transmit Buffer Writing a 1 to a bit in this field causes the associated endpoint to clear any primed buffers. If a packet is in progress for an associated endpoint, that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful.
15–4 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
3–0 FERB	Flush Endpoint Receive Buffer Writing a 1 to a bit in this field causes the associated endpoint to clear any primed buffers. If a packet is in progress for an associated endpoint, that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful. FERB[3] corresponds to endpoint 3.

48.3.36 Endpoint Status Register (USBHS_EPSR)

This register is not defined in the EHCI specification. This register is only used in device mode.

Address: USBHS_EPSR is 4003_4000h base + 1B8h offset = 4003_41B8h

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												ETBR				0												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

USBHS_EPSR field descriptions

Field	Description
31–20 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
19–16 ETBR	Endpoint Transmit Buffer Ready One bit for each endpoint indicates status of the respective endpoint buffer. The hardware sets this bit in response to receiving a command from a corresponding bit in the EPPRIME register. A constant delay exists between setting a bit in the EPPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the EPPRIME register. USB reset, USB DMA system, or EPFLUSH register clears the buffer ready. ETBR[3] (bit 19) corresponds to endpoint 3. **NOTE:** Hardware momentarily clears these bits during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated.

Table continues on the next page...

USBHS_EPSR field descriptions (continued)

Field	Description
15–4 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
3–0 ERBR	Endpoint Receive Buffer Ready One bit for each endpoint indicates status of the respective endpoint buffer. The hardware sets this bit in response to receiving a command from a corresponding bit in the EPPRIME register. A constant delay exists between setting a bit in the EPPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the EPPRIME register. USB reset, USB DMA system, or EPFLUSH register clears the buffer ready. ERBR[3] (bit 19) corresponds to endpoint 3. **NOTE:** Hardware momentarily clears these bits during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated.

48.3.37 Endpoint Complete Register (USBHS_EPCOMPLETE)

This register is not defined in the EHCI specification. This register is used only in device mode.

Address: USBHS_EPCOMPLETE is 4003_4000h base + 1BCh offset = 4003_41BCh

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												ETCE				0												ERCE			
W													w1c																w1c			
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

USBHS_EPCOMPLETE field descriptions

Field	Description
31–20 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
19–16 ETCE	Endpoint Transmit Complete Event Each bit indicates a transmit event (IN/INTERRUPT) occurs and software must read the corresponding endpoint queue to determine the endpoint status. If the corresponding IOC bit is set in the transfer descriptor, this bit is set simultaneously with the USBINT. Writing a 1 clears the corresponding bit in this register. ETCE[3] (bit 19) corresponds to endpoint 3.
15–4 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
3–0 ERCE	Endpoint Receive Complete Event Each bit indicates a received event (OUT/SETUP) occurs and software must read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the transfer

Table continues on the next page...

USBHS_EPCOMPLETE field descriptions (continued)

Field	Description
	descriptor, this bit is set simultaneously with the USBINT. Writing a 1 clears the corresponding bit in this register. ERCE[3] corresponds to endpoint 3.

48.3.38 Endpoint Control Register 0 (USBHS_EPCR0)

This register is not defined in the EHCI specification. Every device implements endpoint 0 as a control endpoint.

Address: USBHS_EPCR0 is 4003_4000h base + 1C0h offset = 4003_41C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TXE	0			TXT		0	TXS
W																
Reset	0	0	0	0	0	0	0	0	1	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								RXE	0			RXT		0	RXS
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

USBHS_EPCR0 field descriptions

Field	Description
31–24 Reserved	This register is not defined in the EHCI specification. Every device implements endpoint 0 as a control endpoint. This read-only field is reserved and always has the value zero. Must be cleared.
23 TXE	TX Endpoint Enable Endpoint zero is always enabled. 1 Enable
22–20 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
19–18 TXT	TX Endpoint Type Endpoint zero is always a control endpoint. 00 Control
17 Reserved	Reserved

Table continues on the next page...

USBHS_EPCR0 field descriptions (continued)

Field	Description
	This read-only field is reserved and always has the value zero. Must be cleared
16 TXS	TX Endpoint Stall Software can write a 1 to this bit to force the endpoint to return a STALL handshake to the host. It continues returning STALL until software clears the bit or it automatically clears upon receipt of a new SETUP request. 0 Endpoint OK 1 Endpoint stalled
15–8 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
7 RXE	RX endpoint Enable Endpoint zero is always enabled. 1 Enabled
6–4 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
3–2 RXT	RX endpoint Type Endpoint zero is always a control endpoint. 00 Control
1 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared
0 RXS	RX endpoint Stall Software can write a 1 to this bit to force the endpoint to return a STALL handshake to the host. It continues returning STALL until software clears the bit or it automatically clears upon receipt of a new SETUP request. 0 Endpoint OK 1 Endpoint stalled

48.3.39 Endpoint Control Register n (USBHS_EPCR*n*)

These registers are not defined in the EHCI specification. There is an EPCRn register for each endpoint in a device.

Addresses: USBHS_EPCR1 is 4003_4000h base + 1C4h offset = 4003_41C4h
USBHS_EPCR2 is 4003_4000h base + 1C8h offset = 4003_41C8h
USBHS_EPCR3 is 4003_4000h base + 1CCh offset = 4003_41CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TXE		TXI	0	TXT		TXD	TXS
W										TXR						
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								RXE		RXI	0	RXT		RXD	RXS
W										RXR						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBHS_EPCR*n* field descriptions

Field	Description
31–24 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
23 TXE	TX endpoint Enable 0 Disabled 1 Enabled
22 TXR	TX data toggle Reset When a configuration event is received for this Endpoint, software must write a 1 to this bit to synchronize the data PID's between the host and device. This bit is self-clearing.
21 TXI	TX data toggle Inhibit This bit is used only for test and should always be written as 0. Writing a 1 to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet. 0 PID sequencing enabled 1 PID sequencing disabled
20 Reserved	Reserved

Table continues on the next page...

USBHS_EPCR*n* field descriptions (continued)

Field	Description
	This read-only field is reserved and always has the value zero. Must be cleared.
19–18 TXT	TX endpoint Type **NOTE:** When only one endpoint (RX or TX, but not both) of an endpoint pair is used, the unused endpoint should be configured as a bulk type endpoint. 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX endpoint Data source This bit should always be written as 0, which selects the dual port memory/DMA engine as the source.
16 TXS	TX endpoint Stall This bit sets automatically upon receipt of a SETUP request if this endpoint is not configured as a control endpoint. It clears automatically upon receipt of a SETUP request if this endpoint is configured as a control endpoint. Software can write a 1 to this bit to force the endpoint to return a STALL handshake to the host. It continues returning STALL until software clears this bit clears or automatically clears as above. 0 Endpoint OK 1 Endpoint stalled
15–8 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
7 RXE	RX endpoint Enable 0 Disabled 1 Enabled
6 RXR	RX data toggle Reset When a configuration event is received for this endpoint, software must write a 1 to this bit to synchronize the data PIDs between the host and device. This bit is self-clearing.
5 RXI	RX data toggle Inhibit This bit is only for testing and should always be written as 0. Writing a 1 to this bit causes this endpoint to ignore the data toggle sequence and always accept data packets regardless of their data PID. 0 PID sequencing enabled 1 PID sequencing disabled
4 Reserved	Reserved This read-only field is reserved and always has the value zero. Must be cleared.
3–2 RXT	RX endpoint Type

Table continues on the next page...

USBHS_EPCR*n* field descriptions (continued)

Field	Description
	NOTE: When only one endpoint (RX or TX, but not both) of an endpoint pair is used, the unused endpoint should be configured as a bulk type endpoint. 00 Control 01 Isochronous 10 Bulk 11 Interrupt
1 RXD	RX endpoint Data sink This bit should always be written as 0, which selects the dual port memory/DMA engine as the sink.
0 RXS	RX endpoint Stall This bit sets automatically upon receipt of a SETUP request if this endpoint is not configured as a control endpoint. It clears automatically upon receipt of a SETUP request if this endpoint is configured as a control endpoint. Software can write a 1 to this bit to force the endpoint to return a STALL handshake to the host. It continues returning STALL until software clears this bit or automatically clears as above. 0 Endpoint OK 1 Endpoint stalled

48.3.40 USB General Control Register (USBHS_USBGENCTRL)

This register is not defined in the EHCI specification.

Address: USBHS_USBGENCTRL is 4003_4000h base + 200h offset = 4003_4200h

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										WU_INT_CLR	Reserved			WU_ULPI_EN	WU_IE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

USBHS_USBGENCTRL field descriptions

Field	Description
31–6 Reserved	Reserved This read-only field is reserved and always has the value zero.
5 WU_INT_CLR	Wakeup Interrupt Clear 0 Default, no action. 1 Clear the wake-up interrupt.
4–2 Reserved	Reserved This field is reserved.
1 WU_ULPI_EN	Wakeup on ULPI Interrupt Event This bit is used to enable the wake up from the ULPI I/F. 0 Disabled 1 Enabled
0 WU_IE	Wakeup Interrupt Enable This bit is used to enable the low power wakeup interrupt. 0 Disabled 1 Enabled

48.4 Functional Description

This module can be broken down into functional sub-blocks as described below.

48.4.1 System Interface

The system interface block contains all the control and status registers to allow a core to interface to the module. These registers allow the processor to control the configuration and ascertain the capabilities of the module and, they control the module's operation.

48.4.2 DMA Engine

The USBHS module contains a local DMA engine. It is responsible for moving all of the data transferred over the USB between the module and system memory.

The DMA controllers must access control information and packet data from system memory. Control information is contained in link list based queue structures. The DMA controllers have state machines able to parse data structures defined in the EHCI specification. In host mode, the data structures are EHCI compliant and represent queues of transfers performed by the host controller, including the split-transaction requests that allow an EHCI controller to direct packets to FS and LS speed devices. In device mode, data structures are similar to those in the EHCI specification and used to allow device responses to be queued for each of the active pipes in the device.

48.4.3 FIFO RAM Controller

The FIFO RAM controller is used for context information and to control FIFOs between the protocol engine and the DMA controller. These FIFOs decouple the system processor/memory bus requests from the extremely tight timing required by USB.

The use of the FIFO buffers differs between host and device mode operation. In host mode, a single data channel is maintained in each direction through the buffer memory. In device mode, multiple FIFO channels are maintained for each of the active endpoints in the system.

In host mode, the USB OTG modules use 16-byte transmit buffers and 16-byte receive buffers. For the USB OTG module, device operation uses a single 16-byte receive buffer and a 16-byte transmit buffer for each endpoint.

48.5 Initialization/Application Information

This section discusses host operation, device data structures, device operation, and servicing interrupts.

48.5.1 Host Operation

Enhanced Host Controller Interface (EHCI) Specification defines the general operational model for a USB module in host mode. The EHCI specification describes the register-level interface for a host controller for USB Revision 2.0. It includes a description of the hardware/software interface between system software and host controller hardware. The next section has information about the initialization of the USB modules; however, full details of the EHCI specification are beyond the scope of this document.

48.5.2 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 interface consists of device queue heads and transfer descriptors.

Note

> Software must ensure that data structures do not span a 4K-page boundary.

The USB OTG uses an array of device endpoint queue heads to organize device transfers. As shown in the next figure, there are two endpoint queue heads in the array for each device endpoint—one for IN and one for OUT. The EPLISTADDR provides a pointer to the first entry in the array.

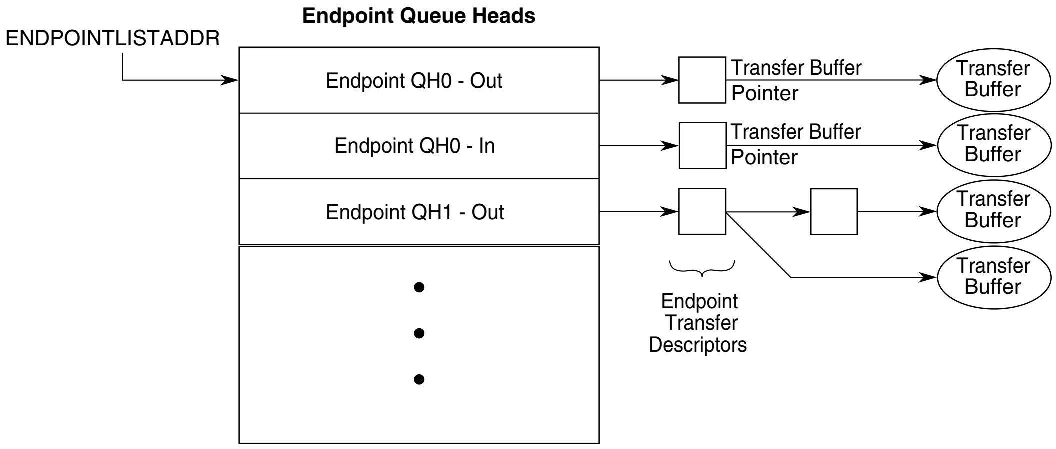


Figure 48-48. End Point Queue Head Organization

48.5.2.1 Endpoint Queue Head

All transfers are managed in the device endpoint queue head (dQH). The dQH is a 48-byte data structure, but must align on 64-byte boundaries. During priming of an endpoint, the dTD (device transfer descriptor) copies into the overlay area of the dQH, which starts at the nextTD pointer and continues through the end of the buffer pointers. After a transfer is complete, the dTD status updates in the dTD pointed to by the currentTD pointer. While a packet is in progress, the overlay area of the dQH acts as a staging area for the dTD so the device controller can access needed information with minimal latency.

The next figure shows the endpoint queue head structure.

Table 48-54. Endpoint Queue Head 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	offset
Mult		ZLT	0	0	Maximum Packet Length											IOS	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x00
Current dTD Pointer																											0	0	0	0	0	0x04
Next dTD Pointer																											0	0	0	0	T	0x08[1]
0	0	Total Bytes														IOC	0	0	0	Mult O		0	0	Status								0x0C[1]
Buffer Pointer (Page 0)																				Current Offset												0x10[1]
Buffer Pointer (Page 1)																				Reserved												0x14[1]
Buffer Pointer (Page 2)																				Reserved												0x18[1]
Buffer Pointer (Page 3)																				Reserved												0x1C[1]
Buffer Pointer (Page 4)																				Reserved												0x20[1]
Reserved																																0x24
Setup Buffer Bytes 3–0																																0x28
Setup Buffer Bytes 7–4																																0x2C

▨ Device controller read/write; all others read-only.

[1] Offsets 0x08 through 0x20 contain the transfer overlay.

48.5.2.1.1 Endpoint Capabilities/Characteristics (Offset = 0x0)

This entry specifies static information about the endpoint. In other words, this information does not change over the lifetime of the endpoint. DCD software must not attempt to modify this information while the corresponding endpoint is enabled.

Table 48-55. Endpoint Capabilities/Characteristics

Field	Description
31–30 Mult	Mult. This field indicates the number of packets executed per transaction description as given by: 00 Execute N Transactions as demonstrated by the USB variable length packet protocol where N computes 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 equal to 00. ISO endpoints must set Mult equal to 01, 10, or 11 as needed.

Table continues on the next page...

Table 48-55. Endpoint Capabilities/Characteristics (continued)

Field	Description
29 ZLT	Zero length termination select. This bit is ignored in isochronous transfers. Clearing this bit enables the hardware to automatically append a zero length packet when the following conditions are true: • The packet transmitted equals maximum packet length • The dTD has exhausted the field Total Bytes After this the dTD retires. When the device is receiving, if the last packet length received equals the maximum packet length and the total bytes is zero, it waits for a zero length packet from the host to retire the current dTD. Setting this bit disables the zero length packet. When the device is transmitting, the hardware does not append any zero length packet. When receiving, it does not require a zero length packet to retire a dTD whose last packet was equal to the maximum packet length packet. The dTD is retired as soon as Total Bytes field goes to zero, or a short packet is received. 0 Enable zero length packet (default). 1 Disable the zero length packet. **Note:** Each transfer is defined by one dTD, so the zero length termination is for each dTD. In some software application cases, the logic transfer does not fit into only one dTD, so it does not make sense to add a zero length termination packet each time a dTD is consumed. On those cases we recommend to disable the ZLT feature, and use software to generate the zero length termination.
28–27	Reserved. Reserved for future use and must be cleared.
26–16 Maximum Packet Length	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 IOS	Interrupt on setup (IOS). This bit used on control type endpoints indicates if USBSTS[UI] is set in response to a setup being received.
14–0	Reserved. Reserved for future use and must be cleared.

48.5.2.1.2 Current dTD Pointer (Offset = 0x4)

The device controller uses the current dTD pointer to locate transfer in progress. This word is for USB OTG (hardware) use only and should not be modified by DCD software.

Table 48-56. Current dTD Pointer

Field	Description
31–5 Current dtd	Current dtd. This field is a pointer to the dTD represented in the transfer overlay area. This field is modified by the device controller to next dTD pointer during endpoint priming or queue advance.
4–0	Reserved. Reserved for future use and must be cleared.

48.5.2.1.3 Transfer Overlay (Offset = 0x8–0x20)

The seven entries 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, it does not read the associated endpoint.

After an endpoint is readied, the dTD is copied into this queue head overlay area by the device controller. Until a transfer expires, software must not write the queue head overlay area or the associated transfer descriptor. When the transfer is complete, the device controller writes the results back to the original transfer descriptor and advance the queue.

48.5.2.1.4 Setup Buffer (Offset = 0x28–0x2C)

The set-up buffer is dedicated storage for the 8-byte data that follows a set-up PID. Refer to Control Endpoint Operation for information on the procedure for reading the setup buffer

Note

> Each endpoint has a TX and an RX dQH associated with it, and only the RX queue head receives setup data packets.

Table 48-57. Multiple Mode Control

Offset	Field	Description
0x28	31–0 Setup Buffer 0	Setup Buffer 0. This buffer contains bytes 3 to 0 of an incoming setup buffer packet and is written by the device controller software reads.
0x2C	31–0 Setup Buffer 1	Setup Buffer 1. This buffer contains bytes 7 to 4 of an incoming setup buffer packet and is written by the device controller software reads.

48.5.2.2 Endpoint Transfer Descriptor (dTD)

The dTD describes to the device controller the location and quantity of data sent/received for a given transfer. The DCD software should not attempt to modify any field in an active dTD except the next dTD pointer.

Table 48-58. 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	offset

Table continues on the next page...

Table 48-58. Endpoint Transfer Descriptor (dTD) (continued)

<table>
<tr><td colspan="13">Next dTD Pointer</td><td>0</td><td>0</td><td>0</td><td>0</td><td>T</td><td>0x00</td></tr>
<tr><td>0</td><td>Total Bytes</td><td>ioc</td><td>0</td><td>0</td><td>0</td><td colspan="2">MultO</td><td>0</td><td>0</td><td colspan="8">Status</td><td>0x04</td></tr>
<tr><td colspan="6">Buffer Pointer (Page 0)</td><td colspan="12">Current Offset</td><td>0x08</td></tr>
<tr><td colspan="6">Buffer Pointer (Page 1)</td><td>0</td><td colspan="11">Frame Number</td><td>0x0C</td></tr>
<tr><td colspan="6">Buffer Pointer (Page 2)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0x10</td></tr>
<tr><td colspan="6">Buffer Pointer (Page 3)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0x14</td></tr>
<tr><td colspan="6">Buffer Pointer (Page 4)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0x18</td></tr>
</table>

▨ Device controller read/write; all others read-only.

48.5.2.2.1 Next dTD Pointer (Offset = 0x0)

The next dTD pointer is used to point the device controller to the next dTD in the linked list.

Table 48-59. Next dTD Pointer

Field	Description
31–5 Next dTD pointer	Next dTD 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. Reserved for future use and must be cleared.
0 T	Terminate. This bit indicates to the device controller no more valid entries exist in the queue. 0=Pointer is valid (points to a valid transfer element descriptor). 1=pointer is invalid.

48.5.2.2.2 dTD Token (Offset = 0x4)

The dTD token is used to specify attributes for the transfer including the number of bytes to read or write and the status of the transaction.

Table 48-60. dTD Token

Field	Description
31	Reserved. Reserved for future use and must be cleared.
30–16 Total Bytes	Total bytes. This field specifies the total number of bytes moved with this transfer descriptor. This field decrements 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(0x5000). This is the maximum number of bytes 5 page pointers can access. Although possible to create a transfer up to 20K, this assumes the first offset into the first page is 0. When the offset cannot be predetermined, crossing past the fifth page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16K (0x4000). **Note:** Larger transfer sizes can be supported, but require disabling ZLT and using multiple dTDs. If the value of the field is 0 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. For IN transfers it is not a requirement for total bytes to transfer be an even multiple of the maximum packet length. If software builds such a transfer descriptor for an IN transfer, the last transaction is always less than maximum packet length. For OUT transfers the total bytes must be evenly divisible by the maximum packet length.
15 IOC	Interrupt on complete. Indicates if USBSTS[UI] is set in response to device controller finished with this dTD.
14–12	Reserved. Reserved for future use and must be cleared.
11–10 MultO	Multiplier Override. This field can possibly transmit-ISOs (ISO-IN) to override the multiplier in the QH. This field must be 0 for all packet types not transmit-ISO. For example, if QH.MULT equals 3; Maximum packet size equals 8; Total Bytes equals 15; MultiO equals 0 [default], then three packets are sent: {Data2(8); Data1(7); Data0(0)}. If QH.MULT equals 3; Maximum packet size equals 8; Total Bytes equals 15; MultO equals 2, then two packets are sent: {Data1(8); Data0(7)} For maximal efficiency, software must compute MultO equals greatest integer of (Total Bytes / Max. Packet Size) except for the case when Total Bytes equals 0; then MultO must be 1. **Note:** Non-ISO and Non-TX endpoints must set MultO equals 00.
9–8	Reserved. Reserved for future use and must be cleared.

Table continues on the next page...

Table 48-60. dTD Token (continued)

Field	Description	
7–0 Status	Status. Device controller communicates individual command execution states back to the DCD software. This field contains the status of the last transaction performed on this dTD. The bit encodings are:	
	Bit	**Status Field Description**
	7	Active. Set by software to enable the execution of transactions by the device controller.
	6	Halted. Set by the device controller during status updates to indicate a serious error has occurred at the device/endpoint addressed by this dTD. Any time a transaction results in the halted bit being set, the active bit is also cleared.
	5	Data Buffer Error. Set by the device controller during status update to indicate the device controller is unable to maintain the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run).
	4	Reserved.
	3	Transaction Error. Set by the device controller during status update in case the device did not receive a valid response from the host (time-out, CRC, bad PID).
	2–0	Reserved.

48.5.2.2.3 dTD Buffer Page Pointer List (Offset = 0x8–0x18)

The buffer page pointer list of a device element transfer descriptor is an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

Table 48-61. Buffer Page Pointer List

Field	Description
31–12 Buffer Pointer	Buffer Pointer. Selects the page offset in memory for the packet buffer. Non virtual memory systems typically set the buffer pointers to a series of incrementing integers.
0;11–0 Current Offset	Current Offset. Offset into the 4kB buffer where the packet begins.
1;10–0 Frame Number	Frame Number. Written by the device controller to indicate the frame number a packet finishes in. Typically correlates relative completion times of packets on an ISO endpoint.

48.5.3 Device Operation

The device controller performs data transfers using a set of linked list transfer descriptors pointed to by a queue head. The next 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.

48.5.3.1 Port State and Control

From a chip or system reset, the USB OTG module enters the powered state. A transition from the powered state to the attach state occurs when the USBCMD[RS] bit is set. After receiving a reset on the bus, the port enters the defaultFS or defaultHS state in accordance with the protocol reset described in Appendix C.2 of the *Universal Serial Bus Specification, Revision 2.0*. The next figure depicts the state of a USB 2.0 device.

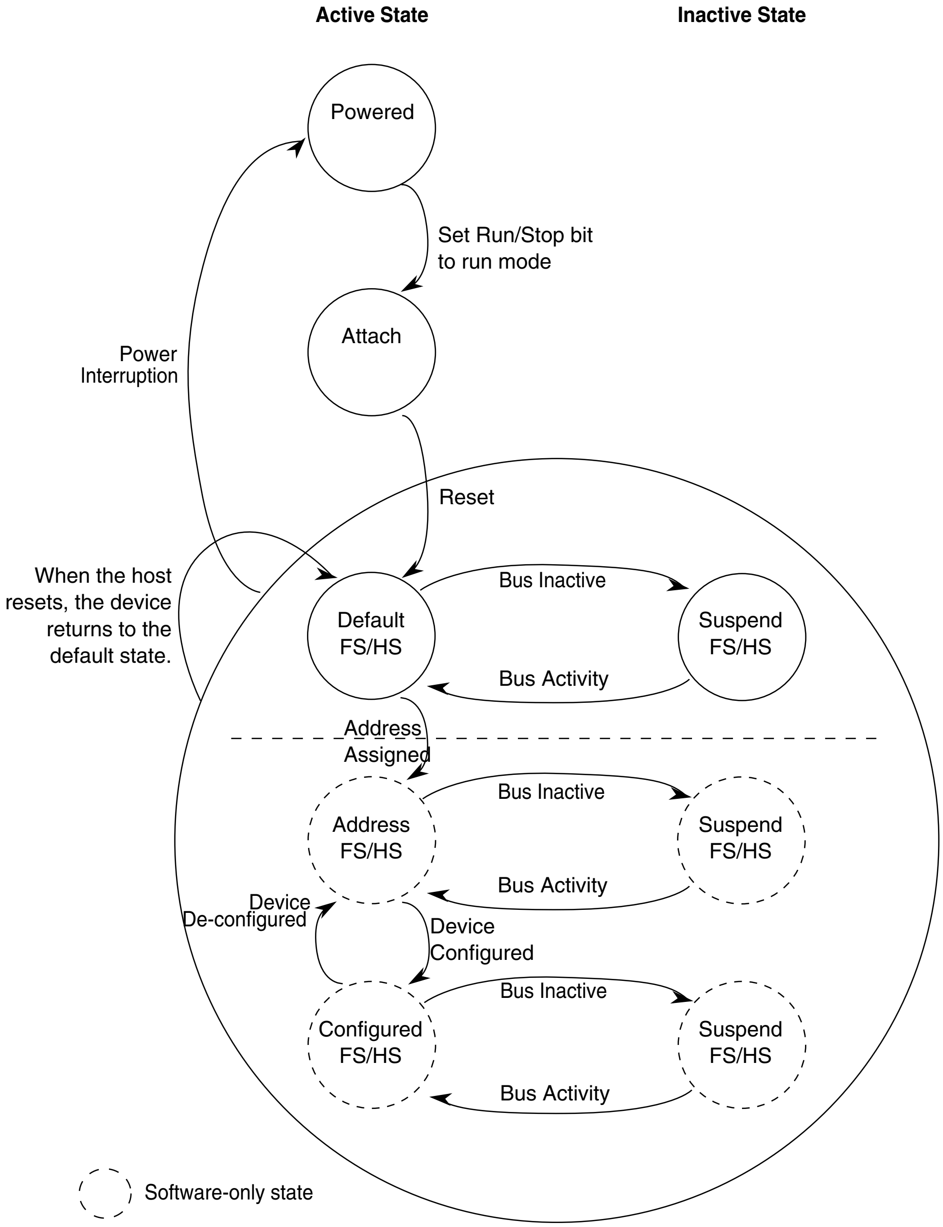


Figure 48-49. USB 2.0 Device States

States powered, attach, defaultFS/HS, suspendFS/HS are implemented in the USB OTG, and they are communicated to the DCD using these status bits:

Table 48-62. Device Controller State Information Bits

Bit	Register
DC Suspend (SLI)	USBSTS
USB Reset Received (URI)	USBSTS
Port Change Detect (PCI)	USBSTS
High-Speed Port (PSPD)	PORTSCn

DCD software must maintain a state variable to differentiate between the defaultFS/HS state and the address/configured states. Change of state from default to the address and 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 DCD must program the device address register (DEVICEADDR).

Entry into the configured state indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the EPCR*n* registers and initializing the associated queue heads.

48.5.3.1.1 Bus Reset

The host uses a bus reset to initialize downstream devices. When a bus reset is detected, USB OTG controller renegotiates its attachment speed, resets the device address to 0, and notifies 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 the device controller cancels any primed transactions. The concept of priming is clarified below, but when a reset is received, the DCD must perform:

1. Clear all setup token semaphores by reading the EPSETUPSR register and writing the same value back to the EPSETUPSR register.
2. Clear all the endpoint complete status bits by reading the EPCOMPLETE register and writing the same value back to the EPCOMPLETE register.
3. Cancel all primed status by waiting until all bits in the EPPRIME are 0 and then writing 0x000F_000F to EPFLUSH.
4. Read the reset bit in the PORTSC*n* register and make sure it remains active. A USB reset occurs for a minimum of 3 ms and the DCD must reach this point in the reset clean-up before end of the reset occurs, otherwise a hardware reset of the device controller is recommended (rare).

a. Setting USBCMD[RST] bit can perform a hardware reset.

Note

> A hardware reset causes the device to detach from the bus by clearing the USBCMD[RS] bit. Therefore, the DCD must completely re-initialize the USB OTG after a hardware reset.

5. Free all allocated dTDs because the device controller no longer executes them. If this is the first time the DCD processes a USB reset event, it is likely no dTDs have been allocated.
6. 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.
7. After a port change detect, the device has reached the default state and the DCD can read the PORTSC*n* register to determine if the device operates in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration according to the chapter 9 Device Framework of the USB specification.

48.5.3.1.2 Suspend/Resume

To conserve power, USB OTG module automatically enters the suspended state when no bus traffic is observed for a specified period. When suspended, the module maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend any time they are powered, regardless if they are 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.

The USB OTG module exits suspend mode when there is bus activity. It may also request the host to exit suspend mode or selective suspend by using electrical signaling to indicate remote wake-up. The ability of a device to signal remote wake-up is optional. The USB OTG is capable of remote wake-up signaling. When the USB OTG is reset, remote wake-up signaling must be disabled.

48.5.3.1.2.1 Suspend Operational Model

The USB OTG 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, an interrupt notifies the DCD (assuming device controller suspend interrupt is enabled, USBINTR[SLE] is set). When the PORTSC*n*[SUSP] is set, the device controller is suspended.

DCD response when the device controller is suspended is application specific and may involve switching to low power operation. Find information on the bus power limits in suspend state in USB 2.0 specification.

48.5.3.1.2.2 Resume

If the USB OTG is suspended, its operation resumes when any non-idle signaling is received on its upstream facing port. In addition, the USB OTG can signal the system to resume operation by forcing resume signaling to the upstream port. Setting the PORTSC*n*[FPR] bit while the device is in suspend state sends resume signaling upstream. 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 use of resume signaling, the host must enable it by using the set feature command defined in chapter 9 Device Framework of the USB 2.0 specification.

48.5.3.2 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. Combination of the endpoint number and the endpoint direction specifies endpoint address.

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 are supported by USB include bulk, interrupt, and isochronous. Each endpoint type has specific behavior related to packet response and error managing. Find more detail on endpoint operation in the USB 2.0 specification.

The USB OTG supports up to four endpoint specified numbers. The DCD can enable, disable, and configure each endpoint.

Each endpoint direction is essentially independent and can have 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, for example, is always a control endpoint and uses both directions.

Each endpoint direction requires a queue head allocated in memory. If the maximum is four endpoint numbers (one for each endpoint direction used by the device controller), eight queue heads are required. The operation of an endpoint and use of queue heads are described later in this document.

48.5.3.2.1 Endpoint Initialization

After hardware reset, all endpoints except endpoint 0 are uninitialized and disabled. The DCD must configure and enable each endpoint by writing to the appropriate EPCR*n* register. Each EPCR*n* is split into an upper and lower half. The lower half of EPCR*n* configures the receive or OUT endpoint, and the upper half configures the corresponding transmit or IN endpoint. Control endpoints must be configured the same in the upper and lower half of the EPCR*n* register; otherwise, behavior is undefined. The next table shows how to construct a configuration word for endpoint initialization.

Table 48-63. Device Controller Endpoint Initialization

Field	Value
Data Toggle Reset (TXR, RXR)	1 Synchronize the data PIDs
Data Toggle Inhibit (TXI, RXI)	0 PID sequencing disabled
Endpoint Type (TXT, RXT)	00 Control 01 Isochronous 10 Bulk 11 Interrupt
Endpoint Stall (TXS, RXS)	0 Not stalled

48.5.3.2.2 Stalling

There USB OTG has two occasions it may need to return to the host a STALL:

- The first is the functional stall, a condition set by the DCD as described in the USB 2.0 Device Framework chapter. A functional stall is used only on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the EPCR*n* register associated with the given endpoint and the given direction. In a functional stall condition, the device controller continues 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 and automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, DCD must enable the stall bits as a pair (TXS and RXS bits). A single write to the EPCR*n* register can ensure both stall bits are set at the same instant.

Note

Any write to the EPCR*n* register during operational mode must preserve the endpoint type field (perform a read-modify-write).

Table 48-64. 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
IN/OUT/PING packet received by a control endpoint.	0	None	ACK/NAK/NYET

48.5.3.2.3 Data Toggle

Data toggle maintains data coherency between host and device for any given data pipe. For more information on data toggle, refer to the USB 2.0 specification.

48.5.3.2.3.1 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 setting the data toggle reset bit in the EPCR*n* register. This should only happen when configuring/initializing an endpoint or returning from a STALL condition.

48.5.3.2.3.2 Data Toggle Inhibit

This feature is for test purposes only and must never be used during normal device controller operation.

Setting the data toggle inhibit bit causes the USB OTG module to ignore the data toggle pattern normally sent and accepts all incoming data packets regardless of the data toggle state.

In normal operation, the USB OTG checks the DATA0/DATA1 bit against the data toggle to determine if the packet is valid. If the 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 USB OTG from re-sending the same packet, the device controller responds to the error packet by acknowledging it with an ACK or NYET response.

48.5.3.3 Packet Transfers

The host initiates all transactions on the USB bus and in turn, the device must respond to any request from the host within the turnaround time stated in the USB 2.0 specification.

A USB host sends requests to the USB OTG 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, expect the host to send IN requests to that endpoint. This USB OTG module prepares 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 appears throughout the documentation to describe the USB OTG operation so the DCD is built properly. Further, the term flushing describes the action of clearing a packet queued for execution.

48.5.3.3.1 Priming Transmit Endpoints

Priming a transmit endpoint causes 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 is 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 manage 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 splits into virtual channels so 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 EPSR 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.

48.5.3.3.2 Priming Receive Endpoints

Priming receives endpoints identical to priming of transmit endpoints from the point of view of the DCD. The major difference in the operational model at the device controller is no data movement of the leading packet data because the data is to be received from the host.

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.

48.5.3.3.3 Interrupt/Bulk Endpoint Operation

The behaviors of the device controller for interrupt and bulk endpoints are identical. All valid IN and OUT transactions to bulk pipes handshake with a NAK unless the endpoint is primed. After the endpoint is primed, data delivery commences.

A dTD is retired by the device controller when the packets described in the transfer descriptor are completed. Each dTD describes N packets to transfer according to the USB variable length transfer protocol. The formula below and the next table describe how the device controller computes the number and length of the packets sent/received by the USB vary according to the total number of bytes and maximum packet length. See Endpoint Capabilities/Characteristics (Offset = 0x0) for details on the ZLT bit.

With zero-length termination (ZLT) cleared:

N = INT(number of bytes/max. packet length) + 1

With zero-length termination (ZLT) set:

N = MAXINT(number of bytes/max. packet length)

Table 48-65. Variable Length Transfer Protocol Example (ZLT=0)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	—
512	256	3	256	256	0
512	512	2	512	0	—

Table 48-66. 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 in the dTD successfully transmit. Total bytes in dTD equal 0 when this occurs.

RX-dTD is complete when:

- All packets described in the dTD are successfully received. Total bytes in dTD equal 0 when this occurs.
- A short packet (number of bytes < maximum packet length) was received.

  This is a successful transfer completion; DCD must check the total bytes field in the dTD to determine the number of bytes 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 discards the remaining packet and set the buffer error bit in the dTD. In addition, the endpoint flushes and the USBERR interrupt becomes active.

Note

Disabling zero-length packet termination allows transfers larger than the total bytes field spanning across two or more dTDs.

Upon successful completion of the packet(s) described by the dTD, the active bit in the dTD is cleared and the next pointer is followed when the terminate bit is clear. When the terminate bit is set, USB OTG flushes the endpoint/direction and ceases operations for that endpoint/direction.

Upon unsuccessful completion of a packet (see long packet above), the dQH is left pointing to the dTD in error. To recover from this error condition, DCD must properly re-initialize 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, are retried automatically by the device controller. There is no required interaction with the DCD for managing such errors.

Table 48-67. Interrupt/Bulk Endpoint Bus Response Matrix

Token Type	Stall	Not Primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	N/A	N/A
In	STALL	NAK	Transmit	BS Error[1]	N/A
Out	STALL	NAK	Receive + NYET/ACK[2]	N/A	NAK
Ping	STALL	NAK	ACK	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

1. Force bit stuff error
2. NYET/ACK — NYET unless the transfer descriptor has packets remaining according to the USB variable length protocol then ACK.

48.5.3.3.4 Control Endpoint Operation

48.5.3.3.4.1 Setup Phase

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase.

Setup packet managing:

- Disable setup lockout by setting the setup lockout mode bit (USBMODE[SLOM]), once at initialization. Setup lockout is not necessary when using the tripwire as described below.

Note

Leaving the setup lockout mode cleared results in a potential compliance issue.

- After receiving an interrupt and inspecting EPSETUPSR to determine a setup packet was received on a particular pipe:

1. Write 1 to clear corresponding bit in EPSETUPSR.
2. Set the setup tripwire bit (USBCMD[SUTW]).
3. Duplicate contents of dQH.SetupBuffer into local software byte array.
4. Read the USBCMD[SUTW] bit. If set, continue; if cleared, goto 2)
5. Clear the USBCMD[SUTW] bit.
6. Poll until the EPSETUPSR bit clears.
7. Process setup packet using the local software byte array copy and execute status/ handshake phases.

Note

> After receiving a new setup packet, status and/or handshake phases may remain pending from a previous control sequence. These should be flushed and de-allocated before linking a new status and/or handshake dTD for the most recent setup packet.

48.5.3.3.4.2 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 is not received by reading the EPSETUPSR register immediately verifying that the prime had completed. A prime completes when the associated bit in the EPPRIME register is cleared and the associated bit in the EPSR register is set. If the EPPRIME bit goes to 0 and the EPSR bit is not set, the prime fails. This can only happen because of improper setup of the dQH, dTD, or a setup arriving during the prime operation. If a new setup packet is indicated after the EPPRIME bit is cleared, then the transfer descriptor can be freed and the DCD must re-interpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller automatically clears the prime status (EPSR) to enforce data coherency with the setup packet.

Note

> Error managing of data phase packets is the same as bulk packets described previously.

48.5.3.3.4.3 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 EPSETUPSR as described above in the data phase.

Note

Error managing of status phase packets is the same as bulk packets described previously.

48.5.3.3.4.4 Control Endpoint Bus Response Matrix

The next table shows the device controller response to packets on a control endpoint according to the device controller state.

Table 48-68. Control Endpoint Bus Response Matrix

Token Type	Endpoint State					Setup Lockout
	Stall	Not Primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	N/A	SYSERR[1]	
In	STALL	NAK	Transmit	BS Error[2]	N/A	N/A
Out	STALL	NAK	Receive + NYET/ ACK[3]	N/A	NAK	N/A
Ping	STALL	NAK	ACK	N/A	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore

[1] SYSERR — System error must never occur when the latency FIFOs are correctly sized and the DCD is responsive.

[2] Force bit stuff error.

[3] NYET/ACK — NYET unless the transfer descriptor has packets remaining according to the USB variable length protocol then ACK.

48.5.3.3.5 Isochronous Endpoint Operation

Isochronous endpoints 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 USB OTG is accomplished by:

- Exactly MULT packets per (micro)frame are transmitted/received.

Note

MULT is a two-bit field in the device queue head. Isochronous endpoints do not use the variable length packet protocol.

- NAK responses are not used. Instead, zero length packets are sent in response to an IN request to 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 ISO-dTD remains active after that frame, ISO-dTD holds ready until executed or canceled by the DCD.

The USB OTG in host mode 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 managing 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 becomes primed only after a SOF is received. After the DCD writes the prime bit, the prime bit clears 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 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. After an ISO transaction is started in a (micro)frame, it retires 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, and the device controller retires the current ISO-dTD and move to the next ISO-dTD.

Fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction stays primed indefinitely. This means it is up to software must 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 managing. 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. If the multiplier override field is zero, the MULT counter initializes to the multiplier in the QH.

- RX packet retired:
    - MULT counter reaches zero.
    - Non-MDATA data PID is received
    - 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, DCD must ensure the dTD linked-list is out ahead of the device controller by at least two (micro)frames.

48.5.3.3.5.1 Isochronous Pipe Synchronization

When it is necessary to synchronize an isochronous data pipe to the host, the (micro)frame number (FRINDEX register) can act as a marker. To cause a packet transfer to occur at a specific (micro)frame number (N), the DCD must interrupt on SOF during frame N-1. When the FRINDEX equals N-1, the DCD must write the prime bit. The USB OTG primes the isochronous endpoint in (micro)frame N-1 so the device controller executes delivery during (micro)frame N.

CAUTION

Priming an endpoint towards the end of (micro)frame N-1 does not guarantee delivery in (micro)frame N. The delivery may actually occur in (micro)frame N+1 if the device controller does not have enough time to complete the prime before the SOF for packet N is received.

48.5.3.3.5.2 Isochronous Endpoint Bus Response Matrix

Table 48-69. Isochronous Endpoint Bus Response Matrix

Token Type	Stall	Not Primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	N/A	N/A
In	NULL[1] Packet	NULL Packet	Transmit	BS Error[2]	N/A
Out	Ignore	Ignore	Receive	N/A	Drop Packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

[1] Zero length packet

[2] Force bit stuff error

48.5.3.4 Managing Queue Heads

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 EPLISTADDR contains a group of all dQH's in a sequential list (see the next figure). The even elements in the list of dQH's receive endpoints (OUT/SETUP) and the odd elements transmit endpoints (IN/INTERRUPT). Device transfer descriptors are linked head to tail starting at the queue head and ending at a terminate bit. After the dTD retires, it is no longer part of the linked list from the queue head. Therefore, software is required to track all transfer descriptors because pointers no longer exist within the queue head after the dTD is retired (see Software Link Pointers).

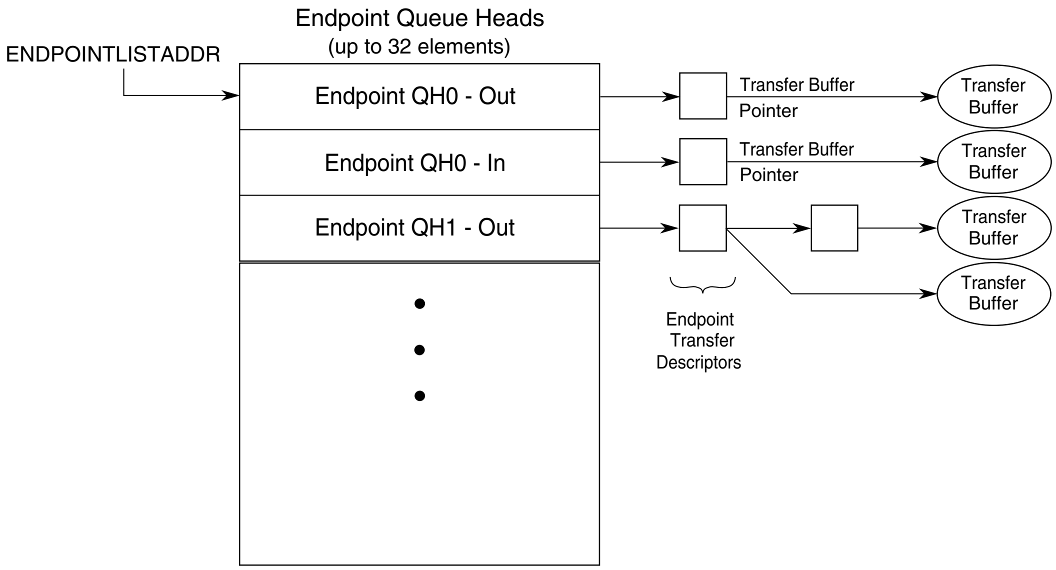


Figure 48-50. Endpoint Queue Head Diagram

In addition to current and next pointers and the dTD overlay examined in Packet Transfers the dQH also contains the following parameters for the associated endpoint: multipler, maximum packet length, and interrupt on setup. The next section includes demonstration of complete initialization of the dQH including these fields.

48.5.3.4.1 Queue Head Initialization

One pair of device queue heads must be initialized for each active endpoint. To initialize a device queue head:

- Write the wMaxPacketSize field as required by the USB specification 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 for bandwidth with the USB specification chapter 9 protocol. In FS mode, the multiplier field can only be 1 for ISO endpoints.
- Set the next dTD terminate bit field.
- Clear the active bit in the status field.
- Clear the halt bit in the status field.

Note

The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTDs.

48.5.3.4.2 Setup Transfers Operation

As discussed in Control Endpoint Operation setup transfers require 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 manage the setup transfer by:

1. Copying setup buffer contents from dQH-RX to software buffer.
2. Acknowledging setup backup by writing a 1 to the corresponding bit in the EPSETUPSR register.

   Note

   The acknowledge must occur before continuing to process the setup packet. After acknowledge occurs, DCD must not attempt to access the setup buffer in dQH-RX. Only local software copy should be examined.

3. Checking for pending data or status dTD's from previous control transfers and flushing if any exist as discussed in Flushing/De-priming an Endpoint.

   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.

4. Decoding setup packet and prepare data phase (optional) and status phase transfer as required by the USB specification chapter 9 or application specific protocol.

48.5.3.5 Managing Transfers with Transfer Descriptors

48.5.3.5.1 Software Link Pointers

It is necessary for the DCD software to maintain head and tail pointers 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 executed. The operations described in the next section for managing dTDs assumes DCD can reference the head and tail of the dTD linked list.

Note

> To conserve memory, the reserved fields at the end of the dQH can be used to store the head and tail pointers, but DCD must continue maintaining the pointers.

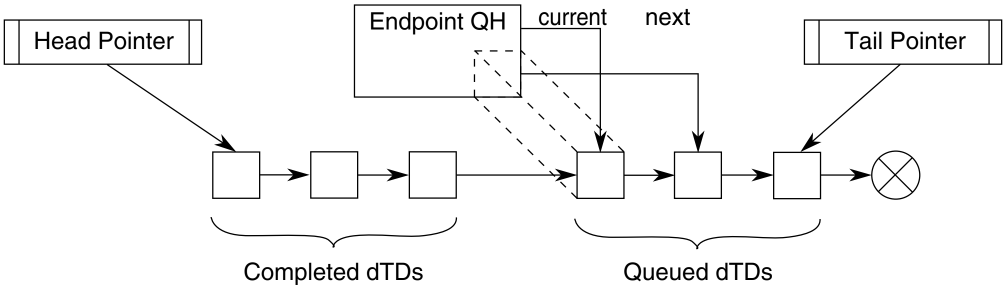


Figure 48-51. Software Link Pointers

Note

> Check the status of each dTD to determine completed status.

48.5.3.5.2 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 a 32-byte dTD block of memory aligned to 32-byte boundaries. The last 5 bits of the address must equal 00000.

Write the following fields:

1. Initialize the first 7 entries (28 bytes) to 0.
2. Set the terminate bit.
3. Fill in total bytes with transfer size.

4. Set the interrupt on complete bit if desired.
5. Initialize the status field with the active bit set, and all remaining status bits cleared.
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 pointers.

48.5.3.5.3 Executing a Transfer Descriptor

To safely add a dTD, the DCD must follow this procedure that manages the event where the device controller reaches the end of the dTD list. At the same time, a new dTD is added to the end of the list.

Determine whether the linked list is empty:

Check the DCD driver to see if the pipe is empty (internal representation of the linked list should indicate if any packets are outstanding)

Case 1: Link list is empty

1. Write dQH next pointer AND dQH terminate bit to 0 as a single 32-bit operation.
2. Clear active and halt bit in dQH (in case set from a previous error).
3. Prime endpoint by writing 1 to the correct bit position in the EPPRIME register.

Case 2: Link list is not empty

1. Add dTD to end of the linked list.
2. Read correct prime bit in EPPRIME - if set, DONE.
3. Set the USBCMD[ATDTW] bit.
4. Read correct status bit in EPSR, and store in a temporary variable for later.
5. Read the USBCMD[ATDTW] bit:

   If clear, go to 3.

   If set, continue to 6.
6. Clear the USBCMD[ATDTW] bit.
7. If status bit read in step 4 is 1 DONE.

8. If status bit read in step 4 is 0 then go to case 1, step 1.

48.5.3.5.4 Transfer Completion

After a dTD is initialized and the associated endpoint is primed, the device controller executes the transfer upon the host-initiated request. The DCD is notified with a USB interrupt if the interrupt-on-complete bit was set, or alternatively, the DCD can poll the endpoint complete register to determine when the dTD had been executed. After a dTD is executed, DCD can check the status bits to determine success or failure.

CAUTION

> Multiple dTDs can be completed in a single endpoint complete notification. After clearing the notification, the DCD must search the dTD linked list and retire all finished (active bit cleared) dTDs.

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 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 decrements by the actual bytes transferred. For transmit packets, a packet is only complete after the actual bytes reaches zero. However, for receive packets, the host may send fewer bytes in the transfer according the USB variable length packet protocol.

48.5.3.5.5 Flushing/De-priming an Endpoint

It is necessary for the DCD to flush or de-prime endpoints during a USB device reset or during a broken control transfer. There may also be application specific requirements to stop transfers in progress. The DCD can use this procedure to stop a transfer in progress:

1. Set the corresponding bit(s) in the EPFLUSH register.
2. Wait until all bits in the EPFLUSH register are cleared.

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 the EPSR register to ensure that for all endpoints commanded to be flushed, that the corresponding bits are now cleared. If the corresponding bits are set after step #2 has finished, flush failed as described below:

   In very rare cases, a packet is in progress to the particular endpoint when commanded to flush using EPFLUSH. 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 by repeating steps 1-3 until each endpoint successfully flushes.

48.5.3.5.6 Device Error Matrix

The following table summarizes packet errors not automatically managed by the USB OTG module.

Table 48-70. Device Error Matrix

Error	Direction	Packet Type	Data Buffer Error Bit	Transaction Error Bit
Data Buffer Overflow	RX	Any	1	0
ISO Packet Error	RX	ISO	0	1
ISO Fulfillment Error	Both	ISO	0	1

The device controller manages 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.

Table 48-71. Error Descriptions

Overflow	Number of bytes received exceeded max. packet size or total buffer length. **Note:** This error also sets the halt bit in the dQH, and if there are dTDs remaining in the linked list for the endpoint, those are not executed.
ISO Packet Error	CRC error on received ISO packet. Contents not guaranteed 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, DCD may need to readjust the data queue because a fulfillment error causes the 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.

48.5.4 Servicing Interrupts

The interrupt service routine must understand there are high frequency, low frequency, and error operations to order accordingly.

48.5.4.1 High Frequency Interrupts

In particular, high frequency interrupts must be managed 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.

Table 48-72. Interrupt Managing Order

Execution Order	Interrupt	Action
1a	USB Interrupt[1] EPSETUPSR	Copy contents of setup buffer and acknowledge setup packet (as indicated in Managing Queue Heads). Process setup packet according to USB specification chapter 9 or application specific protocol.
1b	USB Interrupt EPCOMPLETE	Manage completion of dTD as indicated in Managing Queue Heads.
2	SOF Interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

[1] It is likely multiple interrupts stack up on any call to the interrupt service routine and during interrupt service routine.

48.5.4.1.1 Low Frequency Interrupts

The low frequency events include the following interrupts. These interrupts can be managed in any order because they do not occur often in comparison to the high-frequency interrupts.

Table 48-73. Low Frequency Interrupt Events

Interrupt	Action
Port Change	Change software state information.
Sleep Enable (Suspend)	Change software state information. Low power managing as necessary.
Reset Received	Change software state information. Abort pending transfers.

48.5.4.1.2 Error Interrupts

Error interrupts are least frequent and should be placed last in the interrupt service routine.

Table 48-74. 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 more aptly manages packet-level errors by checking the dTD status field upon receipt of USB interrupt (w/ EPCOMPLETE).
System Error	Unrecoverable error. Immediate reset of module; free transfers buffers in progress and restart the DCD.

48.5.5 Deviations from the EHCI Specifications

The host mode operation of the USB OTG module is nearly EHCI-compatible with a few minor differences. For the most part, the modules conform to the data structures and operations described in Section 3, "Data Structures," and Section 4, "Operational Model," in the EHCI specification. The particulars of the deviations occur in the following areas:

- Embedded transaction translator—Allows direct attachment of HS, 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; therefore, device mode is mostly transparent when in host mode. However, there are a couple exceptions documented in the following sections.
- Embedded design interface—The module does not have a PCI Interface and therefore the PCI configuration registers described in the EHCI specification are not applicable.

For the purposes of the USB OTG implementing a dual-role host/device controller with support for OTG applications, it is necessary to deviate from the EHCI specification. Device and OTG operation are not specified in the EHCI specification, and thus the implementation supported in the USB OTG module is proprietary.

48.5.5.1 Embedded Transaction Translator Function

The USB host mode supports directly connected high-, 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 is 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 existing in the EHCI specification to support full- and low-speed devices.

48.5.5.1.1 Capability Registers

These additions to the capability registers support the embedded Transaction translator function:

- N_TT added to HSCPARAMS - Host Controller Structural Parameters
- N_PTT added to HSCPARAMS - Host Controller Structural Parameters

Refer to the section for the HCSPARAMS register for usage information.

48.5.5.1.2 Operational Registers

These additions to the operational registers support the embedded TT:

- Addition of the TTCTRL register.
- Addition of a two-bit port speed (PSPD) field to the PORTSC*n* register.

48.5.5.1.3 Discovery

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 is set only in a standard EHCI controller implementation after the port reset operation and when the host and device negotiate a high-speed connection (chirp completes successfully).

The module always sets the port enable bit after the port reset operation regardless of the result of the host device chirp result, and the resulting port speed is indicated by the PORTSC*n*[PSPD] field. Therefore, the standard EHCI host controller driver requires an alteration to manage directly connected full- and low-speed devices or hubs. The change is a fundamental one summarized in the next table.

Table 48-75. Functional Differences Between EHCI and EHCI with Embedded TT

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 PORTSC*n*.
FS and LS devices are assumed to be downstream from a HS hub. Therefore, all port-level control performs through the hub class to the nearest hub.	FS and LS device can be downstream from a HS hub or directly attached. When the FS/LS device is downstream from a HS hub, port-level control acts using the hub class through the nearest hub. When a FS/LS device is directly attached, then port-level control is accomplished using PORTSC*n*.

Table continues on the next page...

Table 48-75. Functional Differences Between EHCI and EHCI with Embedded TT (continued)

Standard EHCI	EHCI with embedded Transaction Translator
FS and LS devices are assumed to be downstream from a HS hub with HubAddr equal to X. [where HubAddr > 0 and HubAddr is the address of the hub where the bus transitions from HS to FS/LS (split target hub)]	FS and LS device can be downstream from a HS hub with HubAddr equal to X [HubAddr > 0] or directly attached [where HubAddr equals 0 and HubAddr is the address of the root hub where the bus transitions from HS to FS/LS (split target hub is the root hub)]

48.5.5.1.4 Data Structures

The same data structures used for FS/LS transactions though a HS hub are also used for transactions through the root hub. It is demonstrated here how hub address and endpoint speed fields should be set for directly attached FS/LS devices and hubs:

1. QH (for direct attach FS/LS) – asynchronous (bulk/control endpoints) periodic (interrupt)
   - Hub address equals 0
   - Transactions to direct attached device/hub.
     - QH.EPS equals port speed
   - Transactions to a device downstream from direct attached FS hub.
     - QH.EPS equals downstream device speed

   Note

   When QH.EPS equals 01 (LS) and PORTSC*n*[PSPD] equals 00 (FS), a LS-pre-PID is sent before transmitting LS traffic.

   Maximum packet size must equal 64 or less to prevent undefined behavior.

2. siTD (for direct attach FS) – Periodic (ISO endpoint)
   - All FS ISO transactions:
     - Hub address equals 0
     - siTD.EPS equals 00 (full speed)

   Maximum packet size must equal to 1023 or less to prevent undefined behavior.

48.5.5.1.5 Operational Model

The operational models are well defined for the behavior of the transaction translator (see USB 2.0 specification) and for the EHCI controller moving packets between system memory and a USB-HS hub. Because the embedded transaction translator exists within the USB host controller, no physical bus between EHCI host controller driver and the USB FS/LS bus. These sections briefly discuss the operational model for how the EHCI and transaction translator operational models combine without the physical bus between. The following sections assume the reader is familiar with the EHCI and USB 2.0 transaction translator operational models.

48.5.5.1.5.1 Microframe 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 uses 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 microframe pipeline implemented in the embedded transaction translator, all periodic transfers that are tagged in EHCI to execute in H-frame 0 are ready to execute on the bus in B-frame 0.

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.

After periodic transfers are exhausted, any stored asynchronous transfer is moved. Asynchronous transfers are opportunistic because they execute when possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer cannot babble through the SOF (start of B-frame 0.)

48.5.5.1.5.2 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 next table summarizes the conditions where handshakes are emulated from internal state instead of actual handshakes to HS split bus traffic.

Table 48-76. Emulated Handshakes

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 FS/LS device

48.5.5.1.5.3 Asynchronous Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- USB 2.0 – 11.17.3
    - Sequencing is provided and a packet length estimator ensures no full-/low-speed packet babbles into SOF time.
- USB 2.0 – 11.17.4
    - • Transaction tracking for 2 data pipes.
- USB 2.0 – 11.17.5
    - • Clear_TT_Buffer capability provided though the use of the TTCTRL register.

48.5.5.1.5.4 Periodic Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded transaction translator:

- USB 2.0 – 11.18.6.[1-2]
    - Abort of pending start-splits
        - EOF (and not started in microframes 6)
        - Idle for more than 4 microframes
    - Abort of pending complete-splits

        - EOF
        - Idle for more than 4 microframes
- USB 2.0 - 11.18.[7-8]
    - Transaction tracking for up to four data pipes.
        - No more than 4 periodic transactions (interrupt/isochronous) can be scheduled through the embedded TT per frame.
    - Complete-split transaction searching.

Note

There is no data schedule mechanism for these transactions other than the microframe 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.

48.5.5.2 Device Operation

The co-existence of a device operational controller within the USB OTG module has little effect on EHCI compatibility for host operation. However, given that the USB OTG controller initializes in neither host nor device mode, the USBMODE register must be programmed for host operation before the EHCI host controller driver can begin EHCI host operations.

48.5.5.3 Non-Zero Fields in the Register File

Some of the reserved fields and reserved addresses in the capability registers and operational registers have use in device mode. Adhere to these steps:

- Write operations to all EHCI reserved fields (some of which are device fields in the USB OTG module) in 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 module must properly mask EHCI reserved fields (some of which are device fields in the USB OTG module registers).

48.5.5.4 SOF Interrupt

The SOF interrupt is a free running 125 µs or 1 ms interrupt for host mode. EHCI does not specify this interrupt, but it has been added for convenience and as a potential software time base. The free running interrupt is shared with the device mode start-of-frame interrupt. See the sections for the USBSTS and USBINTR registers for more information.

48.5.5.5 Embedded Design

This is an embedded USB host controller as defined by the EHCI specification; therefore, it does not implement the PCI configuration registers.

48.5.5.5.1 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 those provided by the frame adjust register in the PCI configuration registers. Starts of microframes are timed precisely to 125 µs using the transceiver clock as a reference clock or a 60 Mhz transceiver clock for 8-bit physical interfaces and full-speed serial interfaces.

48.5.5.6 Miscellaneous Variations from EHCI

48.5.5.6.1 Programmable Physical Interface Behavior

The modules support multiple physical interfaces that can operate in different modes when the module is configured with the software programmable physical interface modes. The control bits for selecting the PHY operating mode are added to the PORTSC*n* register providing a capability not defined by the EHCI specification.

48.5.5.6.2 Discovery

48.5.5.6.2.1 Port Reset

The port connect methods specified by EHCI require setting the port reset bit in the PORTSC*n* register for a duration of 10 ms. Due to the complexity required to support the attachment of devices not high speed, a counter is present in the design that can count the 10 ms reset pulse to alleviate the requirement of the software to measure this duration. Therefore, the basic connection is summarized as:

- Port change interrupt—Port connect change occurs to notify the host controller driver that a device has attached.
- Software shall set the PORTSC*n*[PR] bit to reset the device.
- Software shall clear the PORTSC*n*[PR] bit after 10 ms.
    - This step, 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 is ignored and the reset continues until completion.
- Port change interrupt—Port enable change occurs to notify the host controller that the device is now operational and at this point the port speed is determined.

48.5.5.6.2.2 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 re-assigns the port owner for any device that does not connect at high speed, this host controller supports direct attach of non-HS devices. Therefore, the following differences are important regarding port speed detection:

- Port owner hand-off is not implemented. Therefore, PORTSC*n*[PO] bit is read-only and always reads 0.
- A 2-bit port speed indicator field has been added to PORTSC*n* to provide the current operating speed of the port to the host controller driver.
- A 1-bit high-speed indicator bit has been added to PORTSC*n* to signify that the port is in HS vs. FS/LS.
    - This information is redundant with the 2-bit port speed indicator field above.

Chapter 49
CAN (FlexCAN)

49.1 Introduction

NOTE

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

The FlexCAN module is a communication controller implementing the CAN protocol according to the CAN 2.0B protocol specification. A general block diagram is shown in the following figure, which describes the main sub-blocks implemented in the FlexCAN module, including one associated memory for storing Message Buffers, Rx Global Mask Registers, Rx Individual Mask Registers, Rx FIFO and Rx FIFO ID Filters. The functions of the sub-modules are described in subsequent sections.

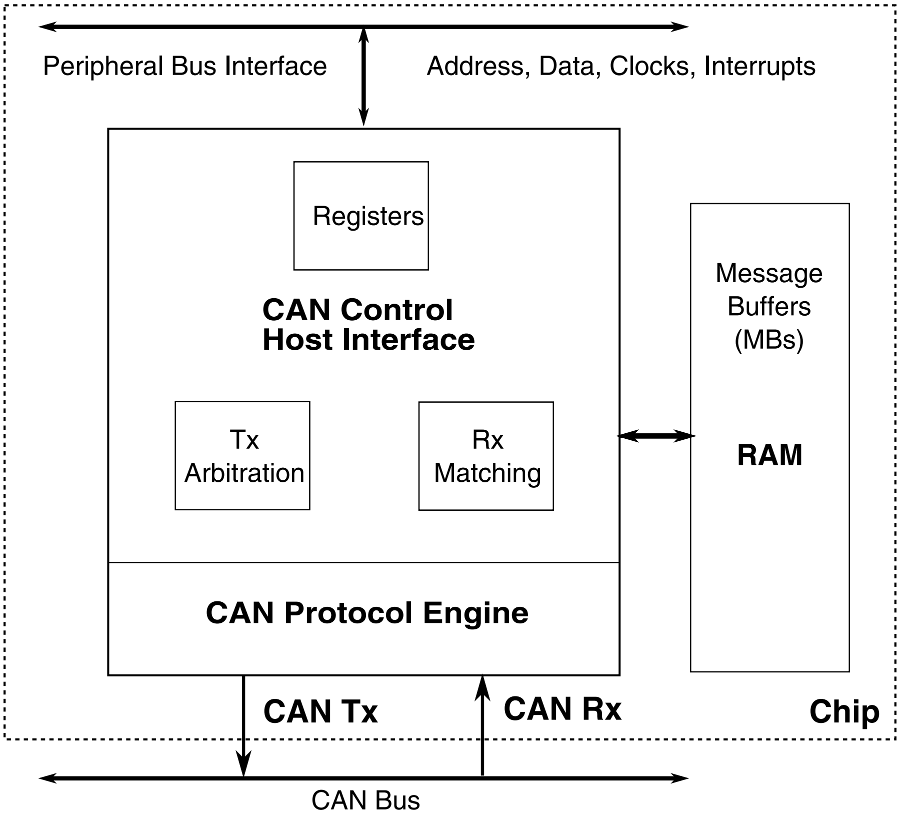


Figure 49-1. FlexCAN Block Diagram

49.1.1 Overview

The CAN protocol was primarily, but not only, designed to be used as a vehicle serial data bus, meeting the specific requirements of this field: real-time processing, reliable operation in the EMI environment of a vehicle, cost-effectiveness and required bandwidth. The FlexCAN module is a full implementation of the CAN protocol specification, Version 2.0 B, which supports both standard and extended message frames. The Message Buffers are stored in an embedded RAM dedicated to the FlexCAN module. See the Chip Configuration details for the actual number of Message Buffers configured in the MCU.

The CAN Protocol Engine (PE) sub-module manages the serial communication on the CAN bus, requesting RAM access for receiving and transmitting message frames, validating received messages and performing error handling. The Controller Host Interface (CHI) sub-module handles Message Buffer selection for reception and transmission, taking care of arbitration and ID matching algorithms. The Bus Interface Unit (BIU) sub-module controls the access to and from the internal interface bus, in order to establish connection to the CPU and to other blocks. Clocks, address and data buses, interrupt outputs and test signals are accessed through the Bus Interface Unit.

49.1.2 FlexCAN Module Features

The FlexCAN module includes these distinctive legacy features:

- Full Implementation of the CAN protocol specification, Version 2.0B
    - Standard data and remote frames
    - Extended data and remote frames
    - Zero to eight bytes data length
    - Programmable bit rate up to 1 Mb/sec
    - Content-related addressing
- Flexible Mailboxes of zero to eight bytes data length
- Each Mailbox configurable as Rx or Tx, all supporting standard and extended messages
- Individual Rx Mask Registers per Mailbox
- Full featured Rx FIFO with storage capacity for up to 6 frames and automatic internal pointer handling
- Transmission abort capability
- Programmable clock source to the CAN Protocol Interface, either bus clock or crystal oscillator
- Unused structures space can be used as general purpose RAM space
- Listen-only mode capability
- Programmable loop-back mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number or highest priority
- Time Stamp based on 16-bit free-running timer
- Global network time, synchronized by a specific message
- Maskable interrupts
- Independent of the transmission medium (an external transceiver is assumed)

- Short latency time due to an arbitration scheme for high-priority messages
- Low power modes, with programmable wake up on bus activity

Furthermore, the new major features below are also provided in addition to the previous FlexCAN version:

- Remote request frames may be handled automatically or by software
- Safe mechanism for ID Filter configuration in Normal Mode
- CAN bit time settings and configuration bits can only be written in "Freeze" mode
- Tx mailbox status (Lowest priority buffer or empty buffer)
- IDHIT register for received frames
- SYNC bit status to inform that the module is synchronous with CAN bus
- Debug Registers
- CRC status for transmitted message
- Rx FIFO Global Mask register
- Selectable priority of reception between Mailboxes and Rx FIFO during matching process
- Powerful Rx FIFO ID filtering, capable of matching incoming IDs against either 128 extended, 256 standard or 512 partial (8 bits) IDs, with up to 32 individual masking capability
- 100% backward compatibility with previous FlexCAN version

49.1.3 Modes of Operation

The FlexCAN module has four functional modes: Normal Mode (User and Supervisor), Freeze Mode, Listen-Only Mode and Loop-Back Mode. There are also three low power modes: Disable Mode, Doze Mode and Stop Mode.

- Normal Mode (User or Supervisor):

  In Normal Mode, the module operates receiving and/or transmitting message frames, errors are handled normally and all the CAN Protocol functions are enabled. User and Supervisor Modes differ in the access to some restricted control registers.

- Freeze Mode:

It is enabled when the FRZ bit in the MCR Register is asserted. If enabled, Freeze Mode is entered when the HALT bit in MCR is set or when Debug Mode is requested at MCU level and the FRZ_ACK bit in the MCR Register is asserted by the FlexCAN. In this mode, no transmission or reception of frames is done and synchronicity to the CAN bus is lost. See Freeze Mode for more information.

- Listen-Only Mode:

  The module enters this mode when the LOM bit in the Control 1 Register is asserted. In this mode, transmission is disabled, all error counters are frozen and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.

- Loop-Back Mode:

  The module enters this mode when the LPB bit in the Control 1 Register is asserted. In this mode, FlexCAN performs an internal loop back that can be used for self test operation. The bit stream output of the transmitter is internally fed back to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic '1'). FlexCAN behaves as it normally does when transmitting and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

- Module Disable Mode:

  This low power mode is entered when the MDIS bit in the MCR Register is asserted by the CPU and the LPM_ACK is asserted by the FlexCAN. When disabled, the module requests to disable the clocks to the CAN Protocol Engine and Controller Host Interface sub-modules. Exit from this mode is done by negating the MDIS bit in the MCR Register. See Module Disable Mode for more information.

- Doze Mode:

  This low power mode is entered when the DOZE bit in MCR is asserted and Doze Mode is requested at MCU level and the LPM_ACK bit in the MCR Register is asserted by the FlexCAN. When in Doze Mode, the module requests to disable the clocks to the CAN Protocol Engine and the CAN Controller-Host Interface sub-modules. Exit from this mode happens when the DOZE bit in MCR is negated, when the MCU is removed from Doze Mode, or when activity is detected on the CAN bus and the Self Wake Up mechanism is enabled. See Doze Mode for more information.

- Stop Mode:

This low power mode is entered when Stop Mode is requested at MCU level and the LPM_ACK bit in the MCR Register is asserted by the FlexCAN. When in Stop Mode, the module puts itself in an inactive state and then informs the CPU that the clocks can be shut down globally. Exit from this mode happens when the Stop Mode request is removed or when activity is detected on the CAN bus and the Self Wake Up mechanism is enabled. See Stop Mode for more information.

49.2 FlexCAN Signal Descriptions

The FlexCAN module has two I/O signals connected to the external MCU pins. These signals are summarized in the following table and described in more detail in the next sub-sections.

Table 49-1. FlexCAN Signal Descriptions

Signal	Description	I/O
CAN Rx	CAN Receive Pin	Input
CAN Tx	CAN Transmit Pin	Output

49.2.1 CAN Rx

This pin is the receive pin from the CAN bus transceiver. Dominant state is represented by logic level '0'. Recessive state is represented by logic level '1'.

49.2.2 CAN Tx

This pin is the transmit pin to the CAN bus transceiver. Dominant state is represented by logic level '0'. Recessive state is represented by logic level '1'.

49.3 Memory Map/Register Definition

This section describes the registers and data structures in the FlexCAN module. The base address of the module depends on the particular memory map of the MCU.

49.3.1 FlexCAN Memory Mapping

The complete memory map for a FlexCAN module is shown in the following table.

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address 0x0080.

Each individual register is identified by its complete name and the corresponding mnemonic. The access type can be Supervisor (S) or Unrestricted (U). Most of the registers can be configured to have either Supervisor or Unrestricted access by programming the SUPV bit in the MCR Register. These registers are identified as S/U in the Access column of Table 49-2.

The registers IFLAG2 and IMASK2 are considered reserved space depending on the number of Mailboxes available in the device.

Table 49-2. Module Memory Map

Register	Access Type	Affected by Hard Reset	Affected by Soft Reset
Module Configuration Register (MCR)	S	Yes	Yes
Control 1 Register (CTRL1)	S/U	Yes	No
Free Running Timer Register (TIMER)	S/U	Yes	Yes
Rx Mailboxes Global Mask Register (RXMGMASK)	S/U	No	No
Rx Buffer 14 Mask Register (RX14MASK)	S/U	No	No
Rx Buffer 15 Mask Register (RX15MASK)	S/U	No	No
Error Counter Register (ECR)	S/U	Yes	Yes
Error and Status 1 Register (ESR1)	S/U	Yes	Yes
Interrupt Masks 2 Register (IMASK2)	S/U	Yes	Yes
Interrupt Masks 1 Register (IMASK1)	S/U	Yes	Yes
Interrupt Flags 2 Register (IFLAG2)	S/U	Yes	Yes
Interrupt Flags 1 Register (IFLAG1)	S/U	Yes	Yes
Control 2 Register (CTRL2)	S/U	Yes	No
Error and Status 2 Register (ESR2)	S/U	Yes	Yes
Individual Matching Elements Update Register (IMUER)	S/U	Yes	Yes
Lost Rx Frames Register (LRFR)	S/U	Yes	Yes
CRC Register (CRCR)	S/U	Yes	Yes
Rx FIFO Global Mask Register (RXFGMASK)	S/U	No	No
Rx FIFO Information Register (RXFIR)	S/U	No	No
Message Buffers	S/U	No	No
Rx Individual Mask Registers	S/U	No	No

The FlexCAN module can store CAN messages for transmission and reception using Mailboxes and Rx FIFO structures.

This module's memory map includes sixteen 128-bit message buffers (MBs) that occupy the range from offset 0x80 to 0x17F.

CAN memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_4000	Module Configuration Register (CAN0_MCR)	32	R/W	D890_000Fh	49.3.2/ 1444
4002_4004	Control 1 Register (CAN0_CTRL1)	32	R/W	0000_0000h	49.3.3/ 1449
4002_4008	Free Running Timer (CAN0_TIMER)	32	R/W	0000_0000h	49.3.4/ 1452
4002_4010	Rx Mailboxes Global Mask Register (CAN0_RXMGMASK)	32	R/W	FFFF_ FFFFh	49.3.5/ 1453
4002_4014	Rx 14 Mask Register (CAN0_RX14MASK)	32	R/W	FFFF_ FFFFh	49.3.6/ 1454
4002_4018	Rx 15 Mask Register (CAN0_RX15MASK)	32	R/W	FFFF_ FFFFh	49.3.7/ 1455
4002_401C	Error Counter (CAN0_ECR)	32	R/W	0000_0000h	49.3.8/ 1456
4002_4020	Error and Status 1 Register (CAN0_ESR1)	32	R/W	0000_0000h	49.3.9/ 1457
4002_4024	Interrupt Masks 2 Register (CAN0_IMASK2)	32	R/W	0000_0000h	49.3.10/ 1461
4002_4028	Interrupt Masks 1 Register (CAN0_IMASK1)	32	R/W	0000_0000h	49.3.11/ 1462
4002_402C	Interrupt Flags 2 Register (CAN0_IFLAG2)	32	R/W	0000_0000h	49.3.12/ 1462
4002_4030	Interrupt Flags 1 Register (CAN0_IFLAG1)	32	R/W	0000_0000h	49.3.13/ 1463
4002_4034	Control 2 Register (CAN0_CTRL2)	32	R/W	00B0_0000h	49.3.14/ 1466
4002_4038	Error and Status 2 Register (CAN0_ESR2)	32	R/W	0000_0000h	49.3.15/ 1469
4002_4044	CRC Register (CAN0_CRCR)	32	R	0000_0000h	49.3.16/ 1470
4002_4048	Rx FIFO Global Mask Register (CAN0_RXFGMASK)	32	R/W	FFFF_ FFFFh	49.3.17/ 1471
4002_404C	Rx FIFO Information Register (CAN0_RXFIR)	32	R	Undefined	49.3.18/ 1472

Table continues on the next page...

CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_4880	Rx Individual Mask Registers (CAN0_RXIMR0)	32	R/W	Undefined	49.3.19/ 1473
4002_4884	Rx Individual Mask Registers (CAN0_RXIMR1)	32	R/W	Undefined	49.3.19/ 1473
4002_4888	Rx Individual Mask Registers (CAN0_RXIMR2)	32	R/W	Undefined	49.3.19/ 1473
4002_488C	Rx Individual Mask Registers (CAN0_RXIMR3)	32	R/W	Undefined	49.3.19/ 1473
4002_4890	Rx Individual Mask Registers (CAN0_RXIMR4)	32	R/W	Undefined	49.3.19/ 1473
4002_4894	Rx Individual Mask Registers (CAN0_RXIMR5)	32	R/W	Undefined	49.3.19/ 1473
4002_4898	Rx Individual Mask Registers (CAN0_RXIMR6)	32	R/W	Undefined	49.3.19/ 1473
4002_489C	Rx Individual Mask Registers (CAN0_RXIMR7)	32	R/W	Undefined	49.3.19/ 1473
4002_48A0	Rx Individual Mask Registers (CAN0_RXIMR8)	32	R/W	Undefined	49.3.19/ 1473
4002_48A4	Rx Individual Mask Registers (CAN0_RXIMR9)	32	R/W	Undefined	49.3.19/ 1473
4002_48A8	Rx Individual Mask Registers (CAN0_RXIMR10)	32	R/W	Undefined	49.3.19/ 1473
4002_48AC	Rx Individual Mask Registers (CAN0_RXIMR11)	32	R/W	Undefined	49.3.19/ 1473
4002_48B0	Rx Individual Mask Registers (CAN0_RXIMR12)	32	R/W	Undefined	49.3.19/ 1473
4002_48B4	Rx Individual Mask Registers (CAN0_RXIMR13)	32	R/W	Undefined	49.3.19/ 1473
4002_48B8	Rx Individual Mask Registers (CAN0_RXIMR14)	32	R/W	Undefined	49.3.19/ 1473
4002_48BC	Rx Individual Mask Registers (CAN0_RXIMR15)	32	R/W	Undefined	49.3.19/ 1473
400A_4000	Module Configuration Register (CAN1_MCR)	32	R/W	D890_000Fh	49.3.2/ 1444
400A_4004	Control 1 Register (CAN1_CTRL1)	32	R/W	0000_0000h	49.3.3/ 1449
400A_4008	Free Running Timer (CAN1_TIMER)	32	R/W	0000_0000h	49.3.4/ 1452
400A_4010	Rx Mailboxes Global Mask Register (CAN1_RXMGMASK)	32	R/W	FFFF_ FFFFh	49.3.5/ 1453

Table continues on the next page...

CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400A_4014	Rx 14 Mask Register (CAN1_RX14MASK)	32	R/W	FFFF_ FFFFh	49.3.6/ 1454
400A_4018	Rx 15 Mask Register (CAN1_RX15MASK)	32	R/W	FFFF_ FFFFh	49.3.7/ 1455
400A_401C	Error Counter (CAN1_ECR)	32	R/W	0000_0000h	49.3.8/ 1456
400A_4020	Error and Status 1 Register (CAN1_ESR1)	32	R/W	0000_0000h	49.3.9/ 1457
400A_4024	Interrupt Masks 2 Register (CAN1_IMASK2)	32	R/W	0000_0000h	49.3.10/ 1461
400A_4028	Interrupt Masks 1 Register (CAN1_IMASK1)	32	R/W	0000_0000h	49.3.11/ 1462
400A_402C	Interrupt Flags 2 Register (CAN1_IFLAG2)	32	R/W	0000_0000h	49.3.12/ 1462
400A_4030	Interrupt Flags 1 Register (CAN1_IFLAG1)	32	R/W	0000_0000h	49.3.13/ 1463
400A_4034	Control 2 Register (CAN1_CTRL2)	32	R/W	00B0_0000h	49.3.14/ 1466
400A_4038	Error and Status 2 Register (CAN1_ESR2)	32	R/W	0000_0000h	49.3.15/ 1469
400A_4044	CRC Register (CAN1_CRCR)	32	R	0000_0000h	49.3.16/ 1470
400A_4048	Rx FIFO Global Mask Register (CAN1_RXFGMASK)	32	R/W	FFFF_ FFFFh	49.3.17/ 1471
400A_404C	Rx FIFO Information Register (CAN1_RXFIR)	32	R	Undefined	49.3.18/ 1472
400A_4880	Rx Individual Mask Registers (CAN1_RXIMR0)	32	R/W	Undefined	49.3.19/ 1473
400A_4884	Rx Individual Mask Registers (CAN1_RXIMR1)	32	R/W	Undefined	49.3.19/ 1473
400A_4888	Rx Individual Mask Registers (CAN1_RXIMR2)	32	R/W	Undefined	49.3.19/ 1473
400A_488C	Rx Individual Mask Registers (CAN1_RXIMR3)	32	R/W	Undefined	49.3.19/ 1473
400A_4890	Rx Individual Mask Registers (CAN1_RXIMR4)	32	R/W	Undefined	49.3.19/ 1473
400A_4894	Rx Individual Mask Registers (CAN1_RXIMR5)	32	R/W	Undefined	49.3.19/ 1473
400A_4898	Rx Individual Mask Registers (CAN1_RXIMR6)	32	R/W	Undefined	49.3.19/ 1473

Table continues on the next page...

CAN memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400A_489C	Rx Individual Mask Registers (CAN1_RXIMR7)	32	R/W	Undefined	49.3.19/ 1473
400A_48A0	Rx Individual Mask Registers (CAN1_RXIMR8)	32	R/W	Undefined	49.3.19/ 1473
400A_48A4	Rx Individual Mask Registers (CAN1_RXIMR9)	32	R/W	Undefined	49.3.19/ 1473
400A_48A8	Rx Individual Mask Registers (CAN1_RXIMR10)	32	R/W	Undefined	49.3.19/ 1473
400A_48AC	Rx Individual Mask Registers (CAN1_RXIMR11)	32	R/W	Undefined	49.3.19/ 1473
400A_48B0	Rx Individual Mask Registers (CAN1_RXIMR12)	32	R/W	Undefined	49.3.19/ 1473
400A_48B4	Rx Individual Mask Registers (CAN1_RXIMR13)	32	R/W	Undefined	49.3.19/ 1473
400A_48B8	Rx Individual Mask Registers (CAN1_RXIMR14)	32	R/W	Undefined	49.3.19/ 1473
400A_48BC	Rx Individual Mask Registers (CAN1_RXIMR15)	32	R/W	Undefined	49.3.19/ 1473

49.3.2 Module Configuration Register (CANx_MCR)

This register defines global system configurations, such as the module operation modes and the maximum message buffer configuration.

Addresses: CAN0_MCR is 4002_4000h base + 0h offset = 4002_4000h
CAN1_MCR is 400A_4000h base + 0h offset = 400A_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MDIS	FRZ	RFEN	HALT	NOTRDY	WAKMSK	SOFTRST	FRZACK	SUPV	SLFWAK	WRNEN	LPMACK	Reserved	0	SRXDIS	IRMQ
W					▨			▨				▨		▨		
Reset	1	1	0	1	1	0	0	0	1	0	0	1	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		LPRIOEN	AEN	0		IDAM		0	MAXMB						
W	▨				▨				▨							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

CANx_MCR field descriptions

Field	Description
31 MDIS	Module Disable This bit controls whether FlexCAN is enabled or not. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface sub-modules. This is the only bit in MCR not affected by soft reset. 0 Enable the FlexCAN module. 1 Disable the FlexCAN module.
30 FRZ	Freeze Enable The FRZ bit specifies the FlexCAN behavior when the HALT bit in the MCR Register is set or when Debug Mode is requested at MCU level. When FRZ is asserted, FlexCAN is enabled to enter Freeze Mode. Negation of this bit field causes FlexCAN to exit from Freeze Mode. 0 Not enabled to enter Freeze Mode 1 Enabled to enter Freeze Mode
29 RFEN	Rx FIFO Enable

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	This bit controls whether the Rx FIFO feature is enabled or not. When RFEN is set, MBs 0 to 5 cannot be used for normal reception and transmission because the corresponding memory region (0x80-0xDC) is used by the FIFO engine as well as additional MBs (up to 32, depending on CTRL2[RFFN] setting) which are used as Rx FIFO ID Filter Table elements. RFEN also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in the table "Minimum Ratio Between Peripheral Clock Frequency and CAN Bit Rate" (in section "Arbitration and Matching Timing"). This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Rx FIFO not enabled 1 Rx FIFO enabled
28 HALT	Halt FlexCAN Assertion of this bit puts the FlexCAN module into Freeze Mode. The CPU should clear it after initializing the Message Buffers and Control Register. No reception or transmission is performed by FlexCAN before this bit is cleared. Freeze Mode cannot be entered while FlexCAN is in a low power mode. 0 No Freeze Mode request. 1 Enters Freeze Mode if the FRZ bit is asserted.
27 NOTRDY	FlexCAN Not Ready This read-only bit indicates that FlexCAN is either in Disable Mode, Stop Mode or Freeze Mode. It is negated once FlexCAN has exited these modes. 0 FlexCAN module is either in Normal Mode, Listen-Only Mode or Loop-Back Mode. 1 FlexCAN module is either in Disable Mode, Stop Mode or Freeze Mode.
26 WAKMSK	Wake Up Interrupt Mask This bit enables the Wake Up Interrupt generation. 0 Wake Up Interrupt is disabled. 1 Wake Up Interrupt is enabled.
25 SOFTRST	Soft Reset When this bit is asserted, FlexCAN resets its internal state machines and some of the memory mapped registers. The following registers are reset: MCR (except the MDIS bit), TIMER, ECR, ESR1, ESR2, IMASK1, IMASK2, IFLAG1, IFLAG2 and CRCR. Configuration registers that control the interface to the CAN bus are not affected by soft reset. The following registers are unaffected: CTRL1, CTRL2, RXIMR0–RXIMR63, RXMGMASK, RX14MASK, RX15MASK, RXFGMASK, RXFIR, all Message Buffers. The SOFTRST bit can be asserted directly by the CPU when it writes to the MCR Register, but it is also asserted when global soft reset is requested at MCU level. Since soft reset is synchronous and has to follow a request/acknowledge procedure across clock domains, it may take some time to fully propagate its effect. The SOFTRST bit remains asserted while reset is pending, and is automatically negated when reset completes. Therefore, software can poll this bit to know when the soft reset has completed. Soft reset cannot be applied while clocks are shut down in a low power mode. The module should be first removed from low power mode, and then soft reset can be applied. 0 No reset request 1 Resets the registers affected by soft reset.
24 FRZACK	Freeze Mode Acknowledge This read-only bit indicates that FlexCAN is in Freeze Mode and its prescaler is stopped. The Freeze Mode request cannot be granted until current transmission or reception processes have finished.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	Therefore the software can poll the FRZACK bit to know when FlexCAN has actually entered Freeze Mode. If Freeze Mode request is negated, then this bit is negated once the FlexCAN prescaler is running again. If Freeze Mode is requested while FlexCAN is in a low power mode, then the FRZACK bit will only be set when the low power mode is exited. See Section "Freeze Mode". **NOTE:** FRZACK will be asserted within 178 CAN bits from the freeze mode request by the CPU, and negated within 2 CAN bits after the freeze mode request removal (see Section "Protocol Timing"). 0 FlexCAN not in Freeze Mode, prescaler running 1 FlexCAN in Freeze Mode, prescaler stopped
23 SUPV	Supervisor Mode This bit configures the FlexCAN to be either in Supervisor or User Mode. The registers affected by this bit are marked as S/U in the Access Type column of the module memory map. Reset value of this bit is ‘1’, so the affected registers start with Supervisor access allowance only. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 FlexCAN is in User Mode. Affected registers allow both Supervisor and Unrestricted accesses. 1 FlexCAN is in Supervisor Mode. Affected registers allow only Supervisor access. Unrestricted access behaves as though the access was done to an unimplemented register location.
22 SLFWAK	Self Wake Up This bit enables the Self Wake Up feature when FlexCAN is in a low power mode other than Disable Mode. When this feature is enabled, the FlexCAN module monitors the bus for wake up event, that is, a recessive-to-dominant transition. If a wake up event is detected during Stop Mode, then FlexCAN generates, if enabled to do so, a Wake Up interrupt to the CPU so that it can exit Stop Mode globally and FlexCAN can request to resume the clocks. When FlexCAN is in a low power mode other than Disable Mode, this bit cannot be written as it is blocked by hardware. 0 FlexCAN Self Wake Up feature is disabled. 1 FlexCAN Self Wake Up feature is enabled.
21 WRNEN	Warning Interrupt Enable When asserted, this bit enables the generation of the TWRNINT and RWRNINT flags in the Error and Status Register. If WRNEN is negated, the TWRNINT and RWRNINT flags will always be zero, independent of the values of the error counters, and no warning interrupt will ever be generated. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 TWRNINT and RWRNINT bits are zero, independent of the values in the error counters. 1 TWRNINT and RWRNINT bits are set when the respective error counter transitions from less than 96 to greater than or equal to 96.
20 LPMACK	Low Power Mode Acknowledge This read-only bit indicates that FlexCAN is in a low power mode (Disable Mode, Stop Mode). A low power mode can not be entered until all current transmission or reception processes have finished, so the CPU can poll the LPMACK bit to know when FlexCAN has actually entered low power mode. **NOTE:** LPMACK will be asserted within 180 CAN bits from the low power mode request by the CPU, and negated within 2 CAN bits after the low power mode request removal (see Section "Protocol Timing").

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	0 FlexCAN is not in a low power mode. 1 FlexCAN is in a low power mode.
19 Reserved	This field is reserved.
18 Reserved	This read-only field is reserved and always has the value zero.
17 SRXDIS	Self Reception Disable This bit defines whether FlexCAN is allowed to receive frames transmitted by itself. If this bit is asserted, frames transmitted by the module will not be stored in any MB, regardless if the MB is programmed with an ID that matches the transmitted frame, and no interrupt flag or interrupt signal will be generated due to the frame reception. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Self reception enabled 1 Self reception disabled
16 IRMQ	Individual Rx Masking and Queue Enable This bit indicates whether Rx matching process will be based either on individual masking and queue or on masking scheme with RXMGMASK, RX14MASK and RX15MASK, RXFGMASK. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Individual Rx masking and queue feature are disabled. For backward compatibility, the reading of C/S word locks the MB even if it is EMPTY. 1 Individual Rx masking and queue feature are enabled.
15–14 Reserved	This read-only field is reserved and always has the value zero.
13 LPRIOEN	Local Priority Enable This bit is provided for backwards compatibility reasons. It controls whether the local priority feature is enabled or not. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word, but the actual transmitted ID still has 11-bit for standard frames and 29-bit for extended frames. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Local Priority disabled 1 Local Priority enabled
12 AEN	Abort Enable This bit is supplied for backwards compatibility reasons. When asserted, it enables the Tx abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. **NOTE:** When MCR[AEN] is asserted, only the abort mechanism (see Section "Transmission Abort Mechanism") must be used for updating Mailboxes configured for transmission. **CAUTION:** Writing the Abort code into Rx Mailboxes can cause unpredictable results when the MCR[AEN] is asserted.

Table continues on the next page...

CANx_MCR field descriptions (continued)

Field	Description
	0 Abort disabled 1 Abort enabled
11–10 Reserved	This read-only field is reserved and always has the value zero.
9–8 IDAM	ID Acceptance Mode This 2-bit field identifies the format of the Rx FIFO ID Filter Table Elements. Note that all elements of the table are configured at the same time by this field (they are all the same format). See Section "Rx FIFO Structure". This field can only be written in Freeze mode as it is blocked by hardware in other modes. 00 Format A: One full ID (standard and extended) per ID Filter Table element. 01 Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID Filter Table element. 10 Format C: Four partial 8-bit Standard IDs per ID Filter Table element. 11 Format D: All frames rejected.
7 Reserved	This read-only field is reserved and always has the value zero.
6–0 MAXMB	Number of the Last Message Buffer This 7-bit field defines the number of the last Message Buffers that will take part in the matching and arbitration processes. The reset value (0x0F) is equivalent to 16 MB configuration. This field can only be written in Freeze Mode as it is blocked by hardware in other modes. Number of the last MB = MAXMB **NOTE:** MAXMB must be programmed with a value smaller than the parameter NUMBER_OF_MB, otherwise the number of the last effective Message Buffer will be: (NUMBER_OF_MB - 1) Additionally, the value of MAXMB must encompass the FIFO size defined by CTRL2[RFFN]. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table "Minimum Ratio Between Peripheral Clock Frequency and CAN Bit Rate" (in Section "Arbitration and Matching Timing").

49.3.3 Control 1 Register (CANx_CTRL1)

This register is defined for specific FlexCAN control features related to the CAN bus, such as bit-rate, programmable sampling point within an Rx bit, Loop Back Mode, Listen-Only Mode, Bus Off recovery behavior and interrupt enabling (Bus-Off, Error, Warning). It also determines the Division Factor for the clock prescaler.

Addresses: CAN0_CTRL1 is 4002_4000h base + 4h offset = 4002_4004h
CAN1_CTRL1 is 400A_4000h base + 4h offset = 400A_4004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRESDIV								RJW		PSEG1			PSEG2		
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	BOFFMSK	ERRMSK	CLKSRC	LPB	TWRNMSK	RWRNMSK	0		SMP	BOFFREC	TSYN	LBUF	LOM	PROPSEG		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_CTRL1 field descriptions

Field	Description
31–24 PRESDIV	Prescaler Division Factor This 8-bit field defines the ratio between the PE clock frequency and the Serial Clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The Maximum value of this field is 0xFF, that gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Section "Protocol Timing". This field can only be written in Freeze mode as it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (PRESDIV + 1)
23–22 RJW	Resync Jump Width This 2-bit field defines the maximum number of time quanta that a bit time can be changed by one re-synchronization. (One time quantum is equal to the Sclock period.) The valid programmable values are 0–3. This field can only be written in Freeze mode as it is blocked by hardware in other modes. Resync Jump Width = RJW + 1.
21–19 PSEG1	Phase Segment 1 This 3-bit field defines the length of Phase Buffer Segment 1 in the bit time. The valid programmable values are 0–7. This field can only be written in Freeze mode as it is blocked by hardware in other modes. Phase Buffer Segment 1 = (PSEG1 + 1) x Time-Quanta.
18–16 PSEG2	Phase Segment 2

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
	This 3-bit field defines the length of Phase Buffer Segment 2 in the bit time. The valid programmable values are 1–7. This field can only be written in Freeze mode as it is blocked by hardware in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) x Time-Quanta.
15 BOFFMSK	Bus Off Mask This bit provides a mask for the Bus Off Interrupt. 0 Bus Off interrupt disabled 1 Bus Off interrupt enabled
14 ERRMSK	Error Mask This bit provides a mask for the Error Interrupt. 0 Error interrupt disabled 1 Error interrupt enabled
13 CLKSRC	CAN Engine Clock Source This bit selects the clock source to the CAN Protocol Engine (PE) to be either the peripheral clock (driven by the PLL) or the crystal oscillator clock. The selected clock is the one fed to the prescaler to generate the Serial Clock (Sclock). In order to guarantee reliable operation, this bit can only be written in Disable mode as it is blocked by hardware in other modes. See Section "Protocol Timing". 0 The CAN engine clock source is the oscillator clock. Under this condition, the oscillator clock frequency must be lower than the bus clock. 1 The CAN engine clock source is the peripheral clock.
12 LPB	Loop Back Mode This bit configures FlexCAN to operate in Loop-Back Mode. In this mode, FlexCAN performs an internal loop back that can be used for self test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic ‘1’). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field, generating an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. **NOTE:** In this mode, the MCR[SRXDIS] cannot be asserted because this will impede the self reception of a transmitted message. 0 Loop Back disabled 1 Loop Back enabled
11 TWRNMSK	Tx Warning Interrupt Mask This bit provides a mask for the Tx Warning Interrupt associated with the TWRNINT flag in the Error and Status Register. This bit is read as zero when MCR[WRNEN] bit is negated. This bit can only be written if MCR[WRNEN] bit is asserted. 0 Tx Warning Interrupt disabled 1 Tx Warning Interrupt enabled
10 RWRNMSK	Rx Warning Interrupt Mask

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
	This bit provides a mask for the Rx Warning Interrupt associated with the RWRNINT flag in the Error and Status Register. This bit is read as zero when MCR[WRNEN] bit is negated. This bit can only be written if MCR[WRNEN] bit is asserted. 0 Rx Warning Interrupt disabled 1 Rx Warning Interrupt enabled
9–8 Reserved	This read-only field is reserved and always has the value zero.
7 SMP	CAN Bit Sampling This bit defines the sampling mode of CAN bits at the Rx input. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Just one sample is used to determine the bit value. 1 Three samples are used to determine the value of the received bit: the regular one (sample point) and 2 preceding samples; a majority rule is used.
6 BOFFREC	Bus Off Recovery This bit defines how FlexCAN recovers from Bus Off state. If this bit is negated, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If the bit is asserted, automatic recovering from Bus Off is disabled and the module remains in Bus Off state until the bit is negated by the user. If the negation occurs before 128 sequences of 11 recessive bits are detected on the CAN bus, then Bus Off recovery happens as if the BOFFREC bit had never been asserted. If the negation occurs after 128 sequences of 11 recessive bits occurred, then FlexCAN will re-synchronize to the bus by waiting for 11 recessive bits before joining the bus. After negation, the BOFFREC bit can be re-asserted again during Bus Off, but it will only be effective the next time the module enters Bus Off. If BOFFREC was negated when the module entered Bus Off, asserting it during Bus Off will not be effective for the current Bus Off recovery. 0 Automatic recovering from Bus Off state enabled, according to CAN Spec 2.0 part B 1 Automatic recovering from Bus Off state disabled
5 TSYN	Timer Sync This bit enables a mechanism that resets the free-running timer each time a message is received in Message Buffer 0. This feature provides means to synchronize multiple FlexCAN stations with a special “SYNC” message (i.e., global network time). If the RFEN bit in MCR is set (Rx FIFO enabled), the first available Mailbox, according to CTRL2[RFFN] setting, is used for timer synchronization instead of MB0. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Timer Sync feature disabled 1 Timer Sync feature enabled
4 LBUF	Lowest Buffer Transmitted First This bit defines the ordering mechanism for Message Buffer transmission. When asserted, the LPRIOEN bit does not affect the priority arbitration. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Buffer with highest priority is transmitted first. 1 Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode

Table continues on the next page...

CANx_CTRL1 field descriptions (continued)

Field	Description
	This bit configures FlexCAN to operate in Listen-Only Mode. In this mode, transmission is disabled, all error counters are frozen and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error (without changing the REC), as if it was trying to acknowledge the message. Listen-Only Mode acknowledgement can be obtained by the state of ESR1[FLTCONF] field which is Passive Error when Listen-Only Mode is entered. There can be some delay between the Listen-Only Mode request and acknowledge. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Listen-Only Mode is deactivated. 1 FlexCAN module operates in Listen-Only Mode.
2–0 PROPSEG	Propagation Segment This 3-bit field defines the length of the Propagation Segment in the bit time. The valid programmable values are 0–7. This field can only be written in Freeze mode as it is blocked by hardware in other modes. Propagation Segment Time = (PROPSEG + 1) * Time-Quanta. Time-Quantum = one Sclock period.

49.3.4 Free Running Timer (CANx_TIMER)

This register represents a 16-bit free running counter that can be read and written by the CPU. The timer starts from 0x0 after Reset, counts linearly to 0xFFFF, and wraps around.

The timer is clocked by the FlexCAN bit-clock (which defines the baud rate on the CAN bus). During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable, Stop and Freeze Modes.

The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the Time Stamp entry in a message buffer after a successful reception or transmission of a message.

If bit CTRL1[TSYN] is asserted the Timer is reset whenever a message is received in the first available Mailbox, according to CTRL2[RFFN] setting.

The CPU can write to this register anytime. However, if the write occurs at the same time that the Timer is being reset by a reception in the first Mailbox, then the write value is discarded.

Reading this register affects the Mailbox Unlocking procedure; see Section "Message Buffer Lock Mechanism".

Addresses: CAN0_TIMER is 4002_4000h base + 8h offset = 4002_4008h
CAN1_TIMER is 400A_4000h base + 8h offset = 400A_4008h

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																TIMER															
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

CANx_TIMER field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 TIMER	Timer value Contains the free-running counter value.

49.3.5 Rx Mailboxes Global Mask Register (CANx_RXMGMASK)

This register is located in RAM.

RXMGMASK is provided for legacy support.

- When the MCR[IRMQ] bit is negated, RXMGMASK is always in effect.
- When the MCR[IRMQ] bit is asserted, RXMGMASK has no effect.

RXMGMASK is used to mask the filter fields of all Rx MBs, excluding MBs 14-15, which have individual mask registers.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Addresses: CAN0_RXMGMASK is 4002_4000h base + 10h offset = 4002_4010h
CAN1_RXMGMASK is 400A_4000h base + 10h offset = 400A_4010h

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	MG[31:0]																															
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

CANx_RXMGMASK field descriptions

Field	Description
31–0 MG[31:0]	Rx Mailboxes Global Mask Bits

CANx_RXMGMASK field descriptions (continued)

Field	Description
	These bits mask the Mailbox filter bits. Note that the alignment with the ID word of the Mailbox is not perfect as the two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the Mailbox. The following table shows in detail which MG bits mask each Mailbox filter field.

SMB[RTR][1]	CTRL2[RRS]	CTRL2[EACEN]	Mailbox filter fields			
			MB[RTR]	MB[IDE]	MB[ID]	Reserved
0	-	0	note[2]	note[3]	MG[28:0]	MG[31:29]
0	-	1	MG[31]	MG[30]	MG[28:0]	MG[29]
1	0	-	-	-	-	MG[31:0]
1	1	0	-	-	MG[28:0]	MG[31:29]
1	1	1	MG[31]	MG[30]	MG[28:0]	MG[29]

1. RTR bit of the Incoming Frame. It is saved into an auxiliary MB called Rx Serial Message Buffer (Rx SMB).
2. If the CTRL2[EACEN] bit is negated, the RTR bit of Mailbox is never compared with the RTR bit of the incoming frame.
3. If the CTRL2[EACEN] bit is negated, the IDE bit of Mailbox is always compared with the IDE bit of the incoming frame.

0 The corresponding bit in the filter is "don't care."
1 The corresponding bit in the filter is checked.

1. RTR bit of the Incoming Frame. It is saved into an auxiliary MB called Rx Serial Message Buffer (Rx SMB).
2. If the CTRL2[EACEN] bit is negated, the RTR bit of Mailbox is never compared with the RTR bit of the incoming frame.
3. If the CTRL2[EACEN] bit is negated, the IDE bit of Mailbox is always compared with the IDE bit of the incoming frame.

49.3.6 Rx 14 Mask Register (CANx_RX14MASK)

This register is located in RAM.

RX14MASK is provided for legacy support. When the MCR[IRMQ] bit is asserted, RX14MASK has no effect.

RX14MASK is used to mask the filter fields of Message Buffer 14.

This register can only be programmed while the module is in Freeze Mode as it is blocked by hardware in other modes.

Addresses: CAN0_RX14MASK is 4002_4000h base + 14h offset = 4002_4014h
CAN1_RX14MASK is 400A_4000h base + 14h offset = 400A_4014h

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	RX14M[31:0]																															
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

CANx_RX14MASK field descriptions

Field	Description
31–0 RX14M[31:0]	Rx Buffer 14 Mask Bits Each mask bit masks the corresponding Mailbox 14 filter field in the same way that RXMGMASK masks other Mailboxes' filters. See the description of the CAN_RXMGMASK register. 0 The corresponding bit in the filter is "don’t care." 1 The corresponding bit in the filter is checked.

49.3.7 Rx 15 Mask Register (CANx_RX15MASK)

This register is located in RAM.

RX15MASK is provided for legacy support. When the MCR[IRMQ] bit is asserted, RX15MASK has no effect.

RX15MASK is used to mask the filter fields of Message Buffer 15.

This register can only be programmed while the module is in Freeze Mode as it is blocked by hardware in other modes.

Addresses: CAN0_RX15MASK is 4002_4000h base + 18h offset = 4002_4018h
CAN1_RX15MASK is 400A_4000h base + 18h offset = 400A_4018h

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	RX15M[31:0]																															
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

CANx_RX15MASK field descriptions

Field	Description
31–0 RX15M[31:0]	Rx Buffer 15 Mask Bits Each mask bit masks the corresponding Mailbox 15 filter field in the same way that RXMGMASK masks other Mailboxes' filters. See the description of the CAN_RXMGMASK register. 0 The corresponding bit in the filter is "don’t care." 1 The corresponding bit in the filter is checked.

49.3.8 Error Counter (CANx_ECR)

This register has two 8-bit fields reflecting the value of two FlexCAN error counters: Transmit Error Counter (TXERRCNT field) and Receive Error Counter (RXERRCNT field). The rules for increasing and decreasing these counters are described in the CAN protocol and are completely implemented in the FlexCAN module. Both counters are read-only except in Freeze Mode, where they can be written by the CPU.

FlexCAN responds to any bus state as described in the protocol, e.g. transmit 'Error Active' or 'Error Passive' flag, delay its transmission start time ('Error Passive') and avoid any influence on the bus when in 'Bus Off' state. The following are the basic rules for FlexCAN bus state transitions.

- If the value of TXERRCNT or RXERRCNT increases to be greater than or equal to 128, the FLTCONF field in the Error and Status Register is updated to reflect 'Error Passive' state.
- If the FlexCAN state is 'Error Passive', and either TXERRCNT or RXERRCNT decrements to a value less than or equal to 127 while the other already satisfies this condition, the FLTCONF field in the Error and Status Register is updated to reflect 'Error Active' state.
- If the value of TXERRCNT increases to be greater than 255, the FLTCONF field in the Error and Status Register is updated to reflect 'Bus Off' state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.
- If FlexCAN is in 'Bus Off' state, then TXERRCNT is cascaded together with another internal counter to count the 128th occurrences of 11 consecutive recessive bits on the bus. Hence, TXERRCNT is reset to zero and counts in a manner where the internal counter counts 11 such bits and then wraps around while incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, the FLTCONF field in the Error and Status Register is updated to be 'Error Active' and both error counters are reset to zero. At any instance of dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value.
- If during system start-up, only one node is operating, then its TXERRCNT increases in each message it is trying to transmit, as a result of acknowledge errors (indicated by the ACKERR bit in the Error and Status Register). After the transition to 'Error Passive' state, the TXERRCNT does not increment anymore by acknowledge errors. Therefore the device never goes to the 'Bus Off' state.
- If the RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected while being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to resume to 'Error Active' state.

Addresses: CAN0_ECR is 4002_4000h base + 1Ch offset = 4002_401Ch
CAN1_ECR is 400A_4000h base + 1Ch offset = 400A_401Ch

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																RXERRCNT								TXERRCNT							
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

CANx_ECR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–8 RXERRCNT	Receive Error Counter
7–0 TXERRCNT	Transmit Error Counter

49.3.9 Error and Status 1 Register (CANx_ESR1)

This register reflects various error conditions, some general status of the device and it is the source of interrupts to the CPU.

The CPU read action clears bits 15-10, therefore the reported error conditions (bits 15-10) are those that occurred since the last time the CPU read this register. Bits 9-3 are status bits.

The following table shows the FlexCAN state variables and their meanings. Other combinations not shown in the table are reserved.

SYNCH	IDLE	TX	RX	FlexCAN State
0	0	0	0	Not synchronized to CAN bus
1	1	x	x	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

Addresses: CAN0_ESR1 is 4002_4000h base + 20h offset = 4002_4020h
CAN1_ESR1 is 400A_4000h base + 20h offset = 400A_4020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													SYNCH	TWRNINT	RWRNINT
W															w1c	w1c
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	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	WAKINT
W														w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_ESR1 field descriptions

Field	Description
31–19 Reserved	This read-only field is reserved and always has the value zero.
18 SYNCH	CAN Synchronization Status This read-only flag indicates whether the FlexCAN is synchronized to the CAN bus and able to participate in the communication process. It is set and cleared by the FlexCAN. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not synchronized to the CAN bus. 1 FlexCAN is synchronized to the CAN bus.
17 TWRNINT	Tx Warning Interrupt Flag If the WRNEN bit in MCR is asserted, the TWRNINT bit is set when the TXWRN flag transitions from ‘0’ to ‘1’, meaning that the Tx error counter reached 96. If the corresponding mask bit in the Control Register (TWRNMSK) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to ‘1’. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit. Otherwise it will be set when the WRNEN is set again. Writing ‘0’ has no effect. This flag is not generated during “Bus Off” state. This bit is not updated during Freeze mode. 0 No such occurrence 1 The Tx error counter transitioned from less than 96 to greater than or equal to 96.
16 RWRNINT	Rx Warning Interrupt Flag If the WRNEN bit in MCR is asserted, the RWRNINT bit is set when the RXWRN flag transitions from ‘0’ to ‘1’, meaning that the Rx error counters reached 96. If the corresponding mask bit in the Control Register (RWRNMSK) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to ‘1’. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
	Otherwise it will be set when the WRNEN is set again. Writing ‘0’ has no effect. This bit is not updated during Freeze mode. 0 No such occurrence 1 The Rx error counter transitioned from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a message. **NOTE:** This bit is not set by a transmitter in case of arbitration field or ACK slot, or in case of a node sending a passive error flag that detects dominant bits. 0 No such occurrence 1 At least one bit sent as recessive is received as dominant.
14 BIT0ERR	Bit0 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a message. 0 No such occurrence 1 At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error This bit indicates that an Acknowledge Error has been detected by the transmitter node, i.e., a dominant bit has not been detected during the ACK SLOT. 0 No such occurrence 1 An ACK error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error This bit indicates that a CRC Error has been detected by the receiver node, i.e., the calculated CRC is different from the received. 0 No such occurrence 1 A CRC error occurred since last read of this register.
11 FRMERR	Form Error This bit indicates that a Form Error has been detected by the receiver node, i.e., a fixed-form bit field contains at least one illegal bit. 0 No such occurrence 1 A Form Error occurred since last read of this register.
10 STFERR	Stuffing Error This bit indicates that a Stuffing Error has been detected. 0 No such occurrence 1 A Stuffing Error occurred since last read of this register.
9 TXWRN	TX Error Warning

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
	This bit indicates when repetitive errors are occurring during message transmission. This bit is not updated during Freeze mode. 0 No such occurrence 1 TXERRCNT is greater than or equal to 96.
8 RXWRN	Rx Error Warning This bit indicates when repetitive errors are occurring during message reception. This bit is not updated during Freeze mode. 0 No such occurrence 1 RXERRCNT is greater than or equal to 96.
7 IDLE	This bit indicates when CAN bus is in IDLE state. See the table in the overall CAN_ESR1 register description. 0 No such occurrence 1 CAN bus is now IDLE.
6 TX	FlexCAN in Transmission This bit indicates if FlexCAN is transmitting a message. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not transmitting a message. 1 FlexCAN is transmitting a message.
5–4 FLTCONF	Fault Confinement State This 2-bit field indicates the Confinement State of the FlexCAN module. If the LOM bit in the Control Register is asserted, after some delay that depends on the CAN bit timing the FLTCONF field will indicate “Error Passive”. The very same delay affects the way how FLTCONF reflects an update to ECR register by the CPU. It may be necessary up to one CAN bit time to get them coherent again. Since the Control Register is not affected by soft reset, the FLTCONF field will not be affected by soft reset if the LOM bit is asserted. 00 Error Active 01 Error Passive 1x Bus Off
3 RX	FlexCAN in Reception This bit indicates if FlexCAN is receiving a message. See the table in the overall CAN_ESR1 register description. 0 FlexCAN is not receiving a message. 1 FlexCAN is receiving a message.
2 BOFFINT	‘Bus Off’ Interrupt This bit is set when FlexCAN enters ‘Bus Off’ state. If the corresponding mask bit in the Control Register (BOFFMSK) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to ‘1’. Writing ‘0’ has no effect.

Table continues on the next page...

CANx_ESR1 field descriptions (continued)

Field	Description
	0 No such occurrence 1 FlexCAN module entered ‘Bus Off’ state
1 ERRINT	Error Interrupt This bit indicates that at least one of the Error Bits (bits 15-10) is set. If the corresponding mask bit CTRL1[ERRMSK] is set, an interrupt is generated to the CPU. This bit is cleared by writing it to ‘1’. Writing ‘0’ has no effect. 0 No such occurrence 1 Indicates setting of any Error Bit in the Error and Status Register
0 WAKINT	Wake-Up Interrupt This field applies when FlexCAN is in low power mode: • Stop Mode When a recessive-to-dominant transition is detected on the CAN bus and if the MCR[WAKMSK] bit is set, an interrupt is generated to the CPU. This bit is cleared by writing it to ‘1’. When MCR[SLFWAK] is negated, this flag is masked. The CPU must clear this flag before disabling the bit. Otherwise it will be set when the SLFWAK is set again. Writing ‘0’ has no effect. 0 No such occurrence 1 Indicates a recessive to dominant transition was received on the CAN bus

49.3.10 Interrupt Masks 2 Register (CANx_IMASK2)

This register allows any number of a range of 32 Message Buffer Interrupts to be enabled or disabled. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception (i.e. when the corresponding IFLAG2 bit is set).

Addresses: CAN0_IMASK2 is 4002_4000h base + 24h offset = 4002_4024h
CAN1_IMASK2 is 400A_4000h base + 24h offset = 400A_4024h

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	BUFHM																															
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

CANx_IMASK2 field descriptions

Field	Description
31–0 BUFHM	Buffer MB_i Mask Each bit enables or disables the corresponding FlexCAN Message Buffer Interrupt.

CANx_IMASK2 field descriptions (continued)

Field	Description
	NOTE: Setting or clearing a bit in the IMASK2 Register can assert or negate an interrupt request, if the corresponding IFLAG2 bit is set. 0 The corresponding buffer Interrupt is disabled. 1 The corresponding buffer Interrupt is enabled.

49.3.11 Interrupt Masks 1 Register (CANx_IMASK1)

This register allows any number of a range of 32 Message Buffer Interrupts to be enabled or disabled. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception (i.e. when the corresponding IFLAG1 bit is set).

Addresses: CAN0_IMASK1 is 4002_4000h base + 28h offset = 4002_4028h
CAN1_IMASK1 is 400A_4000h base + 28h offset = 400A_4028h

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	BUFLM																															
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

CANx_IMASK1 field descriptions

Field	Description
31–0 BUFLM	Buffer MB$_i$ Mask Each bit enables or disables the corresponding FlexCAN Message Buffer Interrupt. **NOTE:** Setting or clearing a bit in the IMASK1 Register can assert or negate an interrupt request, if the corresponding IFLAG1 bit is set. 0 The corresponding buffer Interrupt is disabled. 1 The corresponding buffer Interrupt is enabled.

49.3.12 Interrupt Flags 2 Register (CANx_IFLAG2)

This register defines the flags for 32 Message Buffer interrupts. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG2 bit. If the corresponding IMASK2 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB] field, CPU must service the IFLAG2 bits whose MB value is greater than the MCR[MAXMB] to be updated; otherwise, they will remain set and be inconsistent with the amount of MBs available.

Addresses: CAN0_IFLAG2 is 4002_4000h base + 2Ch offset = 4002_402Ch
CAN1_IFLAG2 is 400A_4000h base + 2Ch offset = 400A_402Ch

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	BUFHI																															
W	w1c																															
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

CANx_IFLAG2 field descriptions

Field	Description
31–0 BUFHI	Buffer MB$_i$ Interrupt Each bit flags the corresponding FlexCAN Message Buffer interrupt. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception. 1 The corresponding buffer has successfully completed transmission or reception.

49.3.13 Interrupt Flags 1 Register (CANx_IFLAG1)

This register defines the flags for 32 Message Buffer interrupts. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG1 bit. If the corresponding IMASK1 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

The BUF7I to BUF5I flags are also used to represent FIFO interrupts when the Rx FIFO is enabled. When the bit MCR[RFEN] is set the function of the 8 least significant interrupt flags BUF[7:0]I changes: BUF7I, BUF6I and BUF5I indicate operating conditions of the FIFO, and BUF4TO0I are reserved.

Before enabling the RFEN, the CPU must service the IFLAG bits asserted in the Rx FIFO region; see Section "Rx FIFO". Otherwise, these IFLAG bits will mistakenly show the related MBs now belonging to FIFO as having contents to be serviced. When the RFEN bit is negated, the FIFO flags must be cleared. The same care must be taken when an RFFN value is selected extending Rx FIFO filters beyond MB7. For example, when RFFN is 0x8, the MB0-23 range is occupied by Rx FIFO filters and related IFLAG bits must be cleared.

Before updating MCR[MAXMB] field, CPU must service the IFLAG1 bits whose MB value is greater than the MCR[MAXMB] to be updated; otherwise, they will remain set and be inconsistent with the amount of MBs available.

Addresses: CAN0_IFLAG1 is 4002_4000h base + 30h offset = 4002_4030h
CAN1_IFLAG1 is 400A_4000h base + 30h offset = 400A_4030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BUF31TO8I[bit 8]															
W	w1c															
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	BUF31TO8I[7:0]								BUF7I	BUF6I	BUF5I	BUF4TO0I				
W	w1c								w1c	w1c	w1c	w1c				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_IFLAG1 field descriptions

Field	Description
31–8 BUF31TO8I	Buffer MB_i Interrupt Each bit flags the corresponding FlexCAN Message Buffer interrupt. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception. 1 The corresponding buffer has successfully completed transmission or reception.
7 BUF7I	Buffer MB7 Interrupt or "Rx FIFO Overflow" When the RFEN bit in the MCR is cleared (Rx FIFO disabled), this bit flags the interrupt for MB7. **NOTE:** This flag is cleared by the FlexCAN whenever the bit MCR[RFEN] is changed by CPU writes. The BUF7I flag represents "Rx FIFO Overflow" when MCR[RFEN] is set. In this case, the flag indicates that a message was lost because the Rx FIFO is full. Note that the flag will not be asserted when the Rx FIFO is full and the message was captured by a Mailbox. 0 No occurrence of MB7 completing transmission/reception (when MCR[RFEN]=0) or of Rx FIFO overflow (when MCR[RFEN]=1) 1 MB7 completed transmission/reception (when MCR[RFEN]=0) or Rx FIFO overflow (when MCR[RFEN]=1)
6 BUF6I	Buffer MB6 Interrupt or "Rx FIFO Warning" When the RFEN bit in the MCR is cleared (Rx FIFO disabled), this bit flags the interrupt for MB6. **NOTE:** This flag is cleared by the FlexCAN whenever the bit MCR[RFEN] is changed by CPU writes. The BUF6I flag represents "Rx FIFO Warning" when MCR[RFEN] is set. In this case, the flag indicates when the number of unread messages within the Rx FIFO is increased to 5 from 4 due to the reception of a new one, meaning that the Rx FIFO is almost full. Note that if the flag is cleared while the number of unread messages is greater than 4, it does not assert again until the number of unread messages within the Rx FIFO is decreased to be equal to or less than 4.

Table continues on the next page...

CANx_IFLAG1 field descriptions (continued)

Field	Description
	0 No occurrence of MB6 completing transmission/reception (when MCR[RFEN]=0) or of Rx FIFO almost full (when MCR[RFEN]=1) 1 MB6 completed transmission/reception (when MCR[RFEN]=0) or Rx FIFO almost full (when MCR[RFEN]=1)
5 BUF5I	Buffer MB5 Interrupt or "Frames available in Rx FIFO" When the RFEN bit in the MCR is cleared (Rx FIFO disabled), this bit flags the interrupt for MB5. **NOTE:** This flag is cleared by the FlexCAN whenever the bit MCR[RFEN] is changed by CPU writes. The BUF5I flag represents "Frames available in Rx FIFO" when MCR[RFEN] is set. In this case, the flag indicates that at least one frame is available to be read from the Rx FIFO. 0 No occurrence of MB5 completing transmission/reception (when MCR[RFEN]=0) or of frame(s) available in the Rx FIFO (when MCR[RFEN]=1) 1 MB5 completed transmission/reception (when MCR[RFEN]=0) or frame(s) available in the Rx FIFO (when MCR[RFEN]=1)
4–0 BUF4TO0I	Buffer MB$_i$ Interrupt or "reserved" When the RFEN bit in the MCR is cleared (Rx FIFO disabled), these bits flag the interrupts for MB4 to MB0. **NOTE:** These flags are cleared by the FlexCAN whenever the bit MCR[RFEN] is changed by CPU writes. The BUF4TO0I flags are reserved when MCR[RFEN] is set. 0 The corresponding buffer has no occurrence of successfully completed transmission or reception (when MCR[RFEN]=0). 1 The corresponding buffer has successfully completed transmission or reception (when MCR[RFEN]=0).

49.3.14 Control 2 Register (CANx_CTRL2)

This register contains control bits for CAN errors, FIFO features, and mode selection.

Addresses: CAN0_CTRL2 is 4002_4000h base + 34h offset = 4002_4034h
CAN1_CTRL2 is 400A_4000h base + 34h offset = 400A_4034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R		0		WRMFRZ	RFFN				TASD					MRP	RRS	EACEN
W	0															
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0

Bit	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

CANx_CTRL2 field descriptions

Field	Description
31 Reserved	This field is reserved.
30–29 Reserved	This read-only field is reserved and always has the value zero.
28 WRMFRZ	Write-Access to Memory in Freeze mode Enable unrestricted write access to FlexCAN memory in Freeze mode. This bit can only be written in Freeze mode and has no effect out of Freeze mode. 0 Maintain the write access restrictions. 1 Enable unrestricted write access to FlexCAN memory.
27–24 RFFN	Number of Rx FIFO Filters This 4-bit field defines the number of Rx FIFO filters, as shown in the following table. The maximum selectable number of filters is determined by the MCU. This field can only be written in Freeze mode as it is blocked by hardware in other modes. This field must not be programmed with values that make the number of Message Buffers occupied by Rx FIFO and ID Filter exceed the number of Mailboxes present, defined by MCR[MAXMB]. **NOTE:** Each group of eight filters occupies a memory space equivalent to two Message Buffers which means that the more filters are implemented the less Mailboxes will be available. Considering that the Rx FIFO occupies the memory space originally reserved for MB0-5, RFFN should be programmed with a value correponding to a number of filters not greater than the number of available memory words which can be calculated as follows: (SETUP_MB - 6) x 4 where SETUP_MB is the least between NUMBER_OF_MB and MAXMB.

Table continues on the next page...

CANx_CTRL2 field descriptions (continued)

Field	Description
	The number of remaining Mailboxes available will be: (SETUP_MB - 8) - (RFFN x 2) If the Number of Rx FIFO Filters programmed through RFFN exceeds the SETUP_MB value (memory space available) the exceeding ones will not be functional.

RFFN[3:0]	Number of Rx FIFO filters	Message Buffers occupied by Rx FIFO and ID Filter Table	Remaining Available Mailboxes[1]	Rx FIFO ID Filter Table Elements Affected by Rx Individual Masks[2]	Rx FIFO ID Filter Table Elements Affected by Rx FIFO Global Mask[2]
0x0	8	MB 0-7	MB 8-63	Elements 0-7	none
0x1	16	MB 0-9	MB 10-63	Elements 0-9	Elements 10-15
0x2	24	MB 0-11	MB 12-63	Elements 0-11	Elements 12-23
0x3	32	MB 0-13	MB 14-63	Elements 0-13	Elements 14-31
0x4	40	MB 0-15	MB 16-63	Elements 0-15	Elements 16-39
0x5	48	MB 0-17	MB 18-63	Elements 0-17	Elements 18-47
0x6	56	MB 0-19	MB 20-63	Elements 0-19	Elements 20-55
0x7	64	MB 0-21	MB 22-63	Elements 0-21	Elements 22-63
0x8	72	MB 0-23	MB 24-63	Elements 0-23	Elements 24-71
0x9	80	MB 0-25	MB 26-63	Elements 0-25	Elements 26-79
0xA	88	MB 0-27	MB 28-63	Elements 0-27	Elements 28-87
0xB	96	MB 0-29	MB 30-63	Elements 0-29	Elements 30-95
0xC	104	MB 0-31	MB 32-63	Elements 0-31	Elements 32-103
0xD	112	MB 0-33	MB 34-63	Elements 0-31	Elements 32-111
0xE	120	MB 0-35	MB 36-63	Elements 0-31	Elements 32-119
0xF	128	MB 0-37	MB 38-63	Elements 0-31	Elements 32-127

1. The number of the last remaining available mailboxes is defined by the least value between the parameter NUMBER_OF_MB minus 1 and the MCR[MAXMB] field.
2. If Rx Individual Mask Registers are not enabled then all Rx FIFO filters are affected by the Rx FIFO Global Mask.

Field	Description
23–19 TASD	Tx Arbitration Start Delay This 5-bit field indicates how many CAN bits the Tx arbitration process start point can be delayed from the first bit of CRC field on CAN bus. This field can only be written in Freeze mode as it is blocked by hardware in other modes. This field is useful to optimize the transmit performance based on factors such as: peripheral/serial clock ratio, CAN bit timing and number of MBs. The duration of an arbitration process, in terms of CAN bits, is directly proportional to the number of available MBs and CAN baud rate and inversely proportional to the peripheral clock frequency. The optimal arbitration timing is that in which the last MB is scanned right before the first bit of the Intermission field of a CAN frame. Therefore, if there are few MBs and the system/serial clock ratio is high and the CAN baud rate is low then the arbitration can be delayed and vice-versa.

Table continues on the next page...

CANx_CTRL2 field descriptions (continued)

Field	Description
	If TASD is 0 then the arbitration start is not delayed, thus the CPU has less time to configure a Tx MB for the next arbitration, but more time is reserved for arbitration. In the other hand, if TASD is 24 then the CPU can configure a Tx MB later and less time is reserved for arbitration. If too little time is reserved for arbitration the FlexCAN may be not able to find winner MBs in time to compete with other nodes for the CAN bus. If the arbitration ends too much time before the first bit of Intermission field then there is a chance that the CPU reconfigures some Tx MBs and the winner MB is not the best to be transmitted. The optimal configuration for TASD can be calculated as: $TASD = 25 - \{f_{CANCLK} \times [MAXB + 3 - (RFEN \times 8) - (RFEN \times RFFN \times 2)] \times 2\} / \{f_{SYS} \times [1+(PSEG1+1)+(PSEG2+1)+(PROPSEG+1)] \times (PRESDIV+1)\}$ where: • f_{CANCLK} is the Protocol Engine (PE) Clock (see section "Protocol Timing"), in Hz; • f_{SYS} is the peripheral clock, in Hz; • MAXMB is the value in CTRL1[MAXMB] field; • RFEN is the value in CTRL1[RFEN] bit; • RFFN is the value in CTRL2[RFFN] field; • PSEG1 is the value in CTRL1[PSEG1] field; • PSEG2 is the value in CTRL1[PSEG2] field; • PROPSEG is the value in CTRL1[PROPSEG] field; • PRESDIV is the value in CTRL1[PRESDIV] field. See Section "Arbitration process" and Section "Protocol Timing" for more details. **NOTE:** The recommended value for TASD is 22.
18 MRP	Mailboxes Reception Priority If this bit is set the matching process starts from the Mailboxes and if no match occurs the matching continues on the Rx FIFO. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Matching starts from Rx FIFO and continues on Mailboxes. 1 Matching starts from Mailboxes and continues on Rx FIFO.
17 RRS	Remote Request Storing If this bit is asserted Remote Request Frame is submitted to a matching process and stored in the corresponding Message Buffer in the same fashion of a Data Frame. No automatic Remote Response Frame will be generated. If this bit is negated the Remote Request Frame is submitted to a matching process and an automatic Remote Response Frame is generated if a Message Buffer with CODE=0b1010 is found with the same ID. This bit can only be written in Freeze mode as it is blocked by hardware in other modes. 0 Remote Response Frame is generated. 1 Remote Request Frame is stored.
16 EACEN	Entire Frame Arbitration Field Comparison Enable for Rx Mailboxes This bit controls the comparison of IDE and RTR bits whithin Rx Mailboxes filters with their corresponding bits in the incoming frame by the matching process. This bit does not affect matching for Rx FIFO. This bit can only be written in Freeze mode as it is blocked by hardware in other modes.

Table continues on the next page...

CANx_CTRL2 field descriptions (continued)

Field	Description
	0 Rx Mailbox filter's IDE bit is always compared and RTR is never compared despite mask bits. 1 Enables the comparison of both Rx Mailbox filter's IDE and RTR bit with their corresponding bits within the incoming frame. Mask bits do apply.
15–0 Reserved	This read-only field is reserved and always has the value zero.

1. The number of the last remaining available mailboxes is defined by the least value between the parameter NUMBER_OF_MB minus 1 and the MCR[MAXMB] field.
2. If Rx Individual Mask Registers are not enabled then all Rx FIFO filters are affected by the Rx FIFO Global Mask.

49.3.15 Error and Status 2 Register (CANx_ESR2)

This register reflects various interrupt flags and some general status.

Addresses: CAN0_ESR2 is 4002_4000h base + 38h offset = 4002_4038h

CAN1_ESR2 is 400A_4000h base + 38h offset = 400A_4038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LPTM						
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	VPS	IMB	0												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CANx_ESR2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value zero.
22–16 LPTM	Lowest Priority Tx Mailbox If ESR2[VPS] is asserted, this field indicates the lowest number inactive Mailbox (refer to the IMB bit description). If there is no inactive Mailbox then the Mailbox indicated depends on CTRL1[LBUF] bit value. If CTRL1[LBUF] bit is negated then the Mailbox indicated is the one which has the greatest arbitration value (see the "Highest priority Mailbox first" section). If CTRL1[LBUF] bit is asserted then the Mailbox indicated is the highest number active Tx Mailbox. If a Tx Mailbox is being transmitted it is not considered in LPTM calculation. If ESR2[IMB] is not asserted and a frame is transmitted successfully, LPTM is updated with its Mailbox number.
15 Reserved	This read-only field is reserved and always has the value zero.
14 VPS	Valid Priority Status This bit indicates whether IMB and LPTM contents are currently valid or not. VPS is asserted upon every complete Tx arbitration process unless the CPU writes to Control and Status word of a Mailbox that has

Table continues on the next page...

CANx_ESR2 field descriptions (continued)

Field	Description
	already been scanned (i.e. it is behind Tx Arbitration Pointer) during the Tx arbitration process. If there is no inactive Mailbox and only one Tx Mailbox which is being transmitted then VPS is not asserted. VPS is negated upon the start of every Tx arbitration process or upon a write to Control and Status word of any Mailbox. **NOTE:** ESR2[VPS] is not affected by any CPU write into Control Status (C/S) of a MB which is blocked by abort mechanism. When MCR[AEN] is asserted, the abort code write in C/S of a MB that is being transmitted (pending abort), or any write attempt into a Tx MB with IFLAG set is blocked. 0 Contents of IMB and LPTM are invalid. 1 Contents of IMB and LPTM are valid.
13 IMB	Inactive Mailbox If ESR2[VPS] is asserted, this bit indicates whether there is any inactive Mailbox (CODE field is either 0b1000 or 0b0000). This bit is asserted in the following cases: • During arbitration, if an LPTM is found and it is inactive. • If IMB is not asserted and a frame is transmitted successfully. This bit is cleared in all start of arbitration (see Section "Arbitration process"). **NOTE:** LPTM mechanism have the following behavior: if an MB is successfully transmitted and ESR2[IMB]=0 (no inactive Mailbox), then ESR2[VPS] and ESR2[IMB] are asserted and the index related to the MB just transmitted is loaded into ESR2[LPTM]. 0 If ESR2[VPS] is asserted, the ESR2[LPTM] is not an inactive Mailbox. 1 If ESR2[VPS] is asserted, there is at least one inactive Mailbox. LPTM content is the number of the first one.
12–0 Reserved	This read-only field is reserved and always has the value zero.

49.3.16 CRC Register (CANx_CRCR)

This register provides information about the CRC of transmitted messages.

Addresses: CAN0_CRCR is 4002_4000h base + 44h offset = 4002_4044h
CAN1_CRCR is 400A_4000h base + 44h offset = 400A_4044h

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									MBCRC							0	TXCRC														
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

CANx_CRCR field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

CANx_CRCR field descriptions (continued)

Field	Description
22–16 MBCRC	CRC Mailbox This field indicates the number of the Mailbox corresponding to the value in TXCRC field.
15 Reserved	This read-only field is reserved and always has the value zero.
14–0 TXCRC	CRC Transmitted This field indicates the CRC value of the last message transmitted. This field is updated at the same time the Tx Interrupt Flag is asserted.

49.3.17 Rx FIFO Global Mask Register (CANx_RXFGMASK)

This register is located in RAM.

If Rx FIFO is enabled RXFGMASK is used to mask the Rx FIFO ID Filter Table elements that do not have a corresponding RXIMR according to CTRL2[RFFN] field setting.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Addresses: CAN0_RXFGMASK is 4002_4000h base + 48h offset = 4002_4048h
CAN1_RXFGMASK is 400A_4000h base + 48h offset = 400A_4048h

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	FGM[31:0]																															
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

CANx_RXFGMASK field descriptions

Field	Description
31–0 FGM[31:0]	Rx FIFO Global Mask Bits These bits mask the ID Filter Table elements bits in a perfect alignment. The following table shows how the FGM bits correspond to each IDAF field.

CANx_RXFGMASK field descriptions (continued)

Field	Description
	(see below)

Rx FIFO ID Filter Table Elements Format (MCR[IDAM])	Identifier Acceptance Filter Fields					
	RTR	IDE	RXIDA	RXIDB[1]	RXIDC[2]	Reserved
A	FGM[31]	FGM[30]	FGM[29:1]	-	-	FGM[0]
B	FGM[31], FGM[15]	FGM[30], FGM[14]	-	FGM[29:16], FGM[13:0]		-
C	-	-		-	FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]	

1. If MCR[IDAM] field is equivalent to the format B only the fourteen most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.
2. If MCR[IDAM] field is equivalent to the format C only the eight most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.

0 The corresponding bit in the filter is "don't care."

1 The corresponding bit in the filter is checked.

[1] If MCR[IDAM] field is equivalent to the format B only the fourteen most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.

[2] If MCR[IDAM] field is equivalent to the format C only the eight most significant bits of the Identifier of the incoming frame are compared with the Rx FIFO filter.

49.3.18 Rx FIFO Information Register (CANx_RXFIR)

RXFIR provides information on Rx FIFO.

This register is the port through which the CPU accesses the output of the RXFIR FIFO located in RAM. The RXFIR FIFO is written by the FlexCAN whenever a new message is moved into the Rx FIFO as well as its output is updated whenever the output of the Rx FIFO is updated with the next message. See Section "Rx FIFO" for instructions on reading this register.

Addresses: CAN0_RXFIR is 4002_4000h base + 4Ch offset = 4002_404Ch

CAN1_RXFIR is 400A_4000h base + 4Ch offset = 400A_404Ch

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																							IDHIT								
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.

CANx_RXFIR field descriptions

Field	Description
31–9 Reserved	This read-only field is reserved and always has the value zero.
8–0 IDHIT	Identifier Acceptance Filter Hit Indicator This field indicates which Identifier Acceptance Filter was hit by the received message that is in the output of the Rx FIFO. If multiple filters match the incoming message ID then the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only while the IFLAG[BUF5I] is asserted.

49.3.19 Rx Individual Mask Registers (CANx_RXIMR)

These registers are located in RAM.

RXIMR are used as acceptance masks for ID filtering in Rx MBs and the Rx FIFO. If the Rx FIFO is not enabled, one mask register is provided for each available Mailbox, providing ID masking capability on a per Mailbox basis.

When the Rx FIFO is enabled (MCR[RFEN] bit is asserted), up to 32 Rx Individual Mask Registers can apply to the Rx FIFO ID Filter Table elements on a one-to-one correspondence depending on the setting of CTRL2[RFFN].

RXIMR can only be written by the CPU while the module is in Freeze Mode; otherwise, they are blocked by hardware.

The Individual Rx Mask Registers are not affected by reset and must be explicitly initialized prior to any reception.

Addresses: 4002_4000h base + 880h offset + (4d × *n*), where *n* = 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	MI[31:0]																															
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.

CANx_RXIMR*n* field descriptions

Field	Description
31–0 MI[31:0]	Individual Mask Bits Each Individual Mask Bit masks the corresponding bit in both the Mailbox filter and Rx FIFO ID Filter Table element in distinct ways. For Mailbox filters, see the RXMGMASK register description.

CANx_RXIMR*n* field descriptions (continued)

Field	Description
	For Rx FIFO ID Filter Table elements, see the RXFGMASK register description. 0 The corresponding bit in the filter is "don't care." 1 The corresponding bit in the filter is checked.

49.3.56 Message Buffer Structure

The Message Buffer structure used by the FlexCAN module is represented in the following figure. Both Extended and Standard Frames (29-bit Identifier and 11-bit Identifier, respectively) used in the CAN specification (Version 2.0 Part B) are represented. Each individual MB is formed by 16 bytes.

The memory area from 0x80 to 0x47C is used by the Mailboxes.

Table 49-108. Message Buffer Structure

<table>
<tr><th></th><th>31</th><th>30</th><th>29</th><th>28</th><th>27 24</th><th>23</th><th>22</th><th>21</th><th>20</th><th>19</th><th>18</th><th>17</th><th>16</th><th>15 8</th><th>7 0</th></tr>
<tr><td>0x0</td><td colspan="4">▨</td><td>CODE</td><td>▨</td><td>SRR</td><td>IDE</td><td>RTR</td><td colspan="4">DLC</td><td colspan="2">TIME STAMP</td></tr>
<tr><td>0x4</td><td colspan="3">PRIO</td><td colspan="8">ID (Standard/Extended)</td><td colspan="4">ID (Extended)</td></tr>
<tr><td>0x8</td><td colspan="5">Data Byte 0</td><td colspan="8">Data Byte 1</td><td>Data Byte 2</td><td>Data Byte 3</td></tr>
<tr><td>0xC</td><td colspan="5">Data Byte 4</td><td colspan="8">Data Byte 5</td><td>Data Byte 6</td><td>Data Byte 7</td></tr>
</table>

▨ = Unimplemented or Reserved

CODE — Message Buffer Code

This 4-bit field can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 49-109 and Table 49-110. See Functional Description for additional information.

Table 49-109. Message Buffer Code for Rx buffers

CODE Description	Rx Code BEFORE receive New Frame	SRV[1]	Rx Code AFTER successful reception[2]	RRS[3]	Comment
0b0000: INACTIVE- MB is not active.	INACTIVE	-	-	-	MB does not participate in the matching process.

Table continues on the next page...

Table 49-109. Message Buffer Code for Rx buffers (continued)

CODE Description	Rx Code BEFORE receive New Frame	SRV[1]	Rx Code AFTER successful reception[2]	RRS[3]	Comment
0b0100: EMPTY - MB is active and empty.	EMPTY	-	FULL	-	When a frame is received successfully (after move-in process. Refer to Section "Move-in" for details), the CODE field is automatically updated to FULL.
0b0010: FULL - MB is full.	FULL	Yes	FULL	-	The act of reading the C/S word followed by unlocking the MB (SRV) does not make the code return to EMPTY. It remains FULL. If a new frame is moved to the MB after the MB was serviced, the code still remains FULL. Refer to Section "Matching Process" for matching details related to FULL code.
		No	OVERRUN	-	If the MB is FULL and a new frame is moved to this MB before the CPU service it, the CODE field is automatically updated to OVERRUN. Refer to Section "Matching Process" for details about overrun behavior.

Table continues on the next page...

Table 49-109. Message Buffer Code for Rx buffers (continued)

CODE Description	Rx Code BEFORE receive New Frame	SRV[1]	Rx Code AFTER successful reception[2]	RRS[3]	Comment
0b0110: OVERRUN - MB is being overwritten into a full buffer.	OVERRUN	Yes	FULL	-	If the CODE field indicates OVERRUN and CPU has serviced the MB, when a new frame is moved to the MB, the code returns to FULL.
		No	OVERRUN	-	If the CODE field already indicates OVERRUN, and another new frame must be moved, the MB will be overwritten again, and the code will remain OVERRUN. Refer to Section "Matching Process" for details about overrun behavior.

Table continues on the next page...

Table 49-109. Message Buffer Code for Rx buffers (continued)

CODE Description	Rx Code BEFORE receive New Frame	SRV[1]	Rx Code AFTER successful reception[2]	RRS[3]	Comment
0b1010: RANSWER[4] - A frame was configured to recognize a Remote Request Frame and transmit a Response Frame in return.	RANSWER	-	TANSWER(0b1110)	0	A Remote Answer was configured to recognize a remote request frame received, after that a MB is set to transmit a response frame. The code is automatically changed to TANSWER (0b1110). Refer to Section "Matching Process" for details. If CTRL2[RRS] is negated, transmit a response frame whenever a remote request frame with the same ID is received.
		-	-	1	This code is ignored during matching and arbitration process. Refer to Section "Matching Process for details.
CODE[0]=1b1: BUSY - FlexCAN is updating the contents of the MB. The CPU must not access the MB.	BUSY[5]	-	FULL	-	Indicates that the MB is being updated, it will be negated automatically and does not interfere on the next CODE.
		-	OVERRUN	-	

1. SRV: Serviced MB. MB was read and unlocked by reading TIMER or other MB.
2. A frame is considered successful reception after the frame to be moved to MB (move-in process). Refer to Section "Move-in" for details)
3. Remote Request Stored bit from CTRL2 register. Refer to Section "Control 2 Register (CTRL2)" for details.
4. Code 0b1010 is not considered Tx and a MB with this code should not be aborted.
5. Note that for Tx MBs, the BUSY bit should be ignored upon read, except when AEN bit is set in the MCR register. If this bit is asserted, the corresponding MB does not participate in the matching process.

Table 49-110. Message Buffer Code for Tx buffers

CODE Description	Tx Code BEFORE tx frame	MBRTR	Tx Code AFTER successful transmission	Comment
0b1000: INACTIVE - MB is not active	INACTIVE	-	-	MB does not participate in the arbitration process.
0b1001: ABORT - MB is aborted	ABORT	-	-	MB does not participate in the arbitration process.
0b1100: DATA - MB is a Tx Data Frame (MB RTR must be 0)	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission, the MB automatically returns to the INACTIVE state.
0b1100: REMOTE - MB is a Tx Remote Request Frame (MB RTR must be 1)	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the MB automatically becomes an Rx Empty MB with the same ID.
0b1110: TANSWER - MB is a Tx Response Frame from an incoming Remote Request Frame	TANSWER	-	RANSWER	This is an intermediate code that is automatically written to the MB by the CHI as a result of match to a remote request frame. The remote response frame will be transmitted unconditionally once and then the code will automatically return to RANSWER (0b1010). The CPU can also write this code with the same effect. The remote response frame can be either a data frame or another remote request frame depending on the RTR bit value. Refer to section “Matching Process" and section “Arbitration Process" for details.

SRR — Substitute Remote Request

Fixed recessive bit, used only in extended format. It must be set to '1' by the user for transmission (Tx Buffers) and will be stored with the value received on the CAN bus for Rx receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, then it is interpreted as arbitration loss.

1 = Recessive value is compulsory for transmission in Extended Format frames

0 = Dominant is not a valid value for transmission in Extended Format frames

IDE — ID Extended Bit

This bit identifies whether the frame format is standard or extended.

1 = Frame format is extended

0 = Frame format is standard

RTR — Remote Transmission Request

This bit affects the behavior of Remote Frames and is part of the reception filter. Refer to Table 49-109, Table 49-110 and the description of the RRS bit in Control 2 Register (CTRL2) for additional details.

If FlexCAN transmits this bit as '1' (recessive) and receives it as '0' (dominant), it is interpreted as arbitration loss. If this bit is transmitted as '0' (dominant), then if it is received as '1' (recessive), the FlexCAN module treats it as bit error. If the value received matches the value transmitted, it is considered as a successful bit transmission.

1 = Indicates the current MB may have a Remote Request Frame to be transmitted if MB is Tx. If the MB is Rx then incoming Remote Request Frames may be stored.

0 = Indicates the current MB has a Data Frame to be transmitted.. In Rx MB it may be considered in matching processes.

DLC — Length of Data in Bytes

This 4-bit field is the length (in bytes) of the Rx or Tx data, which is located in offset 0x8 through 0xF of the MB space (see Table 49-108). In reception, this field is written by the FlexCAN module, copied from the DLC (Data Length Code) field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR=1, the Frame to be transmitted is a Remote Frame and does not include the data field, regardless of the DLC field.

TIME STAMP — Free-Running Counter Time Stamp

This 16-bit field is a copy of the Free-Running Timer, captured for Tx and Rx frames at the time when the beginning of the Identifier field appears on the CAN bus.

PRIO — Local priority

This 3-bit field is only used when LPRIO_EN bit is set in MCR and it only makes sense for Tx mailboxes. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.

ID — Frame Identifier

In Standard Frame format, only the 11 most significant bits (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In Extended Frame format, all bits are used for frame identification in both receive and transmit cases.

DATA BYTE 0-7 — Data Field

Up to eight bytes can be used for a data frame.

For Rx frames, the data is stored as it is received from the CAN bus. DATA BYTE (n) is valid only if n is less than DLC as shown in the table below.

For Tx frames, the CPU prepares the data field to be transmitted within the frame.

Table 49-111. DATA BYTEs validity

DLC	Valid DATA BYTEs
0	none
1	DATA BYTE 0
2	DATA BYTE 0-1
3	DATA BYTE 0-2
4	DATA BYTE 0-3
5	DATA BYTE 0-4
6	DATA BYTE 0-5
7	DATA BYTE 0-6
8	DATA BYTE 0-7

49.3.57 Rx FIFO Structure

When the MCR[RFEN] bit is set, the memory area from 0x80 to 0xDC (which is normally occupied by MBs 0 to 5) is used by the reception FIFO enginee.

The region 0x80-0x8C contains the output of the FIFO which must be read by the CPU as a Message Buffer. This output contains the oldest message received and not read yet. The region 0x90-0xDC is reserved for internal use of the FIFO engine.

An additional memory area, that starts at 0xE0 and may extend up to 0x2DC (normally occupied by MBs 6 up to 37) depending on the CTRL2[RFFN] field setting, contains the ID Filter Table (configurable from 8 to 128 table elements) that specifies filtering criteria for accepting frames into the FIFO.

Out of reset, the ID Filter Table flexible memory area defaults to 0xE0 and only extends to 0xFC, which corresponds to MBs 6 to 7 for RFFN=0, for backward compatibility with previous versions of FlexCAN.

The following shows the Rx FIFO data structure.

Table 49-112. Rx FIFO Structure

	31	28	24	23	22	21	20	19	18	17	16	15	8	7	0
0x80	▨				SRR	IDE	RTR	DLC				TIME STAMP			
0x84	▨	ID Standard								ID Extended					
0x88	Data Byte 0			Data Byte 1								Data Byte 2		Data Byte 3	
0x8C	Data Byte 4			Data Byte 5								Data Byte 6		Data Byte 7	
0x90 to 0xDC	Reserved														
0xE0	ID Filter Table Element 0														
0xE4	ID Filter Table Element 1														
0xE8 to 0x2D4	ID Filter Table Elements 2 to 125														
0x2D8	ID Filter Table Element 126														
0x2DC	ID Filter Table Element 127														
	▨ = Unimplemented or Reserved														

Each ID Filter Table Element occupies an entire 32-bit word and can be compound by one, two or four Identifier Acceptance Filters (IDAF) depending on the MCR[IDAM] field setting. The following figures show the IDAF indexation.

The following figures show the three different formats of the ID table elements. Note that all elements of the table must have the same format. See Rx FIFO for more information.

Table 49-113. ID Table structure

Format	31	30	29	24	23	16	15	14	13	8	7	1	0
A	RTR	IDE	RXIDA (Standard = 29-19, Extended = 29-1)										▨

Table continues on the next page...

Table 49-113. ID Table structure (continued)

B	RTR	IDE	RXIDB_0 (Standard = 29-19, Extended = 29-16)	RTR	IDE	RXIDB_1 (Standard = 13-3, Extended = 13-0)
C	RXIDC_0 (Std/Ext = 31-24)	RXIDC_1 (Std/Ext = 23-16)	RXIDC_2 (Std/Ext = 15-8)	RXIDC_3 (Std/Ext = 7-0)		
	▨	= Unimplemented or Reserved				

RTR — Remote Frame

This bit specifies if Remote Frames are accepted into the FIFO if they match the target ID.

1 = Remote Frames can be accepted and data frames are rejected

0 = Remote Frames are rejected and data frames can be accepted

IDE — Extended Frame

Specifies whether extended or standard frames are accepted into the FIFO if they match the target ID.

1 = Extended frames can be accepted and standard frames are rejected

0 = Extended frames are rejected and standard frames can be accepted

RXIDA — Rx Frame Identifier (Format A)

Specifies an ID to be used as acceptance criteria for the FIFO. In the standard frame format, only the 11 most significant bits (29 to 19) are used for frame identification. In the extended frame format, all bits are used.

RXIDB_0, RXIDB_1 — Rx Frame Identifier (Format B)

Specifies an ID to be used as acceptance criteria for the FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (29 to 19 and 13 to 3) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.

RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3 — Rx Frame Identifier (Format C)

Specifies an ID to be used as acceptance criteria for the FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.

49.4 Functional Description

The FlexCAN module is a CAN protocol engine with a very flexible mailbox system for transmitting and receiving CAN frames. The mailbox system is composed by a set of up to 64Message Buffers (MB) that store configuration and control data, time stamp, message ID and data (see Message Buffer Structure). The memory corresponding to the first 38 MBs can be configured to support a FIFO reception scheme with a powerful ID filtering mechanism, capable of checking incoming frames against a table of IDs (up to 128 extended IDs or 256 standard IDs or 512 8-bit ID slices), with individual mask register for up to 32 ID tables. Simultaneous reception through FIFO and mailbox is supported. For mailbox reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed on its ID field. A masking scheme makes it possible to match the ID programmed on the MB with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of MBs to be transmitted based on the message ID (optionally augmented by 3 local priority bits) or the MB ordering.

Before proceeding with the functional description, an important concept must be explained. A Message Buffer is said to be "active" at a given time if it can participate in both the Matching and Arbitration processes. An Rx MB with a 0b0000 code is inactive (refer to Table 49-109). Similarly, a Tx MB with a 0b1000 or 0b1001 code is also inactive (refer to Table 49-110).

49.4.1 Transmit Process

In order to transmit a CAN frame, the CPU must prepare a Message Buffer for transmission by executing the following procedure:

1. Check if the respective interrupt bit is set and clear it.
2. If the MB is active (transmission pending), write the ABORT code (0b1001) to the CODE field of the Control and Status word to request an abortion of the transmission. Wait for the corresponding IFLAG to be asserted by polling the IFLAG register or by the interrupt request if enabled by the respective IMASK. Then read back the CODE field to check if the transmission was aborted or transmitted (see Transmission Abort Mechanism). If backwards compatibility is desired (MCR[AEN] bit is negated), just write the INACTIVE code (0b1000) to the CODE field to inactivate the MB but then the pending frame may be transmitted without notification (see Section "Message Buffer Inactivation").

3. Write the ID word.
4. Write the data bytes.
5. Write the DLC, Control and CODE fields of the Control and Status word to activate the MB.

Once the MB is activated in the fourth step, it will participate into the arbitration process and eventually be transmitted according to its priority. At the end of the successful transmission, the value of the Free Running Timer is written into the Time Stamp field, the CODE field in the Control and Status word is updated, the CRC Register is updated, a status flag is set in the Interrupt Flag Register and an interrupt is generated if allowed by the corresponding Interrupt Mask Register bit. The new CODE field after transmission depends on the code that was used to activate the MB in step four (see Table 49-109 and Table 49-110 in Section Message Buffer Structure).

When the Abort feature is enabled (MCR[AEN] is asserted), after the Interrupt Flag is asserted for a MB configured as transmit buffer, the MB is blocked, therefore the CPU is not able to update it until the Interrupt Flag is negated by CPU. This means that the CPU must clear the corresponding IFLAG before starting to prepare this MB for a new transmission or reception.

49.4.2 Arbitration process

The arbitration process scans the Mailboxes searching the Tx one that holds the message to be sent in the next opportunity. This Mailbox is called the *arbitration winner*.

The scan starts from the lowest number Mailbox and runs toward the higher ones.

The arbitration process is triggered in the following events:

- From the CRC field of the CAN frame. The start point depends on the CTRL2[TASD] field value.
- During the Error Delimiter field of a CAN frame.
- During the Overload Delimiter field of a CAN frame.
- When the winner is inactivated and the CAN bus has still not reached the first bit of the Intermission field.
- When there is CPU write to the C/S word of a winner MB and the CAN bus has still not reached the first bit of the Intermission field.
- When CHI is in Idle state and the CPU writes to the C/S word of any MB.

- When FlexCAN exits Bus Off state.
- Upon leaving Freeze Mode or Low Power Mode.

If the arbitration process does not manage to evaluate all Mailboxes before the CAN bus has reached the first bit of the Intermission field the temporary arbitration winner is invalidated and the FlexCAN will not compete for the CAN bus in the next opportunity.

The arbitration process selects the winner among the active Tx Mailboxes at the end of the scan according to both CTRL1[LBUF] and MCR[LPRIO_EN] bits settings.

49.4.2.1 Lowest number Mailbox first

If CTRL1[LBUF] bit is asserted the first (lowest number) active Tx Mailbox found is the arbitration winner. MCR[LPRIO_EN] bit has no effect when CTRL1[LBUF] is asserted.

49.4.2.2 Highest priority Mailbox first

If CTRL1[LBUF] bit is negated then the arbitration process searches the active Tx Mailbox with the highest priority, which means that this Mailbox’s frame would have a higher probability to win the arbitration on CAN bus with multiple nodes driving each Tx Mailbox’s frame at the same time.

The sequence of bits considered for this arbitration is called the *arbitration value* of the Mailbox. The highest priority Tx Mailbox is the one that has the least arbitration value among all Tx Mailboxes.

If two or more Mailboxes have equivalent arbitration values the lowest number Mailbox is the arbitration winner.

The composition of the arbitration value depends on MCR[LPRIO_EN] bit setting.

49.4.2.2.1 Local Priority disabled

If MCR[LPRIO_EN] bit is negated the arbitration value is built in the exact sequence of bits as they would be transmitted in a CAN frame (see the following table) in such a way that the Local Priority is disabled.

Table 49-114. Composition of the arbitration value when Local Priority is disabled

Format	Mailbox Arbitration Value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

49.4.2.2.2 Local Priority enabled

If Local Priority is desired MCR[LPRIO_EN] must be asserted. In this case the Mailbox PRIO field is included at the very left of the arbitration value (see the following table).

Table 49-115. Composition of the arbitration value when Local Priority is enabled

Format	Mailbox Arbitration Value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

As the PRIO field is the most significant part of the arbitration value Mailboxes with low PRIO values have higher priority than Mailboxes with high PRIO values regardless the rest of their arbitration values.

Note that the PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

49.4.2.3 Arbitration Process (continued)

Once the arbitration winner is found, its content is copied to a hidden auxiliary MB called Tx Serial Message Buffer (Tx SMB), which has the same structure as a normal MB but is not user accessible. This operation is called “move-out” and after it is done, write access to the corresponding MB is blocked (if the AEN bit in MCR is asserted). The write access is released in the following events:

- After the MB is transmitted
- FlexCAN enters in Freeze Mode or Bus Off
- FlexCAN loses the bus arbitration or there is an error during the transmission

At the first opportunity window on the CAN bus, the message on the Tx SMB is transmitted according to the CAN protocol rules. FlexCAN transmits up to eight data bytes, even if the DLC (Data Length Code) field value is greater than that.

Arbitration process can be triggered in the following situations:

- During Rx and Tx frames from CAN CRC field to end of frame. Arbitration start point depends on instantiation parameters NUMBER_OF_MB and TASD. Additionally, TASD value may be changed to optimize the arbitration start point.
- During CAN BusOff state from TX_ERR_CNT=124 to 128. Arbitration start point depends on instantiation parameters NUMBER_OF_MB and TASD. Additionally, TASD value may be changed to optimize the arbitration start point.
- During C/S write by CPU in BusIdle. First C/S write starts arbitration process and a second C/S write during this same arbitration restarts the process. If other C/S writes are performed, Tx arbitration process is pending. If there is no arbitration winner after arbitration process has finished, then TX arbitration machine begins a new arbitration process.
- - If there is a pending arbitration and BusIdle state starts then an arbitration process is triggered. In this case the first and second C/S write in BusIdle will not restart the arbitration process. It is possible that there is not enough time to finish arbitration in WaitForBusIdle state and the next state is Idle. In this case the scan is not interrupted, and it is completed during BusIdle state. During this arbitration C/S write does not cause arbitration restart.
- Arbitration winner deactivation during a valid arbitration window.
- Upon Leave Freeze Mode (first bit of the WaitForBusIdle state). If there is a re-synchronization during WaitForBusIdle arbitration process is restarted.

Arbitration process stops in the following situation:

- All Mailboxes were scanned.
- A Tx active Mailbox is found in case of Lowest Buffer feature enabled.
- Arbitration winner inactivation or abort during any arbitration process.
- There was not enough time to finish Tx arbitration process. For instance, a deactivation was performed near the end of frame). In this case arbitration process is pending.
- Error or Overload flag in the bus .
- Low Power or Freeze Mode request in Idle state

Arbitration is considered pending as described below:

- It was not possible to finish arbitration process in time.
- C/S write during arbitration if write is performed in a MB which number is lower than the Tx arbitration pointer .
- Any C/S write if there is no Tx Arbitration process in progress.
- Rx Match has just updated a Rx Code to Tx Code.
- Entering Busoff state.

C/S write during arbitration has the following effect:

- If C/S write is performed in the arbitration winner, a new process is restarted immediately.
- If C/S write is performed in a MB whose number is higher than the Tx arbitration pointer, the ongoing arbitration process will scan this MB as normal.

49.4.3 Receive Process

To be able to receive CAN frames into a Mailbox, the CPU must prepare it for reception by executing the following steps:

1. If the Mailbox is active (either Tx or Rx) inactivate the Mailbox (see Section "Message Buffer Inactivation"), preferably with a safe inactivation (see Transmission Abort Mechanism).
2. Write the ID word
3. Write the EMPTY code (0b0100) to the CODE field of the Control and Status word to activate the Mailbox.

Once the MB is activated, it will be able to receive frames that match the programmed filter. At the end of a successful reception, the Mailbox is updated by the *move-in* process (see Section "Move-in") as follows:

1. The received Data field (8 bytes at most) is stored.
2. The received Identifier field is stored.
3. The value of the Free Running Timer at the time of the second bit of frame's Identifier field is written into the Mailbox's Time Stamp field.
4. The received SRR, IDE, RTR and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 49-109 and Table 49-110 in Section Message Buffer Structure).
6. A status flag is set in the Interrupt Flag Register and an interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The recommended way for CPU servicing (read) the frame received in an Mailbox is using the following procedure:

1. Read the Control and Status word of that Mailbox.
2. Check if the BUSY bit is deasserted, indicating that the Mailbox is locked. Repeat step 1) while it is asserted. See Section "Message Buffer Lock Mechanism".

3. Read the contents of the Mailbox. Once Mailbox is locked now, its contents won't be modified by FlexCAN Move-in processes. See Section "Move-in".
4. Acknowledge the proper flag at IFLAG registers.
5. Read the Free Running Timer. It is optional but recommended to unlock Mailbox as soon as possible and make it available for reception.

The CPU should synchronize to frame reception by the status flag bit for the specific Mailbox in one of the IFLAG Registers and not by the CODE field of that Mailbox. Polling the CODE field does not work because once a frame was received and the CPU services the Mailbox (by reading the C/S word followed by unlocking the Mailbox), the CODE field will not return to EMPTY. It will remain FULL, as explained in Table 49-109 . If the CPU tries to workaround this behavior by writing to the C/S word to force an EMPTY code after reading the Mailbox without a prior *safe inactivation*, a newly received frame matching the filter of that Mailbox may be lost.

CAUTION

> In summary: *never do polling by reading directly the C/S word of the Mailboxes. Instead, read the IFLAG registers.*

Note that the received frame's Identifier field is always stored in the matching Mailbox, thus the contents of the ID field in an Mailbox may change if the match was due to masking. Note also that FlexCAN does receive frames transmitted by itself if there exists a matching Rx Mailbox, provided the MCR[SRXDIS] bit is not asserted. If the MCR[SRXDIS] bit is asserted, FlexCAN will not store frames transmitted by itself in any MB, even if it contains a matching MB, and no interrupt flag or interrupt signal will be generated due to the frame reception.

To be able to receive CAN frames through the Rx FIFO, the CPU must enable and configure the Rx FIFOduring Freeze Mode (see Rx FIFO). Upon receiving the Frames Available in Rx FIFO interrupt (see the description of the IFLAG[BUF5I] "Frames available in Rx FIFO" bit in the IMASK1 register), the CPU should service the received frame using the following procedure:

1. Read the Control and Status word (optional – needed only if a mask was used for IDE and RTR bits)
2. Read the ID field (optional – needed only if a mask was used)
3. Read the Data field
4. Read the RXFIR register (optional)

5. Clear the Frames Available in Rx FIFO interrupt by writing 1 to IFLAG[BUF5I] bit (mandatory – releases the MB and allows the CPU to read the next Rx FIFO entry)

49.4.4 Matching Process

The matching process scans the MB memory looking for Rx MBs programmed with the same ID as the one received from the CAN bus. If the FIFO is enabled, the priority of scanning can be selected between Mailboxes and FIFO filters. In any case, the matching starts from the lowest number Message Buffer toward the higher ones. If no match is found within the first structure then the other is scanned subsequently. In the event that the FIFO is full, the matching algorithm will always look for a matching MB outside the FIFO region.

As the frame is being received, it is stored in a hidden auxiliary MB called Rx Serial Message Buffer (Rx SMB).

The matching process start point depends on the following conditions:

- if the received frame is a remote frame, the start point is the CRC field of the frame;
- if the received frame is a data frame with DLC field equal to zero, the start point is the CRC field of the frame;
- if the received frame is a data frame with DLC field different than zero, the start point is the DATA field of the frame;

If a matching ID is found in the FIFO table or in one of the Mailboxes, the contents of the SMB will be transferred to the FIFO or to the matched Mailbox by the move-in process. If any CAN protocol error is detected then no match results will be transferred to the FIFO or to the matched Mailbox at the end of reception.

The matching process scans all matching elements of both Rx FIFO (if enabled) and active Rx Mailboxes (CODE is EMPTY, FULL, OVERRUN or RANSWER) in search of a successful comparison with the matching elements of the Rx SMB that is receiving the frame on the CAN bus. The SMB has the same structure of a Mailbox. The reception structures (Rx FIFO or Mailboxes) associated with the matching elements that had a successful comparison are the *matched structures*. The *matching winner* is selected at the end of the scan among those matched structures and depends on conditions described ahead. See the following table.

Table 49-116. Matching Architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Mailbox	0	-	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY or FULL or OVERRUN
Mailbox	0	-	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	0	-	cmp	no_cmp	cmp	RANSWER
Mailbox	1	1	0	cmp	no_cmp	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
FIFO[5]	-	-	-	cmp_msk	cmp_msk	cmp_msk	-

[1] For Mailbox structure, If SMB[IDE] is asserted, the ID is 29 bits (ID Standard + ID Extended). If SMB[IDE] is negated, the ID is only 11 bits (ID Standard). For FIFO structure, the ID depends on IDAM.
[2] cmp: Compares the SMB contents with the MB contents regardless the masks.
[3] no_cmp: The SMB contents are not compared with the MB contents
[4] cmp_msk: Compares the SMB contents with MB contents taking into account the masks.
[5] SMB[IDE] and SMB[RTR] are not taken into account when IDAM is type C.

A reception structure is *free-to-receive* when any of the following conditions is satisfied:

- the CODE field of the Mailbox is EMPTY;
- the CODE field of the Mailbox is either FULL or OVERRUN and it has already been serviced (the C/S word was read by the CPU and unlocked as described in Message Buffer Lock Mechanism);
- the CODE field of the Mailbox is either FULL or OVERRUN and a inactivation (see Message Buffer Inactivation) is performed;
- the Rx FIFO is not full.

The scan order for Mailboxes and Rx FIFO is from the matching element with lowest number to the higher ones.

The matching winner search for Mailboxes is affected by the MCR[IRMQ] bit. If it is negated the matching winner is the first matched Mailbox regardless if it is free-to-receive or not. If it is asserted, the matching winner is selected according to the priority below:

1. the first free-to-receive matched Mailbox;
2. the last non free-to-receive matched Mailbox.

It is possible to select the priority of scan between Mailboxes and Rx FIFO by the CTRL2[MRP] bit.

If the selected priority is Rx FIFO first:

- if the Rx FIFO is a matched structure and is free-to-receive then the Rx FIFO is the matching winner regardless of the scan for Mailboxes;
- otherwise (the Rx FIFO is not a matched structure or is not free-to-receive), then the matching winner is searched among Mailboxes as described above.

If the selected priority is Mailboxes first:

- if a free-to-receive matched Mailbox is found, it is the matching winner regardless the scan for Rx FIFO;
- if no matched Mailbox is found, then the matching winner is searched in the scan for the Rx FIFO;

If both conditions above are not satisfied and a non free-to-receive matched Mailbox is found then the matching winner determination is conditioned by the MCR[IRMQ] bit:

- if MCR[IRMQ] bit is negated the matching winner is the first matched Mailbox;
- if MCR[IRMQ] bit is asserted the matching winner is the Rx FIFO if it is a free-to-receive matched structure, otherwise the matching winner is the last non free-to-receive matched Mailbox.

See the following table for a summary of matching possibilities.

Table 49-117. Matching Possibilities and Resulting Reception Structures

RFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception Structure	Description
No FIFO, only MB, match is always MB first						
0	0	X[1]	None[2]	-[3]	None	Frame lost by no match
0	0	X	Free[4]	-	FirstMB	
0	1	X	None	-	None	Frame lost by no match
0	1	X	Free	-	FirstMb	
0	1	X	NotFree	-	LastMB	Overrun
FIFO enabled, no match in FIFO is as if FIFO does not exist						
1	0	X	None	None[5]	None	Frame lost by no match
1	0	X	Free	None	FirstMB	
1	1	X	None	None	None	Frame lost by no match
1	1	X	Free	None	FirstMb	
1	1	X	NotFree	None	LastMB	Overrun

Table continues on the next page...

Table 49-117. Matching Possibilities and Resulting Reception Structures (continued)

RFEN	IRMQ	MRP	Matched in MB	Matched in FIFO	Reception Structure	Description
FIFO enabled, Queue disabled						
1	0	0	X	NotFull[6]	FIFO	
1	0	0	None	Full[7]	None	Frame lost by FIFO full (FIFO Overflow)
1	0	0	Free	Full	FirstMB	
1	0	0	NotFree	Full	FirstMB	
1	0	1	None	NotFull	FIFO	
1	0	1	None	Full	None	Frame lost by FIFO full (FIFO Overflow)
1	0	1	Free	X	FirstMB	
1	0	1	NotFree	X	FirtsMb	Overrun
FIFO enabled, Queue enabled						
1	1	0	X	NotFull	FIFO	
1	1	0	None	Full	None	Frame lost by FIFO full (FIFO Overflow)
1	1	0	Free	Full	FirstMB	
1	1	0	NotFree	Full	LastMb	Overrun
1	1	1	None	NotFull	FIFO	
1	1	1	Free	X	FirstMB	
1	1	1	NotFree	NotFull	FIFO	
1	1	1	NotFree	Full	LastMb	Overrun

1. This is a don't care condition.
2. Matched in MB "None" means that the frame has not matched any MB (free-to-receive or non-free-to-receive).
3. This is a forbidden condition.
4. Matched in MB "Free" means that the frame matched at least one MB free-to-receive regardless of whether it has matched MBs non-free-to-receive.
5. Matched in FIFO "None" means that the frame has not matched any filter in FIFO. It is as if the FIFO didn't exist (CTRL2[RFEN]=0).
6. Matched in FIFO "NotFull" means that the frame has matched a FIFO filter and has empty slots to receive it.
7. Matched in FIFO "Full" means that the frame has matched a FIFO filter but couldn't store it because it has no empty slots to receive it.

If a non-safe Mailbox inactivation (see Message Buffer Inactivation) occurs during matching process and the Mailbox inactivated is the temporary matching winner then the temporary matching winner is invalidated. The matching elements scan is not stopped nor restarted, it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist, therefore a message may be lost.

Suppose, for example, that the FIFO is disabled, IRMQ is enabled and there are two MBs with the same ID, and FlexCAN starts receiving messages with that ID. Let us say that these MBs are the second and the fifth in the array. When the first message arrives, the matching algorithm will find the first match in MB number 2. The code of this MB is EMPTY, so the message is stored there. When the second message arrives, the matching algorithm will find MB number 2 again, but it is not "free-to-receive", so it will keep looking and find MB number 5 and store the message there. If yet another message with the same ID arrives, the matching algorithm finds out that there are no matching MBs that are "free-to-receive", so it decides to overwrite the last matched MB, which is number 5. In doing so, it sets the CODE field of the MB to indicate OVERRUN.

The ability to match the same ID in more than one MB can be exploited to implement a reception queue (in addition to the full featured FIFO) to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages will be queued into the MBs. The CPU can examine the Time Stamp field of the MBs to determine the order in which the messages arrived.

Matching to a range of IDs is possible by using ID Acceptance Masks. FlexCAN supports individual masking per MB. Refer to the description of the Rx Individual Mask Registers (RXIMRx). During the matching algorithm, if a mask bit is asserted, then the corresponding ID bit is compared. If the mask bit is negated, the corresponding ID bit is "don't care". Please note that the Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed while the module is in Freeze Mode; otherwise, they are blocked by hardware.

FlexCAN also supports an alternate masking scheme with only four mask registers (RGXMASK, RX14MASK, RX15MASK and RXFGMASK) for backwards compatibility. This alternate masking scheme is enabled when the IRMQ bit in the MCR Register is negated.

49.4.5 Move Process

There are two types of move process: move-in and move-out.

49.4.5.1 Move-in

The move-in process is the copy of a message received by an Rx SMB to a Rx Mailbox or FIFO that has matched it. If the move destination is the Rx FIFO, attributes of the message are also copied to the RXFIR FIFO. Each Rx SMB has its own move-in process, but only one is performed at a given time as described ahead. The move-in starts only when the message held by the Rx SMB has a corresponding matching winner (see Section "Matching Process") and all of the following conditions are true:

- the CAN bus has reached or let past either:
- - the second bit of Intermission field next to the frame that carried the message that is in the Rx SMB;
  - the first bit of an overload frame next to the frame that carried the message that is in the Rx SMB;
- there is no ongoing matching process;
- the destination Mailbox is not locked by the CPU;
- there is no ongoing move-in process from another Rx SMB. If more than one move-in processes are to be started at the same time both are performed and the newest substitutes the oldest.

The term *pending move-in* is used throughout the document and stands for a move-to-be that still does not satisfy all of the aforementioned conditions.

The move-in is cancelled and the Rx SMB is able to receive another message if any of the following conditions is satisfied:

- the destination Mailbox is inactivated after the CAN bus has reached the first bit of Intermission field next to the frame that carried the message and its matching process has finished;
- there is a previous pending move-in to the same destination Mailbox;
- the Rx SMB is receiving a frame transmitted by the FlexCAN itself and the self-reception is disabled (MCR[SRXDIS] bit is asserted);
- any CAN protocol error is detected.

Note that the pending move-in is not cancelled if the module enters Freeze or Low Power Mode. It only stays on hold waiting for exiting Freeze and Low Pwer Mode and to be unlocked. If an MB is unlocked during Freeze Mode, the move-in happens immediately.

The move-in process is the execution by the FlexCAN of the following steps:

1. if the message is destined to the Rx FIFO, push IDHIT into the RXFIR FIFO;
2. reads the words DATA0-3 and DATA4-7 from the Rx SMB;
3. writes it in the words DATA0-3 and DATA4-7 of the Rx Mailbox;
4. reads the words Control/Status and ID from the Rx SMB;
5. writes it in the words Control/Status and ID of the Rx Mailbox, updating the CODE field.

The move-in process is not atomic, in such a way that it is immediately cancelled by the inactivation of the destination Mailbox (see Section "Message Buffer Inactivation") and in this case the Mailbox may be left partially updated, thus incoherent. The exception is if the move-in destination is an Rx FIFO Message Buffer, then the process cannot be cancelled.

The BUSY Bit (least significant bit of the CODE field) of the destination Message Buffer is asserted while the move-in is being performed to alert the CPU that the Message Buffer content is temporarily incoherent.

49.4.5.2 Move-out

The move-out process is the copy of the content from a Tx Mailbox to the Tx SMB when a message for transmission is available (see Section "Arbitration process"). The move-out occurs in the following conditions:

- the first bit of Intermission field;
- during Bussoff field when TX Error Counter is in the 124 to 128 range;
- during BusIdle field
- during Wait For Bus Idle field

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently out of BusIdle state. In BusIdle, the move-out has the lowest priority to the concurrent memory accesses.

49.4.6 Data Coherence

In order to maintain data coherency and FlexCAN proper operation, the CPU must obey the rules described in Transmit Process and Receive Process. Any form of CPU accessing an MB structure within FlexCAN other than those specified may cause FlexCAN to behave in an unpredictable way.

49.4.6.1 Transmission Abort Mechanism

The abort mechanism provides a safe way to request the abortion of a pending transmission. A feedback mechanism is provided to inform the CPU if the transmission was aborted or if the frame could not be aborted and was transmitted instead.

Two primary conditions must be fulfilled in order to abort a transmission:

- MCR[AEN] bit must be asserted;
- the first CPU action must be the writing of abort code (0b1001) into the CODE field of the Control and Status word.

The active MBs configured as transmission must be aborted first and then they may be updated. If the abort code is written to a Mailbox that is currently being transmitted, or to a Mailbox that was already loaded into the SMB for transmission, the write operation is blocked and the MB is kept active, but the abort request is captured and kept pending until one of the following conditions are satisfied:

- The module loses the bus arbitration
- There is an error during the transmission
- The module is put into Freeze Mode
- The module enters in BusOff state
- There is an overload frame

If none of conditions above are reached, the MB is transmitted correctly, the interrupt flag is set in the IFLAG register and an interrupt to the CPU is generated (if enabled). The abort request is automatically cleared when the interrupt flag is set. On the other hand, if one of the above conditions is reached, the frame is not transmitted; therefore, the abort code is written into the CODE field, the interrupt flag is set in the IFLAG and an interrupt is (optionally) generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, then the write operation is not blocked; therefore, the MB is updated and the interrupt flag is set. In this way the CPU just needs to read the abort code to make sure the active MB was *safely inactivated*. Although the AEN bit is asserted and the CPU wrote the abort code, in this case the MB is inactivated and not aborted, because the transmission did not start yet. One Mailbox is only aborted when the abort request is captured and kept pending until one of the previous conditions are satisfied.

The abort procedure can be summarized as follows:

- CPU checks the corresponding IFLAG and clears it, if asserted.
- CPU writes 0b1001 into the CODE field of the C/S word.
- CPU waits for the corresponding IFLAG indicating that the frame was either transmitted or aborted.

- CPU reads the CODE field to check if the frame was either transmitted (CODE=0b1000) or aborted (CODE=0b1001).
- It is necessary to clear the corresponding IFLAG in order to allow the MB to be reconfigured.

49.4.6.2 Message Buffer Inactivation

Inactivation is a mechanism provided to protect the Mailbox against updates by the FlexCAN internal processes, thus allowing the CPU to rely on Mailbox data coherence after having updated it, even in Normal Mode.

Inactivation of transmission Mailboxes must be performed just when MCR[AEN] bit is deasserted.

If a Mailbox is inactivated it participates in neither the arbitration process nor the matching process until it is reactivated. See Section "Transmit Process" and Section "Receive Process" for more detailed instruction on how to inactivate and reactivate a Mailbox.

In order to inactivate a Mailbox the CPU must update its CODE field to INACTIVE (either 0b0000 or 0b1000).

As the user is not able to synchronize the CODE field update with the FlexCAN internal processes an inactivation can lead to undesirable results:

- a frame in the bus that matches the filtering of the inactivated Rx Mailbox may be lost without notice, even if there are other Mailboxes with the same filter;
- a frame containing the message within the inactivated Tx Mailbox may be transmitted without notice.

In order to eliminate such risk and perform a *safe inactivation* the CPU must use the following mechanism along with the inactivation itself:

- for Tx Mailboxes, the Transmission Abort (see Section "Transmission Abort Mechanism");

The inactivation automatically unlocks the Mailbox (see Section "Message Buffer Lock Mechanism").

NOTE

> Message Buffers that are part of the Rx FIFO cannot be inactivated. There is no write protection on the FIFO region by

FlexCAN. CPU must maintain data coherency in the FIFO region when RFEN is asserted.

49.4.6.3 Message Buffer Lock Mechanism

Besides MB inactivation, FlexCAN has another data coherence mechanism for the receive process. When the CPU reads the Control and Status word of an Rx MB with codes FULL or OVERRUN, FlexCAN assumes that the CPU wants to read the whole MB in an atomic operation, and thus it sets an internal lock flag for that MB. The lock is released when the CPU reads the Free Running Timer (global unlock operation), or when it reads the Control and Status word of another MB regardless of its code, or when the CPU writes into C/S word from locked MB. The MB locking is done to prevent a new frame to be written into the MB while the CPU is reading it.

NOTE

The locking mechanism only applies to Rx MBs that are not part of FIFO and have a code different than INACTIVE (0b0000) or EMPTY[1] (0b0100). Also, Tx MBs can not be locked.

Suppose, for example, that the FIFO is disabled and the second and the fifth MBs of the array are programmed with the same ID, and FlexCAN has already received and stored messages into these two MBs. Suppose now that the CPU decides to read MB number 5 and at the same time another message with the same ID is arriving. When the CPU reads the Control and Status word of MB number 5, this MB is locked. The new message arrives and the matching algorithm finds out that there are no "free-to-receive" MBs, so it decides to override MB number 5. However, this MB is locked, so the new message can not be written there. It will remain in the SMB waiting for the MB to be unlocked, and only then will be written to the MB. If the MB is not unlocked in time and yet another new message with the same ID arrives, then the new message overwrites the one on the SMB and there will be no indication of lost messages either in the CODE field of the MB or in the Error and Status Register.

While the message is being moved-in from the SMB to the MB, the BUSY bit on the CODE field is asserted. If the CPU reads the Control and Status word and finds out that the BUSY bit is set, it should defer accessing the MB until the BUSY bit is negated.

Note

If the BUSY bit is asserted or if the MB is empty, then reading the Control and Status word does not lock the MB.

[1] In previous FlexCAN versions, reading the C/S word locked the MB even if it was EMPTY. This behavior is maintained when the IRMQ bit is negated.

Inactivation takes precedence over locking. If the CPU inactivates a locked Rx MB, then its lock status is negated and the MB is marked as invalid for the current matching round. Any pending message on the SMB will not be transferred anymore to the MB. An MB is unlocked when the CPU reads the Free Running Timer Register (see Section "Free Running Timer Register (TIMER)"), or the C/S word of another MB.

Lock and unlock mechanisms have the same functionality in both Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. However, the move-in is postponed if an unlock occurs during any of the low power modes (see in Section "Modes of Operation" specific information on Module Disable, Doze or Stop modes) and it will take place only when the module resumes to Normal or Freeze modes.

49.4.7 Rx FIFO

The receive-only FIFO is enabled by asserting the RFEN bit in the MCR. The reset value of this bit is zero to maintain software backward compatibility with previous versions of the module that did not have the FIFO feature. The FIFO is 6-message deep, therefore when the FIFO is enabled, the memory region occupied by the first 6 Message Buffers is reserved for use of the FIFO engine (see Rx FIFO Structure). The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a Message Buffer structure at the output of the FIFO.

The IFLAG[BUF5I] (Frames available in Rx FIFO) is asserted when there is at least one frame available to be read from the FIFO. An interrupt is generated if it is enabled by the corresponding mask bit. Upon receiving the interrupt, the CPU can read the message (accessing the output of the FIFO as a Message Buffer) and the RXFIR register and then clear the interrupt. If there are more messages in the FIFO the act of clearing the interrupt updates the output of the FIFO with the next message and update the RXFIR with the attributes of that message, reissuing the interrupt to the CPU. Otherwise, the flag remains negated. The output of the FIFO is only valid whilst the IFLAG[BUF5I] is asserted.

The IFLAG[BUF6I] (Rx FIFO Warning) is asserted when the number of unread messages within the Rx FIFO is increased to 5 from 4 due to the reception of a new one, meaning that the Rx FIFO is almost full. The flag remains asserted until the CPU clears it.

The IFLAG[BUF7I] (Rx FIFO Overflow) is asserted when an incoming message was lost because the Rx FIFO is full. Note that the flag will not be asserted when the Rx FIFO is full and the message was captured by a Mailbox. The flag remains asserted until the CPU clears it.

Clearing one of those three flags does not affect the state of the other two.

An interrupt is generated if an IFLAG bit is asserted and the corresponding mask bit is asserted too.

A powerful filtering scheme is provided to accept only frames intended for the target application, thus reducing the interrupt servicing work load. The filtering criteria is specified by programming a table of up to 128 32-bit registers, according to CTRL2[RFFN] setting, that can be configured to one of the following formats (see also Rx FIFO Structure):

- Format A: 128 IDAFs (extended or standard IDs including IDE and RTR)
- Format B: 256 IDAFs (standard IDs or extended 14-bit ID slices including IDE and RTR)
- Format C: 512 IDAFs (standard or extended 8-bit ID slices)

Note

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the FIFO has a corresponding IDHIT (Identifier Acceptance Filter Hit Indicator) that can be read by accessing the RXFIR register. The RXFIR[IDHIT] field refers to the message at the output of the FIFO and is valid while the IFLAG[BUF5I] flag is asserted. The RXFIR register must be read only before clearing the flag, which guarantees that the information refers to the correct frame within the FIFO.

Up to thirty two elements of the filter table are individually affected by the Individual Mask Registers (RXIMRx), according to the setting of CTRL2[RFFN], allowing very powerful filtering criteria to be defined. If the IRMQ bit is negated, then the FIFO filter table is affected by RXFGMASK.

49.4.8 CAN Protocol Related Features

This section describes the CAN protocol related features.

49.4.8.1 Remote Frames

Remote frame is a special kind of frame. The user can program a mailbox to be a Remote Request Frame by writing the mailbox as Transmit with the RTR bit set to '1'. After the remote request frame is transmitted successfully, the mailbox becomes a Receive Message Buffer, with the same ID as before.

When a remote request frame is received by FlexCAN, it can be treated in three ways, depending on Remote Request Storing (CTRL2[RRS]) and Rx FIFO Enable (MCR[RFEN]) bits:

- If RRS is negated the frame's ID is compared to the IDs of the Transmit Message Buffers with the CODE field 0b1010. If there is a matching ID, then this mailbox frame will be transmitted. Note that if the matching mailbox has the RTR bit set, then FlexCAN will transmit a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response. The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. In the case that a remote request frame was received and matched a mailbox, this message buffer immediately enters the internal arbitration process, but is considered as normal Tx mailbox, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.
- If RRS is asserted the frame's ID is compared to the IDs of the receive mailboxes with the CODE field 0b0100, 0b0010 or 0b0110. If there is a matching ID, then this mailbox will store the remote frame in the same fashion of a data frame. No automatic remote response frame will be generated. The mask registers are used in the matching process.
- If RFEN is asserted FlexCAN will not generate an automatic response for remote request frames that match the FIFO filtering criteria. If the remote frame matches one of the target IDs, it will be stored in the FIFO and presented to the CPU. Note that for filtering formats A and B, it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted (if they match the ID). Remote Request Frames are considered as normal frames, and generate a FIFO overflow when a successful reception occurs and the FIFO is already full.

49.4.8.2 Overload Frames

FlexCAN does transmit overload frames due to detection of following conditions on CAN bus:

- Detection of a dominant bit in the first/second bit of Intermission
- Detection of a dominant bit at the 7th bit (last) of End of Frame field (Rx frames)
- Detection of a dominant bit at the 8th bit (last) of Error Frame Delimiter or Overload Frame Delimiter

49.4.8.3 Time Stamp

The value of the Free Running Timer is sampled at the beginning of the Identifier field on the CAN bus, and is stored at the end of "move-in" in the TIME STAMP field, providing network behavior with respect to time.

Note that the Free Running Timer can be reset upon a specific frame reception, enabling network time synchronization. Refer to the TSYN description in the description of the Control 1 Register (CTRL1).

49.4.8.4 Protocol Timing

The following figure shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) sub-module. The clock source bit CLKSRC in the CTRL1 Register defines whether the internal clock is connected to the output of a crystal oscillator (Oscillator Clock) or to the Peripheral Clock (generally from a PLL). In order to guarantee reliable operation, the clock source should be selected while the module is in Disable Mode (bit MDIS set in the Module Configuration Register).

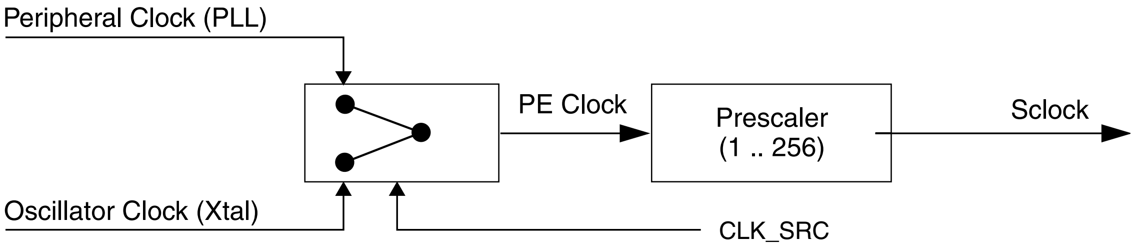


Figure 49-104. CAN Engine Clocking Scheme

The crystal oscillator clock should be selected whenever a tight tolerance (up to 0.1%) is required in the CAN bus timing. The crystal oscillator clock has better jitter performance than PLL generated clocks.

The FlexCAN module supports a variety of means to setup bit timing parameters that are required by the CAN protocol. The Control Register has various fields used to control bit timing parameters: PRESDIV, PROPSEG, PSEG1, PSEG2 and RJW. See the description of the Control 1 Register (CTRL1).

The PRESDIV field controls a prescaler that generates the Serial Clock (Sclock), whose period defines the 'time quantum' used to compose the CAN waveform. A time quantum is the atomic unit of time handled by the CAN engine.

$$f_{Tq} = \frac{f_{CANCLK}}{(\text{Prescaler Value})}$$

A bit time is subdivided into three segments[2] (reference Figure 49-105 and Table 49-118):

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section
- Time Segment 1: This segment includes the Propagation Segment and the Phase Segment 1 of the CAN standard. It can be programmed by setting the PROPSEG and the PSEG1 fields of the CTRL1 Register so that their sum (plus 2) is in the range of 4 to 16 time quanta
- Time Segment 2: This segment represents the Phase Segment 2 of the CAN standard. It can be programmed by setting the PSEG2 field of the CTRL1 Register (plus 1) to be 2 to 8 time quanta long

$$\text{Bit Rate} = \frac{f_{Tq}}{(\text{number of Time Quanta})}$$

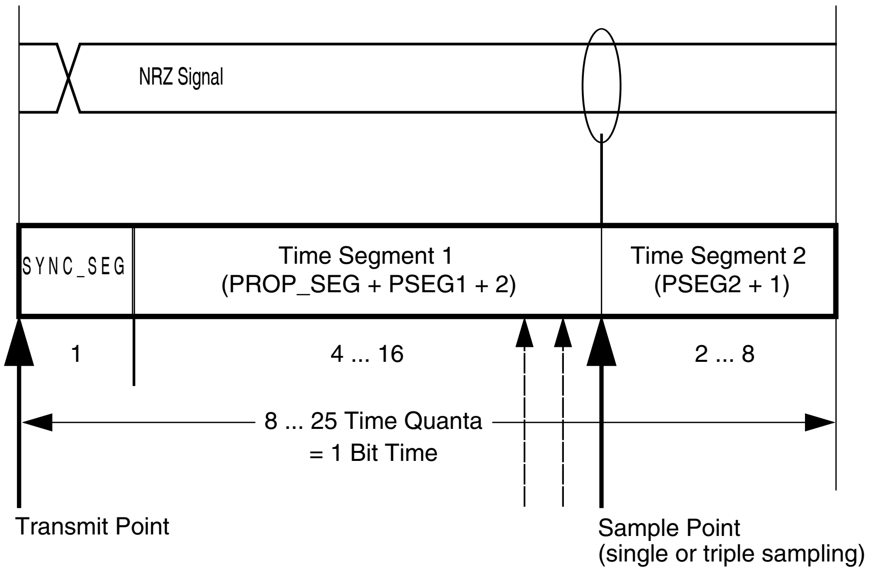


Figure 49-105. Segments within the Bit Time

[2] For further explanation of the underlying concepts please refer to ISO/DIS 11519–1, Section 10.3. Reference also the Bosch CAN 2.0A/B protocol specification dated September 1991 for bit timing.

Whenever CAN bit is used as a measure of duration (e.g. MCR[FRZACK] and MCR[LPMACK]), the number of peripheral clocks in one CAN bit can be calculated as:

$$\text{NCCP} = \frac{f_{\text{SYS}} \text{ x } [1 + (\text{PSEG1} + 1) + (\text{PSEG2} + 1) + (\text{PROPSEG} + 1)] \text{ x } (\text{PRESDIV} + 1)}{f_{\text{CANCLK}}}$$

where:

- NCCP is the number of peripheral clocks in one CAN bit;
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure "CAN Engine Clocking Scheme"), in Hz;
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz;
- PSEG1 is the value in CTRL1[PSEG1] field;
- PSEG2 is the value in CTRL1[PSEG2] field;
- PROPSEG is the value in CTRL1[PROPSEG] field;
- PRESDIV is the value in CTRL1[PRESDIV] field.

For example, 180 CAN bits = 180 x NCCP peripheral clock periods.

Table 49-118. Time Segment Syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the bus during this period.
Transmit Point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample Point	A node samples the bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The following table gives an overview of the CAN compliant segment settings and the related parameter values.

Table 49-119. CAN Standard Compliant Bit Time Segment Settings

Time Segment 1	Time Segment 2	Re-synchronization Jump Width
5 .. 10	2	1 .. 2
4 .. 11	3	1 .. 3
5 .. 12	4	1 .. 4
6 .. 13	5	1 .. 4
7 .. 14	6	1 .. 4
8 .. 15	7	1 .. 4
9 .. 16	8	1 .. 4

Note

It is the user's responsibility to ensure the bit time settings are in compliance with the CAN standard. For bit time calculations, use an IPT (Information Processing Time) of 2, which is the value implemented in the FlexCAN module.

49.4.8.5 Arbitration and Matching Timing

During normal reception and transmission of frames, the matching, arbitration, move-in and move-out processes are executed during certain time windows inside the CAN frame, as shown in the following figures.

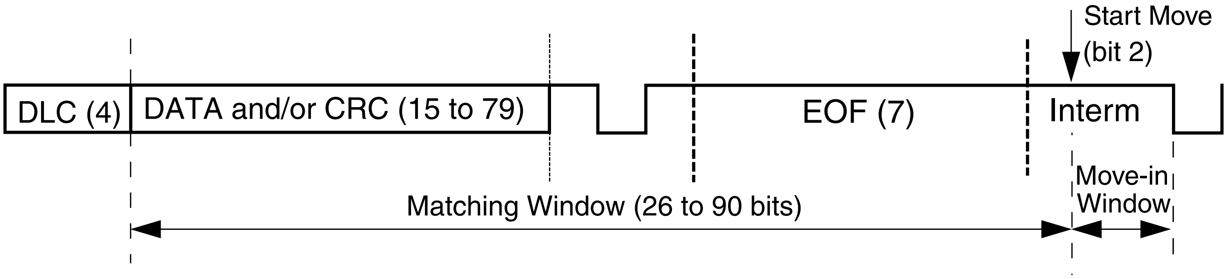


Figure 49-106. Matching and Move-In Time Windows

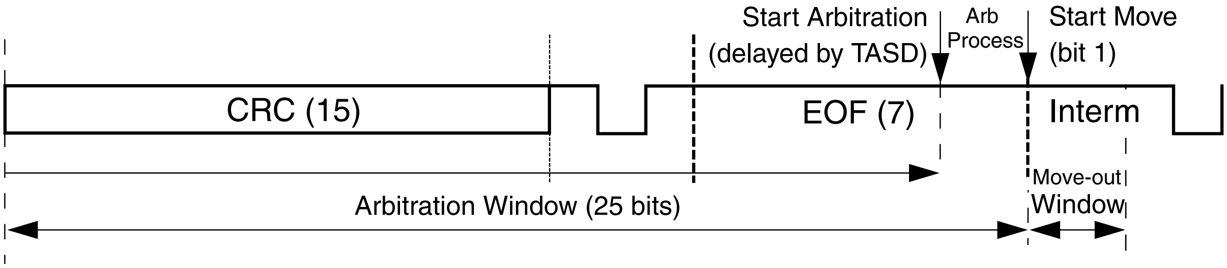


Figure 49-107. Arbitration and Move-Out Time Windows

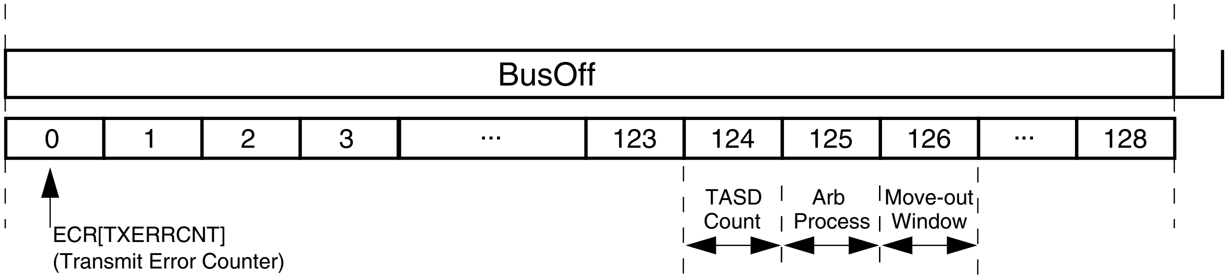


Figure 49-108. Arbitration at the end of Bus Off and Move-Out Time Windows

NOTE

The matching and arbitration timing shown in the preceding figures do not take into account the delay caused by the concurrent memory access due to the CPU or other internal peripheral.

When doing matching and arbitration, FlexCAN needs to scan the whole Message Buffer memory during the available time slot. In order to have sufficient time to do that, the following requirements must be observed:

- A valid CAN bit timing must be programmed, as indicated in Table 49-119
- The peripheral clock frequency can not be smaller than the oscillator clock frequency, i.e. the PLL can not be programmed to divide down the oscillator clock; see Clock domains and restrictions
- There must be a minimum ratio between the peripheral clock frequency and the CAN bit rate, as specified in the following table

Table 49-120. Minimum Ratio Between Peripheral Clock Frequency and CAN Bit Rate

Number of Message Buffers	RFEN	Minimum Number of Peripheral Clocks per CAN bit
16 and 32	0	16
64	0	25
16	1	16
32	1	17
64	1	30

A direct consequence of the first requirement is that the minimum number of time quanta per CAN bit must be 8, so the oscillator clock frequency should be at least 8 times the CAN bit rate. The minimum frequency ratio specified in the preceding table can be achieved by choosing a high enough peripheral clock frequency when compared to the oscillator clock frequency, or by adjusting one or more of the bit timing parameters (PRESDIV, PROPSEG, PSEG1, PSEG2) contained in the Control 1 Register (CTRL1).

In case of synchronous operation (when the peripheral clock frequency is equal to the oscillator clock frequency), the number of peripheral clocks per CAN bit can be adjusted by selecting an adequate value for PRESDIV in order to meet the requirement in the preceding table. In case of asynchronous operation (the peripheral clock frequency greater than the oscillator clock frequency), the number of peripheral clocks per CAN bit can be adjusted by both PRESDIV and/or the frequency ratio.

As an example, taking the case of 64 MBs, if the oscillator and peripheral clock frequencies are equal and the CAN bit timing is programmed to have 8 time quanta per bit, then the prescaler factor (PRESDIV + 1) should be at least 2. For prescaler factor equal to one and CAN bit timing with 8 time quanta per bit, the ratio between peripheral and oscillator clock frequencies should be at least 2.

49.4.9 Modes of Operation Details

The FlexCAN module has four functional modes (Normal Mode, Freeze Mode, Listen-Only Mode and Loop-Back Mode) and three low power modes (Disable Mode, Doze Mode and Stop Mode). See Modes of Operation for an introductory description of all these modes of operation. The following sub-sections contain functional details on Freeze mode and the low power modes.

CAUTION

> "Permanent Dominant" failure on CAN Bus line is not supported by FlexCAN. If a Low Power request or Freeze Mode request is done during a "Permanent Dominant", the corresponding acknowledge can never be asserted.

49.4.9.1 Freeze Mode

This mode is requested by the CPU through the assertion of the HALT bit in the MCR Register or when the MCU is put into Debug Mode. In both cases it is also necessary that the FRZ bit is asserted in the MCR Register and the module is not in any of the low power modes (Disable, Doze, Stop). The acknowledgement is obtained through the assertion by the FlexCAN of FRZ_ACK bit in the same register. The CPU must only consider the FlexCAN in Freeze Mode when both request and acknowledgement conditions are satisfied.

When Freeze Mode is requested during transmission or reception, FlexCAN does the following:

- Waits to be in either Intermission, Passive Error, Bus Off or Idle state
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores the Rx input pin and drives the Tx pin as recessive
- Stops the prescaler, thus halting all CAN protocol activities

- Grants write access to the Error Counters Register, which is read-only in other modes
- Sets the NOT_RDY and FRZ_ACK bits in MCR

After requesting Freeze Mode, the user must wait for the FRZ_ACK bit to be asserted in MCR before executing any other action, otherwise FlexCAN may operate in an unpredictable way. In Freeze mode, all memory mapped registers are accessible, except for CTRL1[CLK_SRC] bit that can be read but cannot be written.

Exiting Freeze Mode is done in one of the following ways:

- CPU negates the FRZ bit in the MCR Register
- The MCU is removed from Debug Mode and/or the HALT bit is negated

The FRZ_ACK bit is negated after the protocol engine recognizes the negation of the freeze request. Once out of Freeze Mode, FlexCAN tries to re-synchronize to the CAN bus by waiting for 11 consecutive recessive bits.

49.4.9.2 Module Disable Mode

This low power mode is normally used to temporarily disable a complete FlexCAN block, with no power consumption. It is requested by the CPU through the assertion of the MDIS bit in the MCR Register and the acknowledgement is obtained through the assertion by the FlexCAN of the LPM_ACK bit in the same register. The CPU must only consider the FlexCAN in Disable Mode when both request and acknowledgement conditions are satisfied.

If the module is disabled during Freeze Mode, it requests to disable the clocks to the PE and CHI sub-modules, sets the LPM_ACK bit and negates the FRZ_ACK bit. If the module is disabled during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and then checks it to be recessive
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Shuts down the clocks to the PE and CHI sub-modules
- Sets the NOTRDY and LPMACK bits in MCR

The Bus Interface Unit continues to operate, enabling the CPU to access memory mapped registers, except the Rx Mailboxes Global Mask Registers, the Rx Buffer 14 Mask Register, the Rx Buffer 15 Mask Register, the Rx FIFO Global Mask Register. The Rx FIFO Information Register, the Message Buffers, the Rx Individual Mask Registers, and the reserved words within RAM may not be accessed when the module is in Disable Mode. Exiting from this mode is done by negating the MDIS bit by the CPU, which causes the FlexCAN to request to resume the clocks and negate the LPM_ACK bit after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

49.4.9.3 Doze Mode

This is a system low power mode in which the CPU bus is kept alive and a global Doze Mode request is sent to all peripherals asking them to enter low power mode. When Doze Mode is globally requested, the DOZE bit in MCR Register needs to have been asserted previously for Doze Mode to be triggered. The acknowledgement is obtained through the assertion by the FlexCAN of the LPM_ACK bit in the same register. The CPU must only consider the FlexCAN in Doze Mode when both request and acknowledgement conditions are satisfied.

If Doze Mode is triggered during Freeze Mode, FlexCAN requests to shut down the clocks to the PE and CHI sub-modules, sets the LPM_ACK bit and negates the FRZ_ACK bit. If Doze Mode is triggered during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and checks it to be recessive
- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Shuts down the clocks to the PE and CHI sub-modules
- Sets the NOT_RDY and LPM_ACK bits in MCR

The Bus Interface Unit continues to operate, enabling the CPU to access memory mapped registers, except the Rx Mailboxes Global Mask Registers, the Rx Buffer 14 Mask Register, the Rx Buffer 15 Mask Register, the Rx FIFO Global Mask Register. The Rx FIFO Information Register, the Message Buffers, the Rx Individual Mask Registers, and the reserved words within RAM may not be accessed when the module is in Doze Mode.

Exiting Doze Mode is done in one of the following ways:

- CPU removing the Doze Mode request
- CPU negating the DOZE bit of the MCR Register
- Self Wake mechanism

In the Self Wake mechanism, if the SLF_WAK bit in MCR Register was set at the time FlexCAN entered Doze Mode, then upon detection of a recessive to dominant transition on the CAN bus, FlexCAN negates the DOZE bit, requests to resume its clocks and negates the LPM_ACK after the CAN protocol engine recognizes the negation of the Doze Mode request. It also sets the WAK_INT bit in the ESR Register and, if enabled by the WAK_MSK bit in MCR, generates a Wake Up interrupt to the CPU. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up. The following table details the effect of SLF_WAK and WAK_MSK upon wake-up from Doze Mode.

Table 49-121. Wake-up from Doze Mode

SLF_WAK	WAK_INT	WAK_MSK	FlexCAN Clocks Enabled	Wake-up Interrupt Generated
0	-	-	No	No
0	-	-	No	No
1	0	0	No	No
1	0	1	No	No
1	1	0	Yes	No
1	1	1	Yes	Yes

49.4.9.4 Stop Mode

This is a system low power mode in which all MCU clocks can be stopped for maximum power savings. The Stop Mode is globally requested by the CPU and the acknowledgement is obtained through the assertion by the FlexCAN of a Stop Acknowledgement signal. The CPU must only consider the FlexCAN in Stop Mode when both request and acknowledgement conditions are satisfied.

If FlexCAN receives the global Stop Mode request during Freeze Mode, it sets the LPM_ACK bit, negates the FRZ_ACK bit and then sends the Stop Acknowledge signal to the CPU, in order to shut down the clocks globally. If Stop Mode is requested during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and checks it to be recessive

- Waits for all internal activities like arbitration, matching, move-in and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive
- Sets the NOT_RDY and LPM_ACK bits in MCR
- Sends a Stop Acknowledge signal to the CPU, so that it can shut down the clocks globally

Exiting Stop Mode is done in one of the following ways:

- CPU resuming the clocks and removing the Stop Mode request
- CPU resuming the clocks and Stop Mode request as a result of the Self Wake mechanism

In the Self Wake mechanism, if the SLF_WAK bit in MCR Register was set at the time FlexCAN entered Stop Mode, then upon detection of a recessive to dominant transition on the CAN bus, FlexCAN sets the WAK_INT bit in the ESR Register and, if enabled by the WAK_MSK bit in MCR, generates a Wake Up interrupt to the CPU. Upon receiving the interrupt, the CPU should resume the clocks and remove the Stop Mode request. FlexCAN will then wait for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, it will not receive the frame that woke it up. The following table details the effect of SLF_WAK and WAK_MSK upon wake-up from Stop Mode. Note that wake-up from Stop Mode only works when both bits are asserted.

After the CAN protocol engine recognizes the negation of the Stop Mode request, the FlexCAN negates the LPM_ACK bit.

Table 49-122. Wake-up from Stop Mode

SLF_WAK	WAK_INT	WAK_MSK	MCU Clocks Enabled	Wake-up Interrupt Generated
0	-	-	No	No
0	-	-	No	No
1	0	0	No	No
1	0	1	No	No
1	1	0	No	No
1	1	1	Yes	Yes

49.4.10 Interrupts

Each one of the message buffers can be an interrupt source, if its corresponding IMASK bit is set. There is no distinction between Tx and Rx interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each of the buffers has assigned a flag bit in the IFLAG Registers. The bit is set when the corresponding buffer completes a successful transmission/reception and is cleared when the CPU writes it to '1' (unless another interrupt is generated at the same time).

Note

> It must be guaranteed that the CPU only clears the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.

If the Rx FIFO is enabled (bit RFEN on MCR set), the interrupts corresponding to MBs 0 to 7 have a different behavior. Bit 7 of the IFLAG1 becomes the "FIFO Overflow" flag; bit 6 becomes the FIFO Warning flag, bit 5 becomes the "Frames Available in FIFO flag" and bits 4-0 are unused. See the description of the Interrupt Flags 1 Register (IFLAG1) for more information.

A combined interrupt for all MBs is generated by an Or of all the interrupt sources from MBs. This interrupt gets generated when any of the MBs or FIFO generates an interrupt. In this case the CPU must read the IFLAG Registers to determine which MB or FIFO caused the interrupt.

The other interrupt sources (Bus Off, Error, Tx Warning, Rx Warning, and Wake Up) generate interrupts like the MB ones, and can be read from both the Error and Status Register 1 and 2. The Bus Off, Error, Tx Warning and Rx Warning interrupt mask bits are located in the Control 1 Register; the Wake-Up interrupt mask bit is located in the MCR.

49.4.11 Bus Interface

The CPU access to FlexCAN registers are subject to the following rules:

- Unrestricted read and write access to supervisor registers (registers identified with S/U in Table "Module Memory Map" in Supervisor Mode or with S only) results in access error.
- Read and write access to implemented reserved address space results in access error.

- Write access to positions whose bits are all currently read-only results in access error. If at least one of the bits is not read-only then no access error is issued. Write permission to positions or some of their bits can change depending on the mode of operation or transitory state. Refer to register and bit descriptions for details.
- Read and write access to unimplemented address space results in access error.
- Read and write access to RAM located positions during Low Power Mode results in access error.
- If MAXMB is programmed with a value smaller than the available number of MBs, then the unused memory space can be used as general purpose RAM space. Note that reserved words within RAM cannot be used. As an example, suppose FlexCAN is configured with 16 MBs, RFFN is 0x0, and MAXMB is programmed with zero. The maximum number of MBs in this case becomes one. The RAM starts at 0x0080, and the space from 0x0080 to 0x008F is used by the one MB. The memory space from 0x0090 to 0x017F is available. The space between 0x0180 and 0x087F is reserved. The space from 0x0880 to 0x0883 is used by the one Individual Mask and the available memory in the Mask Registers space would be from 0x0884 to 0x08BF. From 0x08C0 through 0x09DF there are reserved words for internal use which cannot be used as general purpose RAM. As a general rule, free memory space for general purpose depends only on MAXMB.

Note

> Unused MB space must not be used as general purpose RAM while FlexCAN is transmitting and receiving CAN frames.

49.5 Initialization/Application Information

This section provide instructions for initializing the FlexCAN module.

49.5.1 FlexCAN Initialization Sequence

The FlexCAN module may be reset in three ways:

- MCU level hard reset, which resets all memory mapped registers asynchronously
- MCU level soft reset, which resets some of the memory mapped registers synchronously (refer to Table 49-2 to see what registers are affected by soft reset)
- SOFT_RST bit in MCR, which has the same effect as the MCU level soft reset

Soft reset is synchronous and has to follow an internal request/acknowledge procedure across clock domains. Therefore, it may take some time to fully propagate its effects. The SOFT_RST bit remains asserted while soft reset is pending, so software can poll this bit to know when the reset has completed. Also, soft reset can not be applied while clocks are shut down in any of the low power modes. The low power mode should be exited and the clocks resumed before applying soft reset.

The clock source (CLK_SRC bit) should be selected while the module is in Disable Mode. After the clock source is selected and the module is enabled (MDIS bit negated), FlexCAN automatically goes to Freeze Mode. In Freeze Mode, FlexCAN is un-synchronized to the CAN bus, the HALT and FRZ bits in MCR Register are set, the internal state machines are disabled and the FRZ_ACK and NOT_RDY bits in the MCR Register are set. The Tx pin is in recessive state and FlexCAN does not initiate any transmission or reception of CAN frames. Note that the Message Buffers and the Rx Individual Mask Registers are not affected by reset, so they are not automatically initialized.

For any configuration change/initialization it is required that FlexCAN is put into Freeze Mode (see Freeze Mode). The following is a generic initialization sequence applicable to the FlexCAN module:

- Initialize the Module Configuration Register
  - Enable the individual filtering per MB and reception queue features by setting the IRMQ bit
  - Enable the warning interrupts by setting the WRN_EN bit
  - If required, disable frame self reception by setting the SRX_DIS bit
  - Enable the Rx FIFO by setting the RFEN bit
  - Enable the abort mechanism by setting the AEN bit
  - Enable the local priority feature by setting the LPRIO_EN bit
- Initialize the Control Register
  - Determine the bit timing parameters: PROPSEG, PSEG1, PSEG2, RJW
  - Determine the bit rate by programming the PRESDIV field
  - Determine the internal arbitration mode (LBUF bit)
- Initialize the Message Buffers
  - The Control and Status word of all Message Buffers must be initialized

    - If Rx FIFO was enabled, the ID filter table must be initialized
    - Other entries in each Message Buffer should be initialized as required
- Initialize the Rx Individual Mask Registers
- Set required interrupt mask bits in the IMASK Registers (for all MB interrupts), in CTRL Register (for Bus Off and Error interrupts) and in MCR Register for Wake-Up interrupt
- Negate the HALT bit in MCR

Starting with the last event, FlexCAN attempts to synchronize to the CAN bus.

Chapter 50
SPI (DSPI)

50.1 Introduction

NOTE

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

The serial peripheral interface module provides a synchronous serial bus for communication between an MCU and an external peripheral device.

50.1.1 Block Diagram

The block diagram of this module is as follows:

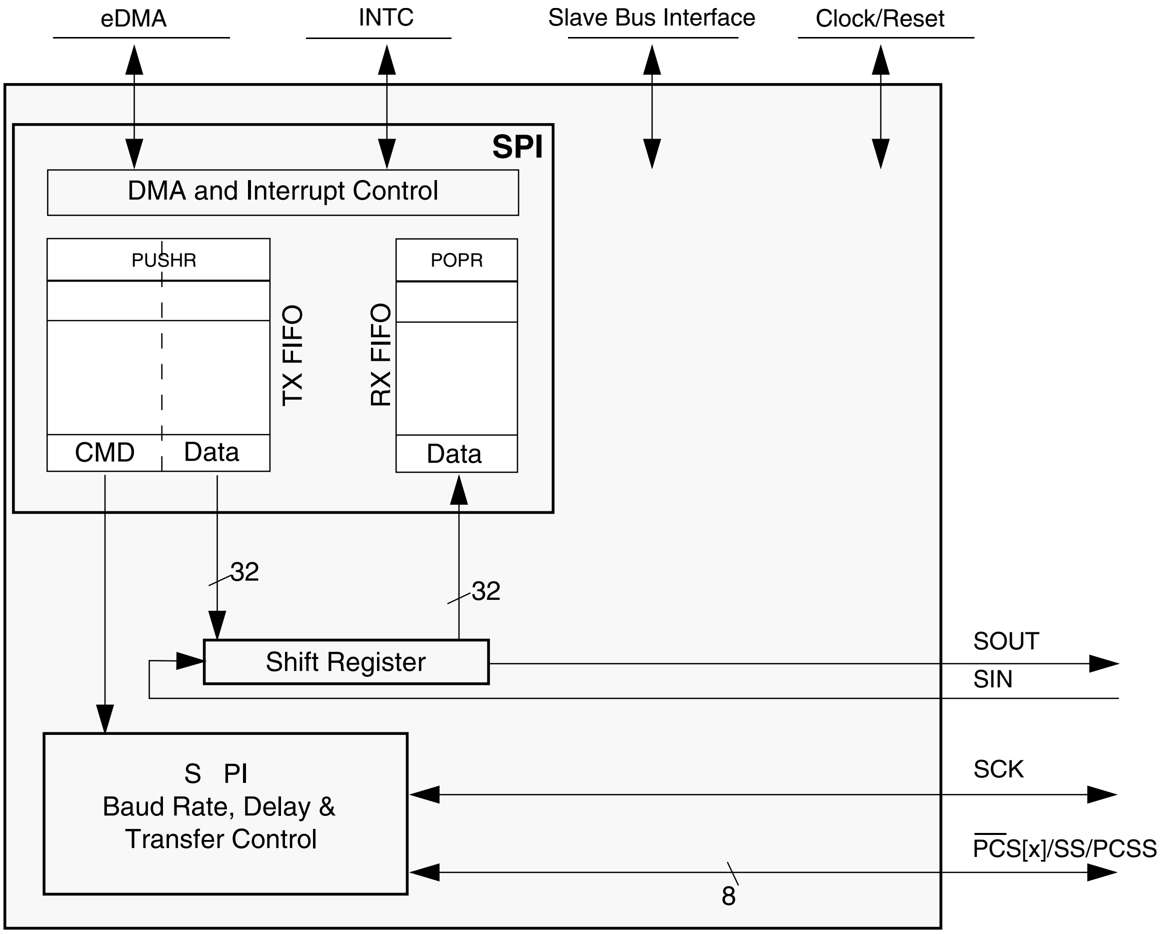


Figure 50-1. DSPI Block Diagram

50.1.2 Features

The DSPI supports these SPI features:

- Full-duplex, Four-wire synchronous transfers
- Master and slave modes
    - Data streaming operation in slave mode with continuous slave selection
- Buffered transmit operation using the TX FIFO with depth of 4 entries
- Buffered receive operation using the RX FIFO with depth of 4 entries
- Asynchronous clocking scheme for Register and Protocol Interfaces
- TX and RX FIFOs can be disabled individually for low-latency updates to SPI queues

- Visibility into TX and RX FIFOs for ease of debugging
- Programmable transfer attributes on a per-frame basis:
    - 2 transfer attribute registers
    - Serial clock with programmable polarity and phase
    - Various programmable delays
    - Programmable serial frame size of 4 to 16 bits, expandable by software control
        - SPI frames longer than 16 bits can be supported using the continuous selection format.
    - Continuously held chip select capability
- 6 Peripheral Chip Selects, expandable to 64 with external demultiplexer
- Deglitching support for up to 32 Peripheral Chip Select with external demultiplexer
- DMA support for adding entries to TX FIFO and removing entries from RX FIFO:
    - TX FIFO is not full (TFFF)
    - RX FIFO is not empty (RFDF)
- Interrupt conditions:
    - End of queue reached (EOQF)
    - TX FIFO is not full (TFFF)
    - Transfer of current frame complete (TCF)
    - Attempt to transmit with an empty Transmit FIFO (TFUF)
    - RX FIFO is not empty (RFDF)
    - Frame received while Receive FIFO is full (RFOF)
- Global interrupt request line
- Modified SPI transfer formats for communication with slower peripheral devices
- Power-saving architectural features
    - Support for stop mode
    - Support for doze mode

50.1.3 DSPI Configurations

The DSPI module always operates in SPI configuration.

50.1.3.1 SPI Configuration

The SPI configuration allows the DSPI to send and receive serial data. This configuration allows the DSPI to operate as a basic SPI block with internal FIFOs supporting external queues operation. Transmit data and received data reside in separate FIFOs. The host CPU or a DMA controller read the received data from the receive FIFO and write transmit data to the transmit FIFO.

For queued operations the SPI queues can reside in system RAM, external to the DSPI. Data transfers between the queues and the DSPI FIFOs are accomplished by a DMA controller or host CPU. The following figure shows a system example with DMA, DSPI and external queues in system RAM.

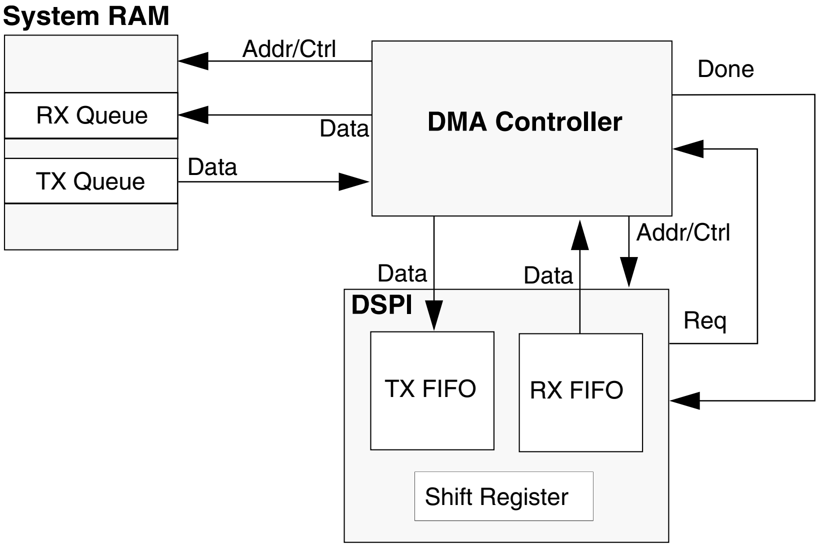


Figure 50-2. DSPI with Queues and DMA

50.1.4 Modes of Operation

The DSPI supports the following modes of operation that can be divided into two categories;

- Module-specific modes:
    - Master mode

    - Slave mode
    - Module disable mode
- MCU-specific modes:
    - External stop mode
    - Debug mode

The DSPI enters module-specific modes when the host writes a DSPI register. The MCU-specific modes are controlled by signals, external to the DSPI. The MCU-specific modes are modes that an MCU may enter in parallel to the DSPI block-specific modes.

50.1.4.1 Master Mode

Master mode allows the DSPI to initiate and control serial communication. In this mode, the SCK signal and the PCS[*x*] signals are controlled by the DSPI and configured as outputs.

50.1.4.2 Slave Mode

Slave mode allows the DSPI to communicate with SPI bus masters. In this mode, the DSPI responds to externally controlled serial transfers. The SCK signal and the PCS[0]/$\overline{\text{SS}}$ signals are configured as inputs and driven by a SPI bus master.

50.1.4.3 Module Disable Mode

The module disable mode can be used for MCU power management. The clock to the non-memory mapped logic in the DSPI can be stopped while in the module disable mode.

50.1.4.4 External Stop Mode

External stop mode is used for MCU power management. The DSPI supports the Peripheral Bus stop mode mechanism. When a request is made to enter external stop mode, the DSPI block acknowledges the request and completes the transfer in progress. When the DSPI reaches the frame boundary it signals that the system clock to the DSPI module may be shut off.

50.1.4.5 Debug Mode

Debug mode is used for system development and debugging. The MCR[FRZ] bit controls DSPI behavior in the debug mode. If the bit is set, the DSPI stops all serial transfers, when the MCU is in debug mode. If the bit is cleared, the MCU debug mode has no effect on the DSPI.

50.2 DSPI Signal Descriptions

This section provides the DSPI signals description.

The following table lists the signals that may connect off chip depending on device implementation.

Table 50-1. DSPI Signal Descriptions

Signal	Description	I/O
PCS0/$\overline{\text{SS}}$	Master mode: Peripheral Chip Select 0 output Slave mode: Slave Select input	I/O
PCS[3:1]	Master mode: Peripheral Chip Select 1 - 3 Slave mode: Unused	O
PCS4	Master mode: Peripheral Chip Select 4 Slave mode: Unused	O
PCS5/ $\overline{\text{PCSS}}$	Master mode: Peripheral Chip Select 5 / Peripheral Chip Select Strobe Slave mode: Unused	O
SIN	Serial Data In	I
SOUT	Serial Data Out	O
SCK	Master mode: Serial Clock (output) Slave mode: Serial Clock (input)	I/O

50.2.1 PCS0/$\overline{\text{SS}}$ — Peripheral Chip Select/Slave Select

In master mode, the PCS0 signal is a Peripheral Chip Select output that selects which slave device the current transmission is intended for.

In slave mode, the active low $\overline{\text{SS}}$ signal is a Slave Select input signal that allows a SPI master to select the DSPI as the target for transmission.

50.2.2 PCS1 - PCS3 — Peripheral Chip Selects 1 - 3

PCS1 - PCS3 are Peripheral Chip Select output signals in master mode.

In slave mode, these signals are unused.

50.2.3 PCS4 — Peripheral Chip Select 4

In master mode, PCS4 is a Peripheral Chip Select output signal.

In slave mode, this signal is unused.

50.2.4 PCS5/$\overline{\text{PCSS}}$ — Peripheral Chip Select 5/Peripheral Chip Select Strobe

PCS5 is a Peripheral Chip Select output signal. When the DSPI is in master mode and the MCR[PCSSE] bit is cleared, this signal selects which slave device the current transfer is intended for.

When the DSPI is in master mode and the MCR[PCSSE] bit is set, the $\overline{\text{PCSS}}$ signal acts as a strobe to an external peripheral chip select demultiplexer, which decodes the PCS0 - PCS4 signals, preventing glitches on the demultiplexer outputs.

This signal is not used in slave mode.

50.2.5 SIN — Serial Input

SIN is a serial data input signal.

50.2.6 SOUT — Serial Output

SOUT is a serial data output signal.

50.2.7 SCK — Serial Clock

SCK is a serial communication clock signal. In master mode, the DSPI generates the SCK. In slave mode, SCK is an input from an external bus master.

50.3 Memory Map/Register Definition

Register accesses to memory addresses that are reserved or undefined result in a transfer error. Write access to the POPR register also results in a transfer error.

SPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_C000	DSPI Module Configuration Register (SPI0_MCR)	32	R/W	0000_4001h	50.3.1/ 1527
4002_C008	DSPI Transfer Count Register (SPI0_TCR)	32	R/W	0000_0000h	50.3.2/ 1530
4002_C00C	DSPI Clock and Transfer Attributes Register (In Master Mode) (SPI0_CTAR0)	32	R/W	7800_0000h	50.3.3/ 1530
4002_C00C	DSPI Clock and Transfer Attributes Register (In Slave Mode) (SPI0_CTAR0_SLAVE)	32	R/W	7800_0000h	50.3.4/ 1535
4002_C010	DSPI Clock and Transfer Attributes Register (In Master Mode) (SPI0_CTAR1)	32	R/W	7800_0000h	50.3.3/ 1530
4002_C02C	DSPI Status Register (SPI0_SR)	32	R/W	See section	50.3.5/ 1536
4002_C030	DSPI DMA/Interrupt Request Select and Enable Register (SPI0_RSER)	32	R/W	0000_0000h	50.3.6/ 1539
4002_C034	DSPI PUSH TX FIFO Register In Master Mode (SPI0_PUSHR)	32	R/W	0000_0000h	50.3.7/ 1541
4002_C034	DSPI PUSH TX FIFO Register In Slave Mode (SPI0_PUSHR_SLAVE)	32	R/W	0000_0000h	50.3.8/ 1543
4002_C038	DSPI POP RX FIFO Register (SPI0_POPR)	32	R	0000_0000h	50.3.9/ 1543
4002_C03C	DSPI Transmit FIFO Registers (SPI0_TXFR0)	32	R	0000_0000h	50.3.10/ 1544
4002_C040	DSPI Transmit FIFO Registers (SPI0_TXFR1)	32	R	0000_0000h	50.3.10/ 1544
4002_C044	DSPI Transmit FIFO Registers (SPI0_TXFR2)	32	R	0000_0000h	50.3.10/ 1544
4002_C048	DSPI Transmit FIFO Registers (SPI0_TXFR3)	32	R	0000_0000h	50.3.10/ 1544
4002_C07C	DSPI Receive FIFO Registers (SPI0_RXFR0)	32	R	0000_0000h	50.3.11/ 1545
4002_C080	DSPI Receive FIFO Registers (SPI0_RXFR1)	32	R	0000_0000h	50.3.11/ 1545

Table continues on the next page...

SPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_C084	DSPI Receive FIFO Registers (SPI0_RXFR2)	32	R	0000_0000h	50.3.11/ 1545
4002_C088	DSPI Receive FIFO Registers (SPI0_RXFR3)	32	R	0000_0000h	50.3.11/ 1545
4002_D000	DSPI Module Configuration Register (SPI1_MCR)	32	R/W	0000_4001h	50.3.1/ 1527
4002_D008	DSPI Transfer Count Register (SPI1_TCR)	32	R/W	0000_0000h	50.3.2/ 1530
4002_D00C	DSPI Clock and Transfer Attributes Register (In Master Mode) (SPI1_CTAR0)	32	R/W	7800_0000h	50.3.3/ 1530
4002_D00C	DSPI Clock and Transfer Attributes Register (In Slave Mode) (SPI1_CTAR0_SLAVE)	32	R/W	7800_0000h	50.3.4/ 1535
4002_D010	DSPI Clock and Transfer Attributes Register (In Master Mode) (SPI1_CTAR1)	32	R/W	7800_0000h	50.3.3/ 1530
4002_D02C	DSPI Status Register (SPI1_SR)	32	R/W	See section	50.3.5/ 1536
4002_D030	DSPI DMA/Interrupt Request Select and Enable Register (SPI1_RSER)	32	R/W	0000_0000h	50.3.6/ 1539
4002_D034	DSPI PUSH TX FIFO Register In Master Mode (SPI1_PUSHR)	32	R/W	0000_0000h	50.3.7/ 1541
4002_D034	DSPI PUSH TX FIFO Register In Slave Mode (SPI1_PUSHR_SLAVE)	32	R/W	0000_0000h	50.3.8/ 1543
4002_D038	DSPI POP RX FIFO Register (SPI1_POPR)	32	R	0000_0000h	50.3.9/ 1543
4002_D03C	DSPI Transmit FIFO Registers (SPI1_TXFR0)	32	R	0000_0000h	50.3.10/ 1544
4002_D040	DSPI Transmit FIFO Registers (SPI1_TXFR1)	32	R	0000_0000h	50.3.10/ 1544
4002_D044	DSPI Transmit FIFO Registers (SPI1_TXFR2)	32	R	0000_0000h	50.3.10/ 1544
4002_D048	DSPI Transmit FIFO Registers (SPI1_TXFR3)	32	R	0000_0000h	50.3.10/ 1544
4002_D07C	DSPI Receive FIFO Registers (SPI1_RXFR0)	32	R	0000_0000h	50.3.11/ 1545
4002_D080	DSPI Receive FIFO Registers (SPI1_RXFR1)	32	R	0000_0000h	50.3.11/ 1545
4002_D084	DSPI Receive FIFO Registers (SPI1_RXFR2)	32	R	0000_0000h	50.3.11/ 1545
4002_D088	DSPI Receive FIFO Registers (SPI1_RXFR3)	32	R	0000_0000h	50.3.11/ 1545

SPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400A_C000	DSPI Module Configuration Register (SPI2_MCR)	32	R/W	0000_4001h	50.3.1/ 1527
400A_C008	DSPI Transfer Count Register (SPI2_TCR)	32	R/W	0000_0000h	50.3.2/ 1530
400A_C00C	DSPI Clock and Transfer Attributes Register (In Master Mode) (SPI2_CTAR0)	32	R/W	7800_0000h	50.3.3/ 1530
400A_C00C	DSPI Clock and Transfer Attributes Register (In Slave Mode) (SPI2_CTAR0_SLAVE)	32	R/W	7800_0000h	50.3.4/ 1535
400A_C010	DSPI Clock and Transfer Attributes Register (In Master Mode) (SPI2_CTAR1)	32	R/W	7800_0000h	50.3.3/ 1530
400A_C02C	DSPI Status Register (SPI2_SR)	32	R/W	See section	50.3.5/ 1536
400A_C030	DSPI DMA/Interrupt Request Select and Enable Register (SPI2_RSER)	32	R/W	0000_0000h	50.3.6/ 1539
400A_C034	DSPI PUSH TX FIFO Register In Master Mode (SPI2_PUSHR)	32	R/W	0000_0000h	50.3.7/ 1541
400A_C034	DSPI PUSH TX FIFO Register In Slave Mode (SPI2_PUSHR_SLAVE)	32	R/W	0000_0000h	50.3.8/ 1543
400A_C038	DSPI POP RX FIFO Register (SPI2_POPR)	32	R	0000_0000h	50.3.9/ 1543
400A_C03C	DSPI Transmit FIFO Registers (SPI2_TXFR0)	32	R	0000_0000h	50.3.10/ 1544
400A_C040	DSPI Transmit FIFO Registers (SPI2_TXFR1)	32	R	0000_0000h	50.3.10/ 1544
400A_C044	DSPI Transmit FIFO Registers (SPI2_TXFR2)	32	R	0000_0000h	50.3.10/ 1544
400A_C048	DSPI Transmit FIFO Registers (SPI2_TXFR3)	32	R	0000_0000h	50.3.10/ 1544
400A_C07C	DSPI Receive FIFO Registers (SPI2_RXFR0)	32	R	0000_0000h	50.3.11/ 1545
400A_C080	DSPI Receive FIFO Registers (SPI2_RXFR1)	32	R	0000_0000h	50.3.11/ 1545
400A_C084	DSPI Receive FIFO Registers (SPI2_RXFR2)	32	R	0000_0000h	50.3.11/ 1545
400A_C088	DSPI Receive FIFO Registers (SPI2_RXFR3)	32	R	0000_0000h	50.3.11/ 1545

50.3.1 DSPI Module Configuration Register (SPIx_MCR)

Contains bits to configure various attributes associated with DSPI operations. The HALT and MDIS bits can be changed at any time, but they only take effect on the next frame boundary. Only the HALT and MDIS bits in the MCR can be changed, while the DSPI is in the Running state.

Addresses: SPI0_MCR is 4002_C000h base + 0h offset = 4002_C000h
SPI1_MCR is 4002_D000h base + 0h offset = 4002_D000h
SPI2_MCR is 400A_C000h base + 0h offset = 400A_C000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MSTR	CONT_SCKE	DCONF		FRZ	MTFE	PCSSE	ROOE	0		PCSIS[5:0]					
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	DOZE	MDIS	DIS_TXF	DIS_RXF	0	0	SMPL_PT		0						0	HALT
W					CLR_TXF	CLR_RXF										
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1

SPIx_MCR field descriptions

Field	Description
31 MSTR	Master/Slave Mode Select Configures the DSPI for either master mode or slave mode. 0 DSPI is in slave mode. 1 DSPI is in master mode.
30 CONT_SCKE	Continuous SCK Enable Enables the Serial Communication Clock (SCK) to run continuously. 0 Continuous SCK disabled. 1 Continuous SCK enabled.
29–28 DCONF	DSPI Configuration Selects among the different configurations of the DSPI. 00 SPI 01 Reserved

Table continues on the next page...

SPIx_MCR field descriptions (continued)

Field	Description
	10 Reserved 11 Reserved
27 FRZ	Freeze Enables the DSPI transfers to be stopped on the next frame boundary when the device enters Debug mode. 0 Do not halt serial transfers in debug mode. 1 Halt serial transfers in debug mode.
26 MTFE	Modified Timing Format Enable Enables a modified transfer format to be used. 0 Modified SPI transfer format disabled. 1 Modified SPI transfer format enabled.
25 PCSSE	Peripheral Chip Select Strobe Enable Enables the PCS[5]/ $\overline{\text{PCSS}}$ to operate as a PCS Strobe output signal. 0 PCS[5]/$\overline{\text{PCSS}}$ is used as the Peripheral Chip Select[5] signal. 1 PCS[5]/PCSS is used as an active-low PCS Strobe signal.
24 ROOE	Receive FIFO Overflow Overwrite Enable In the RX FIFO overflow condition, configures the DSPI to ignore the incoming serial data or overwrite existing data. If the RX FIFO is full and new data is received, the data from the transfer, generating the overflow, is ignored or shifted into the shift register. 0 Incoming data is ignored. 1 Incoming data is shifted into the shift register.
23–22 Reserved	This read-only field is reserved and always has the value zero.
21–16 PCSIS[5:0]	Peripheral Chip Select x Inactive State Determines the inactive state of PCSx. 0 The inactive state of PCSx is low. 1 The inactive state of PCSx is high.
15 DOZE	Doze Enable Provides support for an externally controlled Doze mode power-saving mechanism. 0 Doze mode has no effect on DSPI. 1 Doze mode disables DSPI.
14 MDIS	Module Disable Allows the clock to be stopped to the non-memory mapped logic in the DSPI effectively putting the DSPI in a software controlled power-saving state. The reset value of the MDIS bit is parameterized, with a default reset value of "0".

Table continues on the next page...

SPIx_MCR field descriptions (continued)

Field	Description
	0 Enable DSPI clocks. 1 Allow external logic to disable DSPI clocks.
13 DIS_TXF	Disable Transmit FIFO When the TX FIFO is disabled, the transmit part of the DSPI operates as a simplified double-buffered SPI. This bit can only be written when the MDIS bit is cleared. 0 Tx FIFO is enabled. 1 Tx FIFO is disabled.
12 DIS_RXF	Disable Receive FIFO When the RX FIFO is disabled, the receive part of the DSPI operates as a simplified double-buffered SPI. This bit can only be written when the MDIS bit is cleared. 0 Rx FIFO is enabled. 1 Rx FIFO is disabled.
11 CLR_TXF	Clear TX FIFO Flushes the TX FIFO. Writing a 1 to CLR_TXF clears the TX FIFO Counter. The CLR_TXF bit is always read as zero. 0 Do not clear the Tx FIFO counter. 1 Clear the Tx FIFO counter.
10 CLR_RXF	Flushes the RX FIFO. Writing a 1 to CLR_RXF clears the RX Counter. The CLR_RXF bit is always read as zero. 0 Do not clear the Rx FIFO counter. 1 Clear the Rx FIFO counter.
9–8 SMPL_PT	Sample Point Controls when the DSPI master samples SIN in Modified Transfer Format. This field is valid only when CPHA bit in CTAR register is 0. 00 0 system clocks between SCK edge and SIN sample 01 1 system clock between SCK edge and SIN sample 10 2 system clocks between SCK edge and SIN sample 11 Reserved
7–2 Reserved	This read-only field is reserved and always has the value zero.
1 Reserved	This read-only field is reserved and always has the value zero.
0 HALT	Halt Starts and stops DSPI transfers. 0 Start transfers. 1 Stop transfers.

50.3.2 DSPI Transfer Count Register (SPIx_TCR)

TCR contains a counter that indicates the number of SPI transfers made. The transfer counter is intended to assist in queue management. Do not write the TCR when the DSPI is in the Running state.

Addresses: SPI0_TCR is 4002_C000h base + 8h offset = 4002_C008h
SPI1_TCR is 4002_D000h base + 8h offset = 4002_D008h
SPI2_TCR is 400A_C000h base + 8h offset = 400A_C008h

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	SPI_TCNT																0															
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

SPIx_TCR field descriptions

Field	Description
31–16 SPI_TCNT	SPI Transfer Counter Counts the number of SPI transfers the DSPI makes. The SPI_TCNT field increments every time the last bit of a SPI frame is transmitted. A value written to SPI_TCNT presets the counter to that value. SPI_TCNT is reset to zero at the beginning of the frame when the CTCNT field is set in the executing SPI command. The Transfer Counter wraps around; incrementing the counter past 65535 resets the counter to zero.
15–0 Reserved	This read-only field is reserved and always has the value zero.

50.3.3 DSPI Clock and Transfer Attributes Register (In Master Mode) (SPIx_CTAR*n*)

CTAR registers are used to define different transfer attributes. The number of CTAR registers is parameterized in the RTL and can be from two to eight registers. Do not write to the CTAR registers while the DSPI is in the Running state.

In master mode, the CTAR registers define combinations of transfer attributes such as frame size, clock phase and polarity, data bit ordering, baud rate, and various delays. In slave mode, a subset of the bitfields in CTAR0 are used to set the slave transfer attributes.

When the DSPI is configured as an SPI master, the CTAS field in the command portion of the TX FIFO entry selects which of the CTAR register is used. When the DSPI is configured as an SPI bus slave, the CTAR0 register is used.

Addresses: SPI0_CTAR0 is 4002_C000h base + Ch offset = 4002_C00Ch
SPI0_CTAR1 is 4002_C000h base + 10h offset = 4002_C010h
SPI1_CTAR0 is 4002_D000h base + Ch offset = 4002_D00Ch
SPI1_CTAR1 is 4002_D000h base + 10h offset = 4002_D010h
SPI2_CTAR0 is 400A_C000h base + Ch offset = 400A_C00Ch
SPI2_CTAR1 is 400A_C000h base + 10h offset = 400A_C010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DBR	FMSZ				CPOL	CPHA	LSBFE	PCSSCK		PASC		PDT		PBR	
Reset	0	1	1	1	1	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	CSSCK				ASC				DT				BR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPIx_CTAR*n* field descriptions

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>31
DBR</td><td>Double Baud Rate

Doubles the effective baud rate of the Serial Communications Clock (SCK). This field is used only in master mode. It effectively halves the Baud Rate division ratio, supporting faster frequencies, and odd division ratios for the Serial Communications Clock (SCK). When the DBR bit is set, the duty cycle of the Serial Communications Clock (SCK) depends on the value in the Baud Rate Prescaler and the Clock Phase bit as listed in the following table. See the BR field description for details on how to compute the baud rate.

<b>Table 50-32. DSPI SCK Duty Cycle</b>
<table>
<tr><th>DBR</th><th>CPHA</th><th>PBR</th><th>SCK Duty Cycle</th></tr>
<tr><td>0</td><td>any</td><td>any</td><td>50/50</td></tr>
<tr><td>1</td><td>0</td><td>00</td><td>50/50</td></tr>
<tr><td>1</td><td>0</td><td>01</td><td>33/66</td></tr>
<tr><td>1</td><td>0</td><td>10</td><td>40/60</td></tr>
<tr><td>1</td><td>0</td><td>11</td><td>43/57</td></tr>
<tr><td>1</td><td>1</td><td>00</td><td>50/50</td></tr>
<tr><td>1</td><td>1</td><td>01</td><td>66/33</td></tr>
<tr><td>1</td><td>1</td><td>10</td><td>60/40</td></tr>
<tr><td>1</td><td>1</td><td>11</td><td>57/43</td></tr>
</table>
0 The baud rate is computed normally with a 50/50 duty cycle.
1 The baud rate is doubled with the duty cycle depending on the Baud Rate Prescaler.</td></tr>
</table>

Table continues on the next page...

SPIx_CTAR*n* field descriptions (continued)

Field	Description
30–27 FMSZ	Frame Size The number of bits transferred per frame is equal to the FMSZ field value plus 1. The minimum valid FMSZ field value is 3.
26 CPOL	Clock Polarity Selects the inactive state of the Serial Communications Clock (SCK). This bit is used in both master and slave mode. For successful communication between serial devices, the devices must have identical clock polarities. When the Continuous Selection Format is selected, switching between clock polarities without stopping the DSPI can cause errors in the transfer due to the peripheral device interpreting the switch of clock polarity as a valid clock edge. 0 The inactive state value of SCK is low. 1 The inactive state value of SCK is high.
25 CPHA	Clock Phase Selects which edge of SCK causes data to change and which edge causes data to be captured. This bit is used in both master and slave mode. For successful communication between serial devices, the devices must have identical clock phase settings. In Continuous SCK mode, the bit value is ignored and the transfers are done as if the CPHA bit is set to 1. 0 Data is captured on the leading edge of SCK and changed on the following edge. 1 Data is changed on the leading edge of SCK and captured on the following edge.
24 LSBFE	LBS First Specifies whether the LSB or MSB of the frame is transferred first. 0 Data is transferred MSB first. 1 Data is transferred LSB first.
23–22 PCSSCK	PCS to SCK Delay Prescaler Selects the prescaler value for the delay between assertion of PCS and the first edge of the SCK. See the CSSCK field description for information on how to compute the PCS to SCK Delay. Refer PCS to SCK Delay (t_{CSC}) for more details. 00 PCS to SCK Prescaler value is 1. 01 PCS to SCK Prescaler value is 3. 10 PCS to SCK Prescaler value is 5. 11 PCS to SCK Prescaler value is 7.
21–20 PASC	After SCK Delay Prescaler Selects the prescaler value for the delay between the last edge of SCK and the negation of PCS. See the ASC field description for information on how to compute the After SCK Delay.Refer After SCK Delay (t_{ASC}) for more details. 00 Delay after Transfer Prescaler value is 1. 01 Delay after Transfer Prescaler value is 3. 10 Delay after Transfer Prescaler value is 5. 11 Delay after Transfer Prescaler value is 7.
19–18 PDT	Delay after Transfer Prescaler

Table continues on the next page...

SPIx_CTAR*n* field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>Selects the prescaler value for the delay between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame. The PDT field is only used in master mode. See the DT field description for details on how to compute the Delay after Transfer. Refer Delay after Transfer (t_{DT}) for more details.
00 Delay after Transfer Prescaler value is 1.
01 Delay after Transfer Prescaler value is 3.
10 Delay after Transfer Prescaler value is 5.
11 Delay after Transfer Prescaler value is 7.</td></tr>
<tr><td>17–16
PBR</td><td>Baud Rate Prescaler
Selects the prescaler value for the baud rate. This field is used only in master mode. The baud rate is the frequency of the SCK. The system clock is divided by the prescaler value before the baud rate selection takes place. See the BR field description for details on how to compute the baud rate.
00 Baud Rate Prescaler value is 2.
01 Baud Rate Prescaler value is 3.
10 Baud Rate Prescaler value is 5.
11 Baud Rate Prescaler value is 7.</td></tr>
<tr><td>15–12
CSSCK</td><td>PCS to SCK Delay Scaler
Selects the scaler value for the PCS to SCK delay. This field is used only in master mode. The PCS to SCK Delay is the delay between the assertion of PCS and the first edge of the SCK. The delay is a multiple of the system clock period, and it is computed according to the following equation:
t_{CSC} = (1/f_{SYS}) x PCSSCK x CSSCK
The following table lists the delay scaler values.
<b>Table 50-33. Delay Scaler Encoding</b>
<table>
<tr><th>Field Value</th><th>Delay Scaler Value</th></tr>
<tr><td>0000</td><td>2</td></tr>
<tr><td>0001</td><td>4</td></tr>
<tr><td>0010</td><td>8</td></tr>
<tr><td>0011</td><td>16</td></tr>
<tr><td>0100</td><td>32</td></tr>
<tr><td>0101</td><td>64</td></tr>
<tr><td>0110</td><td>128</td></tr>
<tr><td>0111</td><td>256</td></tr>
<tr><td>1000</td><td>512</td></tr>
<tr><td>1001</td><td>1024</td></tr>
<tr><td>1010</td><td>2048</td></tr>
<tr><td>1011</td><td>4096</td></tr>
<tr><td>1100</td><td>8192</td></tr>
<tr><td>1101</td><td>16384</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

SPIx_CTARn field descriptions (continued)

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>

Table 50-33. Delay Scaler Encoding (continued)

<table>
<tr><th>Field Value</th><th>Delay Scaler Value</th></tr>
<tr><td>1110</td><td>32768</td></tr>
<tr><td>1111</td><td>65536</td></tr>
</table>

Refer PCS to SCK Delay (t_{CSC}) for more details.
</td></tr>
<tr><td>11–8
ASC</td><td>

After SCK Delay Scaler

Selects the scaler value for the After SCK Delay. This field is used only in master mode. The After SCK Delay is the delay between the last edge of SCK and the negation of PCS. The delay is a multiple of the system clock period, and it is computed according to the following equation:

t_{ASC} = (1/f_{SYS}) x PASC x ASC

See Delay Scaler Encoding table in CTARn[CSSCK] bit field description for scaler values. Refer After SCK Delay (t_{ASC}) for more details.
</td></tr>
<tr><td>7–4
DT</td><td>

Delay After Transfer Scaler

Selects the Delay after Transfer Scaler. This field is used only in master mode. The Delay after Transfer is the time between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame.

In the Continuous Serial Communications Clock operation, the DT value is fixed to one SCK clock period, The Delay after Transfer is a multiple of the system clock period, and it is computed according to the following equation:

t_{DT} = (1/f_{SYS}) x PDT x DT

See Delay Scaler Encoding table in CTARn[CSSCK] bit field description for scaler values.
</td></tr>
<tr><td>3–0
BR</td><td>

Baud Rate Scaler

Selects the scaler value for the baud rate. This field is used only in master mode. The prescaled system clock is divided by the Baud Rate Scaler to generate the frequency of the SCK. The baud rate is computed according to the following equation:

SCK baud rate = (f_{SYS}/PBR) x [(1+DBR)/BR]

The following table lists the baud rate scaler values.

Table 50-34. DSPI Baud Rate Scaler

<table>
<tr><th>CTARn[BR]</th><th>Baud Rate Scaler Value</th></tr>
<tr><td>0000</td><td>2</td></tr>
<tr><td>0001</td><td>4</td></tr>
<tr><td>0010</td><td>6</td></tr>
<tr><td>0011</td><td>8</td></tr>
<tr><td>0100</td><td>16</td></tr>
<tr><td>0101</td><td>32</td></tr>
<tr><td>0110</td><td>64</td></tr>
<tr><td>0111</td><td>128</td></tr>
</table>
</td></tr>
</table>

Table continues on the next page...

SPIx_CTAR*n* field descriptions (continued)

Field	Description
	Table 50-34. DSPI Baud Rate Scaler (continued)

CTARn[BR]	Baud Rate Scaler Value
1000	256
1001	512
1010	1024
1011	2048
1100	4096
1101	8192
1110	16384
1111	32768

50.3.4 DSPI Clock and Transfer Attributes Register (In Slave Mode) (SPIx_CTAR_SLAVE)

When the DSPI is configured as an SPI bus slave, the CTAR0 register is used.

Addresses: SPI0_CTAR0_SLAVE is 4002_C000h base + Ch offset = 4002_C00Ch
SPI1_CTAR0_SLAVE is 4002_D000h base + Ch offset = 4002_D00Ch
SPI2_CTAR0_SLAVE is 400A_C000h base + Ch offset = 400A_C00Ch

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	FMSZ					CPOL	CPHA	0		0																						
W																																
Reset	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPIx_CTAR*n*_SLAVE field descriptions

Field	Description
31–27 FMSZ	Frame Size The number of bits transferred per frame is equal to the FMSZ field value plus 1. The minimum valid FMSZ field value is 3.
26 CPOL	Clock Polarity Selects the inactive state of the Serial Communications Clock (SCK). 0 The inactive state value of SCK is low. 1 The inactive state value of SCK is high.

Table continues on the next page...

SPIx_CTAR*n*_SLAVE field descriptions (continued)

Field	Description
25 CPHA	Clock Phase Selects which edge of SCK causes data to change and which edge causes data to be captured. This bit is used in both master and slave mode. For successful communication between serial devices, the devices must have identical clock phase settings. In Continuous SCK mode, the bit value is ignored and the transfers are done as the CPHA bit is set to 1. 0 Data is captured on the leading edge of SCK and changed on the following edge. 1 Data is changed on the leading edge of SCK and captured on the following edge.
24–23 Reserved	This read-only field is reserved and always has the value zero.
22–0 Reserved	This read-only field is reserved and always has the value zero.

50.3.5 DSPI Status Register (SPIx_SR)

SR contains status and flag bits. The bits reflect the status of the DSPI and indicate the occurrence of events that can generate interrupt or DMA requests. Software can clear flag bits in the SR by writing a 1 to them. Writing a 0 to a flag bit has no effect. This register may not be writable in module disable mode due to the use of power saving mechanisms.

Addresses: SPI0_SR is 4002_C000h base + 2Ch offset = 4002_C02Ch
SPI1_SR is 4002_D000h base + 2Ch offset = 4002_D02Ch
SPI2_SR is 400A_C000h base + 2Ch offset = 400A_C02Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCF	TXRXS	0	EOQF	TFUF	0	TFFF	0	0	0	0	0	RFOF	0	RFDF	0
W	w1c	w1c		w1c	w1c		w1c						w1c		w1c	
Reset	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	TXCTR				TXNXTPTR				RXCTR				POPNXTPTR			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

* Notes:
- TFFF bitfield: Depends on MCR[MDIS] bit. See bit description for more details.

SPIx_SR field descriptions

Field	Description
31 TCF	Transfer Complete Flag Indicates that all bits in a frame have been shifted out. TCF remains set until it is cleared by writing a 1 to it. 0 Transfer not complete. 1 Transfer complete.
30 TXRXS	TX and RX Status Reflects the run status of the DSPI. 0 Transmit and receive operations are disabled (DSPI is in stopped state). 1 Transmit and receive operations are enabled (DSPI is in running state).
29 Reserved	This read-only field is reserved and always has the value zero.
28 EOQF	End of Queue Flag Indicates that the last entry in a queue has been transmitted when the DSPI is in master mode. The EOQF bit is set when the TX FIFO entry has the EOQ bit set in the command halfword and the end of the transfer is reached. The EOQF bit remains set until cleared by writing a 1 to it. When the EOQF bit is set, the TXRXS bit is automatically cleared. 0 EOQ is not set in the executing command. 1 EOQ is set in the executing SPI command.
27 TFUF	Transmit FIFO Underflow Flag Indicates an underflow condition in the TX FIFO. The transmit underflow condition is detected only for DSPI blocks operating in slave mode and SPI configuration. TFUF is set when the TX FIFO of a DSPI operating in SPI slave mode is empty and an external SPI master initiates a transfer. The TFUF bit remains set until cleared by writing 1 to it. 0 No Tx FIFO underflow. 1 Tx FIFO underflow has occurred.
26 Reserved	This read-only field is reserved and always has the value zero.
25 TFFF	Transmit FIFO Fill Flag Provides a method for the DSPI to request more entries to be added to the TX FIFO. The TFFF bit is set while the TX FIFO is not full. The TFFF bit can be cleared by writing 1 to it or by acknowledgement from the DMA controller to the TX FIFO full request. The Reset Value of this bit is 0 if MCR[MDIS] = 1. The Reset Value of this bit is 1 if MCR[MDIS] = 0. 0 Tx FIFO is full. 1 Tx FIFO is not full.
24 Reserved	This read-only field is reserved and always has the value zero.
23 Reserved	This read-only field is reserved and always has the value zero.
22 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

SPIx_SR field descriptions (continued)

Field	Description
21 Reserved	This read-only field is reserved and always has the value zero.
20 Reserved	This read-only field is reserved and always has the value zero.
19 RFOF	Receive FIFO Overflow Flag Indicates an overflow condition in the RX FIFO. The bit is set when the RX FIFO and shift register are full and a transfer is initiated. The bit remains set until it is cleared by writing a 1 to it. 0 No Rx FIFO overflow. 1 Rx FIFO overflow has occurred.
18 Reserved	This read-only field is reserved and always has the value zero.
17 RFDF	Receive FIFO Drain Flag Provides a method for the DSPI to request that entries be removed from the RX FIFO. The bit is set while the RX FIFO is not empty. The RFDF bit can be cleared by writing 1 to it or by acknowledgement from the DMA controller when the RX FIFO is empty. 0 Rx FIFO is empty. 1 Rx FIFO is not empty.
16 Reserved	This read-only field is reserved and always has the value zero.
15–12 TXCTR	TX FIFO Counter Indicates the number of valid entries in the TX FIFO. The TXCTR is incremented every time the PUSHR is written. The TXCTR is decremented every time a SPI command is executed and the SPI data is transferred to the shift register.
11–8 TXNXTPTR	Transmit Next Pointer Indicates which TX FIFO Entry is transmitted during the next transfer. The TXNXTPTR field is updated every time SPI data is transferred from the TX FIFO to the shift register.
7–4 RXCTR	RX FIFO Counter Indicates the number of entries in the RX FIFO. The RXCTR is decremented every time the POPR is read. The RXCTR is incremented every time data is transferred from the shift register to the RX FIFO.
3–0 POPNXTPTR	Pop Next Pointer Contains a pointer to the RX FIFO entry to be returned when the POPR is read. The POPNXTPTR is updated when the POPR is read.

50.3.6 DSPI DMA/Interrupt Request Select and Enable Register (SPIx_RSER)

RSER controls DMA and interrupt requests. Do not write to the RSER while the DSPI is in the Running state.

Addresses: SPI0_RSER is 4002_C000h base + 30h offset = 4002_C030h
SPI1_RSER is 4002_D000h base + 30h offset = 4002_D030h
SPI2_RSER is 400A_C000h base + 30h offset = 400A_C030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCF_RE	0	0	EOQF_RE	TFUF_RE	0	TFFF_RE	TFFF_DIRS	0	0	0	0	RFOF_RE	0	RFDF_RE	RFDF_DIRS
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															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SPIx_RSER field descriptions

Field	Description
31 TCF_RE	Transmission Complete Request Enable Enables TCF flag in the SR to generate an interrupt request. 0 TCF interrupt requests are disabled. 1 TCF interrupt requests are enabled.
30 Reserved	This read-only field is reserved and always has the value zero.
29 Reserved	This read-only field is reserved and always has the value zero.
28 EOQF_RE	DSPI Finished Request Enable Enables the EOQF flag in the SR to generate an interrupt request. 0 EOQF interrupt requests are disabled. 1 EOQF interrupt requests are enabled.
27 TFUF_RE	Transmit FIFO Underflow Request Enable Enables the TFUF flag in the SR to generate an interrupt request.

Table continues on the next page...

SPIx_RSER field descriptions (continued)

Field	Description
	0 TFUF interrupt requests are disabled. 1 TFUF interrupt requests are enabled.
26 Reserved	This read-only field is reserved and always has the value zero.
25 TFFF_RE	Transmit FIFO Fill Request Enable Enables the TFFF flag in the SR to generate a request. The TFFF_DIRS bit selects between generating an interrupt request or a DMA request. 0 TFFF interrupts or DMA requests are disabled. 1 TFFF interrupts or DMA requests are enabled.
24 TFFF_DIRS	Transmit FIFO Fill DMA or Interrupt Request Select Selects between generating a DMA request or an interrupt request. When the TFFF flag bit in the SR is set and the TFFF_RE bit in the RSER register is set, this bit selects between generating an interrupt request or a DMA request. 0 TFFF flag generates interrupt requests. 1 TFFF flag generates DMA requests.
23 Reserved	This read-only field is reserved and always has the value zero.
22 Reserved	This read-only field is reserved and always has the value zero.
21 Reserved	This read-only field is reserved and always has the value zero.
20 Reserved	This read-only field is reserved and always has the value zero.
19 RFOF_RE	Receive FIFO Overflow Request Enable Enables the RFOF flag in the SR to generate an interrupt request. 0 RFOF interrupt requests are disabled. 1 RFOF interrupt requests are enabled.
18 Reserved	This read-only field is reserved and always has the value zero.
17 RFDF_RE	Receive FIFO Drain Request Enable Enables the RFDF flag in the SR to generate a request. The RFDF_DIRS bit selects between generating an interrupt request or a DMA request. 0 RFDF interrupt or DMA requests are disabled 1 RFDF interrupt or DMA requests are enabled
16 RFDF_DIRS	Receive FIFO Drain DMA or Interrupt Request Select. Selects between generating a DMA request or an interrupt request. When the RFDF flag bit in the SR is set, and the RFDF_RE bit in the RSER is set, the RFDF_DIRS bit selects between generating an interrupt request or a DMA request.

Table continues on the next page...

SPIx_RSER field descriptions (continued)

Field	Description
	0 Interrupt request. 1 DMA request.
15–0 Reserved	This read-only field is reserved and always has the value zero.

50.3.7 DSPI PUSH TX FIFO Register In Master Mode (SPIx_PUSHR)

PUSHR provides the means to write to the TX FIFO. Data written to this register is transferred to the TX FIFO. Eight- or sixteen-bit write accesses to the PUSHR transfer all 32 register bits to the TX FIFO. The register structure is different in master and slave modes. In master mode the register provides 16-bit command and 16-bit data to the TX FIFO. In slave mode all 32 register bits can be used as data, supporting up to 32-bit SPI frame operation.

Addresses: SPI0_PUSHR is 4002_C000h base + 34h offset = 4002_C034h
SPI1_PUSHR is 4002_D000h base + 34h offset = 4002_D034h
SPI2_PUSHR is 400A_C000h base + 34h offset = 400A_C034h

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	CONT	CTAS			EOQ	CTCNT	0		0		PCS[5:0]						TXDATA															
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

SPIx_PUSHR field descriptions

Field	Description
31 CONT	Continuous Peripheral Chip Select Enable Selects a Continuous Selection Format. The bit is used in SPI master mode. The bit enables the selected PCS signals to remain asserted between transfers. 0 Return PCSn signals to their inactive state between transfers. 1 Keep PCSn signals asserted between transfers.
30–28 CTAS	Clock and Transfer Attributes Select. Selects which CTAR register to use in master mode to specify the transfer attributes for the associated SPI frame. In SPI slave mode, CTAR0 is used. See the Chip Configuration chapter to determine how many CTAR registers this device has. You should not program a value in this field for a register that is not present. 000 CTAR0 001 CTAR1

Table continues on the next page...

SPIx_PUSHR field descriptions (continued)

Field	Description
	010 Reserved 011 Reserved 100 Reserved 101 Reserved 110 Reserved 111 Reserved
27 EOQ	End Of Queue Host software uses this bit to signal to the DSPI that the current SPI transfer is the last in a queue. At the end of the transfer, the EOQF bit in the SR is set. 0 The SPI data is not the last data to transfer. 1 The SPI data is the last data to transfer.
26 CTCNT	Clear Transfer Counter. Clears the SPI_TCNT field in the TCR register. The SPI_TCNT field is cleared before the DSPI starts transmitting the current SPI frame. 0 Do not clear the TCR[SPI_TCNT] field. 1 Clear the TCR[SPI_TCNT] field.
25–24 Reserved	This read-only field is reserved and always has the value zero.
23–22 Reserved	This read-only field is reserved and always has the value zero.
21–16 PCS[5:0]	Select which PCS signals are to be asserted for the transfer. Refer to the chip configuration chapter for the number of PCS signals used in this MCU. 0 Negate the PCS[x] signal. 1 Assert the PCS[x] signal.
15–0 TXDATA	Transmit Data Holds SPI data to be transferred according to the associated SPI command.

50.3.8 DSPI PUSH TX FIFO Register In Slave Mode (SPIx_PUSHR_SLAVE)

PUSHR provides the means to write to the TX FIFO. Data written to this register is transferred to the TX FIFO. Eight- or sixteen-bit write accesses to the PUSHR transfer all 32 register bits to the TX FIFO. The register structure is different in master and slave modes. In master mode the register provides 16-bit command and 16-bit data to the TX FIFO. In slave mode all 32 register bits can be used as data, supporting up to 32-bit SPI frame operation.

Addresses: SPI0_PUSHR_SLAVE is 4002_C000h base + 34h offset = 4002_C034h
SPI1_PUSHR_SLAVE is 4002_D000h base + 34h offset = 4002_D034h
SPI2_PUSHR_SLAVE is 400A_C000h base + 34h offset = 400A_C034h

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

SPIx_PUSHR_SLAVE field descriptions

Field	Description
31–0 TXDATA	Transmit Data Holds SPI data to be transferred according to the associated SPI command.

50.3.9 DSPI POP RX FIFO Register (SPIx_POPR)

POPR is used to read the RX FIFO. Eight- or sixteen-bit read accesses to the POPR have the same effect on the RX FIFO as 32-bit read accesses. A write to this register will generate a Transfer Error.

Addresses: SPI0_POPR is 4002_C000h base + 38h offset = 4002_C038h
SPI1_POPR is 4002_D000h base + 38h offset = 4002_D038h
SPI2_POPR is 400A_C000h base + 38h offset = 400A_C038h

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	RXDATA																															
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

SPIx_POPR field descriptions

Field	Description
31–0 RXDATA	Received Data Contains the SPI data from the RX FIFO entry to which the Pop Next Data Pointer points.

50.3.10 DSPI Transmit FIFO Registers (SPIx_TXFR*n*)

TXFRn provide visibility into the TX FIFO for debugging purposes. Each register is an entry in the TX FIFO. The registers are read-only and cannot be modified. Reading the TXFRx registers does not alter the state of the TX FIFO.

Addresses: SPI0_TXFR0 is 4002_C000h base + 3Ch offset = 4002_C03Ch

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	TXCMD_TXDATA																TXDATA															
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

SPIx_TXFR*n* field descriptions

Field	Description
31–16 TXCMD_ TXDATA	Transmit Command or Transmit Data In master mode the TXCMD field contains the command that sets the transfer attributes for the SPI data. In slave mode, the TXDATA contains 16 MSB bits of the SPI data to be shifted out.
15–0 TXDATA	Transmit Data Contains the SPI data to be shifted out.

50.3.11 DSPI Receive FIFO Registers (SPIx_RXFR*n*)

RXFRn provide visibility into the RX FIFO for debugging purposes. Each register is an entry in the RX FIFO. The RXFR registers are read-only. Reading the RXFRx registers does not alter the state of the RX FIFO.

Addresses: SPI0_RXFR0 is 4002_C000h base + 7Ch offset = 4002_C07Ch

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	RXDATA																															
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

SPIx_RXFR*n* field descriptions

Field	Description
31–0 RXDATA	Receive Data Contains the received SPI data.

50.4 Functional Description

The Serial Peripheral Interface (DSPI) block supports full-duplex, synchronous serial communications between MCUs and peripheral devices. All communications are done with SPI-like protocol.

The DSPI has the following configurations:

- SPI Configuration in which the DSPI operates as a basic SPI or a queued SPI.

The DCONF field in the DSPI Module Configuration Register (MCR) determines the DSPI Configuration. See for the DSPI configuration values.

The CTARn registers hold clock and transfer attributes. The SPI configuration allows to select which CTAR to use on a frame by frame basis by setting a field in the SPI command.

See DSPI Clock and Transfer Attributes Registers for information on the fields of CTAR registers.

Typical master to slave connections are shown in the following figure. When a data transfer operation is performed, data is serially shifted a predetermined number of bit positions. Because the modules are linked, data is exchanged between the master and the slave. The data that was in the master shift register is now in the shift register of the slave, and vice versa. At the end of a transfer, the TCF bit in the SR is set to indicate a completed transfer.

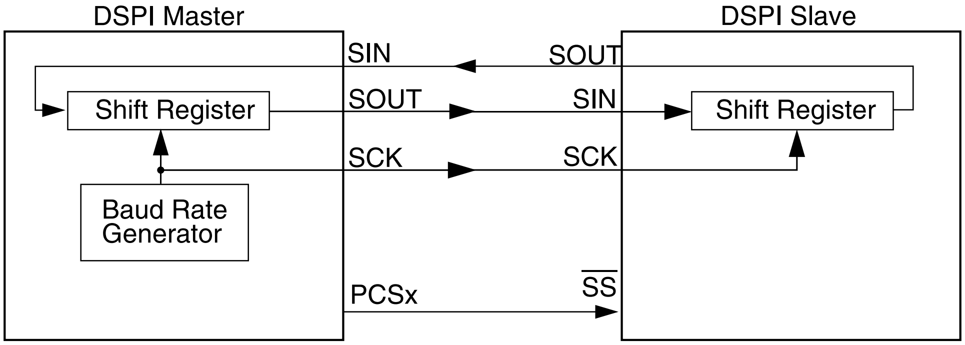


Figure 50-91. SPI Serial Protocol Overview

Generally more than one slave device can be connected to the DSPI master. 6 Peripheral Chip Select (PCS) signals of the DSPI masters can be used to select which of the slaves to communicate with. Refer to the chip configuration chapter for the number of PCS signals used in this MCU.

The three DSPI configurations share transfer protocol and timing properties which are described independently of the configuration in Transfer Formats . The transfer rate and delay settings are described in DSPI Baud Rate and Clock Delay Generation.

50.4.1 Start and Stop of DSPI Transfers

The DSPI has two operating states: STOPPED and RUNNING. The states are independent of DSPI configuration. The default state of the DSPI is STOPPED. In the STOPPED state no serial transfers are initiated in master mode and no transfers are responded to in slave mode. The STOPPED state is also a safe state for writing the various configuration registers of the DSPI without causing undetermined results. In the RUNNING state serial transfers take place.

The TXRXS bit in the SR indicates what state the DSPI in. The bit is set if the module in RUNNING state.

The DSPI is started (DSPI transitions to RUNNING) when all of the following conditions are true:

- SR[EOQF] bit is clear

- MCU is not in the debug mode or the MCR[FRZ] bit is clear
- MCR[HALT] bit is clear

The DSPI stops (transitions from RUNNING to STOPPED) after the current frame when any one of the following conditions exist:

- SR[EOQF] bit is set
- MCU in the debug mode and the MCR[FRZ] bit is set
- MCR[HALT] bit is set

State transitions from RUNNING to STOPPED occur on the next frame boundary if a transfer is in progress, or immediately if no transfers are in progress.

50.4.2 Serial Peripheral Interface (SPI) Configuration

The SPI Configuration transfers data serially using a shift register and a selection of programmable transfer attributes. The DSPI is in SPI Configuration when the DCONF field in the MCR is 0b00. The SPI frames can be 32 bits long. The host CPU or a DMA controller transfers the SPI data from the external to DSPI RAM queues to a transmit FIFO (TX FIFO) buffer. The received data is stored in entries in the Receive FIFO (RX FIFO) buffer. The host CPU or the DMA controller transfers the received data from the RX FIFO to memory external to the DSPI. The FIFO buffers operation is described in Transmit First In First Out (TX FIFO) Buffering Mechanism, and Receive First In First Out (RX FIFO) Buffering Mechanism. The interrupt and DMA request conditions are described in Interrupts/DMA Requests.

The SPI Configuration supports two block-specific modes —master mode and slave mode. The FIFO operations are similar for both modes. The main difference is that in master mode the DSPI initiates and controls the transfer according to the fields in the SPI command field of the TX FIFO entry. In slave mode, the DSPI only responds to transfers initiated by a bus master external to the DSPI and the SPI command field space is used for 16 most significant bit of the transmit data.

50.4.2.1 Master Mode

In SPI master mode the DSPI initiates the serial transfers by controlling the Serial Communications Clock (SCK) and the Peripheral Chip Select (PCS) signals. The SPI command field in the executing TX FIFO entry determines which CTAR registers will be used to set the transfer attributes and which PCS signals to assert. The command field also contains various bits that help with queue management and transfer protocol. See DSPI PUSH TX FIFO Register (PUSHR) for details on the SPI command fields. The data field in the executing TX FIFO entry is loaded into the shift register and shifted out on the Serial Out (SOUT) pin. In SPI master mode, each SPI frame to be transmitted has a command associated with it allowing for transfer attribute control on a frame by frame basis.

50.4.2.2 Slave Mode

In SPI slave mode the DSPI responds to transfers initiated by a SPI bus master. The DSPI does not initiate transfers. Certain transfer attributes such as clock polarity, clock phase and frame size must be set for successful communication with a SPI master. The SPI slave mode transfer attributes are set in the CTAR0. The data is shifted out with MSB first. Shifting out of LSB is not supported in this mode.

50.4.2.3 FIFO Disable Operation

The FIFO disable mechanisms allow SPI transfers without using the TX FIFO or RX FIFO. The DSPI operates as a double-buffered simplified SPI when the FIFOs are disabled. The FIFOs are disabled separately; setting the MCR[DIS_TXF] bit disables the TX FIFO, and setting the MCR[DIS_RXF] bit disables the RX FIFO.

The FIFO Disable mechanisms are transparent to the user and to host software; Transmit data and commands are written to the PUSHR and received data is read from the POPR.

When the TX FIFO is disabled the TFFF, TFUF and TXCTR fields in SR behave as if there is a one-entry FIFO but the contents of the TXFR registers and TXNXTPTR are undefined. Likewise, when the RX FIFO is disabled, the RFDF, RFOF and RXCTR fields in the SR behave as if there is a one-entry FIFO, but the contents of the RXFR registers and POPNXTPTR are undefined.

50.4.2.4 Transmit First In First Out (TX FIFO) Buffering Mechanism

The TX FIFO functions as a buffer of SPI data and SPI commands for transmission. The TX FIFO holds 4 words, each consisting of a command field and a data field. The number of entries in the TX FIFO is device-specific. SPI commands and data are added to the TX FIFO by writing to the DSPI PUSH TX FIFO Register (PUSHR). TX FIFO entries can only be removed from the TX FIFO by being shifted out or by flushing the TX FIFO.

The TX FIFO Counter field (TXCTR) in the DSPI Status Register (SR) indicates the number of valid entries in the TX FIFO. The TXCTR is updated every time the DSPI _PUSHR is written or SPI data is transferred into the shift register from the TX FIFO.

The TXNXTPTR field indicates which TX FIFO Entry will be transmitted during the next transfer. The TXNXTPTR contains the positive offset from TXFR0 in number of 32-bit registers. For example, TXNXTPTR equal to two means that the TXFR2 contains the SPI data and command for the next transfer. The TXNXTPTR field is incremented every time SPI data is transferred from the TX FIFO to the shift register. The maximum value of the field is equal to the maximum implemented TXFR register number and it rolls over after reaching the maximum.

50.4.2.4.1 Filling the TX FIFO

Host software or other intelligent blocks can add (push) entries to the TX FIFO by writing to the PUSHR. When the TX FIFO is not full, the TX FIFO Fill Flag (TFFF) in the SR is set. The TFFF bit is cleared when TX FIFO is full and the DMA controller indicates that a write to PUSHR is complete. Writing a '1' to the TFFF bit also clears it. The TFFF can generate a DMA request or an interrupt request. See Transmit FIFO Fill Interrupt or DMA Request for details.

The DSPI ignores attempts to push data to a full TX FIFO, the state of the TX FIFO does not change and no error condition is indicated.

50.4.2.4.2 Draining the TX FIFO

The TX FIFO entries are removed (drained) by shifting SPI data out through the shift register. Entries are transferred from the TX FIFO to the shift register and shifted out as long as there are valid entries in the TX FIFO. Every time an entry is transferred from the TX FIFO to the shift register, the TX FIFO Counter decrements by one. At the end of a transfer, the TCF bit in the SR is set to indicate the completion of a transfer. The TX FIFO is flushed by writing a '1' to the CLR_TXF bit in MCR.

If an external bus master initiates a transfer with a DSPI slave while the slave's DSPI TX FIFO is empty, the Transmit FIFO Underflow Flag (TFUF) in the slave's SR is set. See Transmit FIFO Underflow Interrupt Request for details.

50.4.2.5 Receive First In First Out (RX FIFO) Buffering Mechanism

The RX FIFO functions as a buffer for data received on the SIN pin. The RX FIFO holds 4 received SPI data frames. The number of entries in the RX FIFO is device-specific. SPI data is added to the RX FIFO at the completion of a transfer when the received data in the shift register is transferred into the RX FIFO. SPI data are removed (popped) from the RX FIFO by reading the DSPI POP RX FIFO Register (POPR). RX FIFO entries can only be removed from the RX FIFO by reading the POPR or by flushing the RX FIFO.

The RX FIFO Counter field (RXCTR) in the DSPI Status Register (SR) indicates the number of valid entries in the RX FIFO. The RXCTR is updated every time the POPR is read or SPI data is copied from the shift register to the RX FIFO.

The POPNXTPTR field in the SR points to the RX FIFO entry that is returned when the POPR is read. The POPNXTPTR contains the positive offset from RXFR0 in number of 32-bit registers. For example, POPNXTPTR equal to two means that the RXFR2 contains the received SPI data that will be returned when POPR is read. The POPNXTPTR field is incremented every time the POPR is read. The maximum value of the field is equal to the maximum implemented RXFR register number and it rolls over after reaching the maximum.

50.4.2.5.1 Filling the RX FIFO

The RX FIFO is filled with the received SPI data from the shift register. While the RX FIFO is not full, SPI frames from the shift register are transferred to the RX FIFO. Every time a SPI frame is transferred to the RX FIFO the RX FIFO Counter is incremented by one.

If the RX FIFO and shift register are full and a transfer is initiated, the RFOF bit in the SR is set indicating an overflow condition. Depending on the state of the ROOE bit in the MCR, the data from the transfer that generated the overflow is either ignored or shifted in to the shift register. If the ROOE bit is set, the incoming data is shifted in to the shift register. If the ROOE bit is cleared, the incoming data is ignored.

50.4.2.5.2 Draining the RX FIFO

Host CPU or a DMA can remove (pop) entries from the RX FIFO by reading the DSPI POP RX FIFO Register (POPR). A read of the POPR decrements the RX FIFO Counter by one. Attempts to pop data from an empty RX FIFO are ignored and the RX FIFO Counter remains unchanged. The data, read from the empty RX FIFO, is undetermined.

When the RX FIFO is not empty, the RX FIFO Drain Flag (RFDF) in the SR is set. The RFDF bit is cleared when the RX_FIFO is empty and the DMA controller indicates that a read from POPR is complete or by writing a '1' to it.

50.4.3 DSPI Baud Rate and Clock Delay Generation

The SCK frequency and the delay values for serial transfer are generated by dividing the system clock frequency by a prescaler and a scaler with the option for doubling the baud rate. The following figure shows conceptually how the SCK signal is generated.

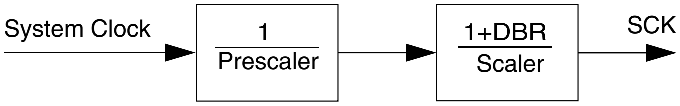


Figure 50-92. Communications Clock Prescalers and Scalers

50.4.3.1 Baud Rate Generator

The Baud Rate is the frequency of the Serial Communication Clock (SCK). The system clock is divided by a prescaler (PBR) and scaler (BR) to produce SCK with the possibility of halving the scaler division. The DBR, PBR and BR fields in the CTAR registers select the frequency of SCK by the formula in the BR field description. The following table shows an example of how to compute the baud rate.

Table 50-106. Baud Rate Computation Example

f_{sys}	PBR	Prescaler	BR	Scaler	DBR	Baud Rate
100 MHz	0b00	2	0b0000	2	0	25 Mb/s
20 MHz	0b00	2	0b0000	2	1	10 Mb/s

NOTE

The clock frequencies mentioned in the preceding table are given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

50.4.3.2 PCS to SCK Delay (t_{CSC})

The PCS to SCK delay is the length of time from assertion of the PCS signal to the first SCK edge. See Figure 50-94 for an illustration of the PCS to SCK delay. The PCSSCK and CSSCK fields in the CTAR*x* registers select the PCS to SCK delay by the formula in the CSSCK field description. The following table shows an example of how to compute the PCS to SCK delay.

Table 50-107. PCS to SCK Delay Computation Example

f_{sys}	PCSSCK	Prescaler	CSSCK	Scaler	PCS to SCK Delay
100 MHz	0b01	3	0b0100	32	0.96 μs

NOTE

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

50.4.3.3 After SCK Delay (t_{ASC})

The After SCK Delay is the length of time between the last edge of SCK and the negation of PCS. See Figure 50-94 and Figure 50-95 for illustrations of the After SCK delay. The PASC and ASC fields in the CTAR*x* registers select the After SCK Delay by the formula in the ASC field description. The following table shows an example of how to compute the After SCK delay.

Table 50-108. After SCK Delay Computation Example

f_{sys}	PASC	Prescaler	ASC	Scaler	After SCK Delay
100 MHz	0b01	3	0b0100	32	0.96 µs

NOTE

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

50.4.3.4 Delay after Transfer (t_{DT})

The Delay after Transfer is the minimum time between negation of the PCS signal for a frame and the assertion of the PCS signal for the next frame. See Figure 50-94 for an illustration of the Delay after Transfer. The PDT and DT fields in the CTAR*x* registers select the Delay after Transfer by the formula in the DT field description. The following table shows an example of how to compute the Delay after Transfer.

Table 50-109. Delay after Transfer Computation Example

f_{sys}	PDT	Prescaler	DT	Scaler	Delay after Transfer
100 MHz	0b01	3	0b1110	32768	0.98 ms

NOTE

The clock frequency mentioned in the preceding table is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

When in non-continuous clock mode the t_{DT} delay is configured according to the equation specified in the CTAR[DT] bitfield description. When in continuous clock mode, the delay is fixed at 1 SCK period.

50.4.3.5 Peripheral Chip Select Strobe Enable ($\overline{\text{PCSS}}$)

The $\overline{\text{PCSS}}$ signal provides a delay to allow the PCS signals to settle after a transition occurs thereby avoiding glitches. When the DSPI is in master mode and the PCSSE bit is set in the MCR, $\overline{\text{PCSS}}$ provides a signal for an external demultiplexer to decode the PCS[0] - PCS[4] signals into as many as 128 glitch-free PCS signals. The following figure shows the timing of the $\overline{\text{PCSS}}$ signal relative to PCS signals.

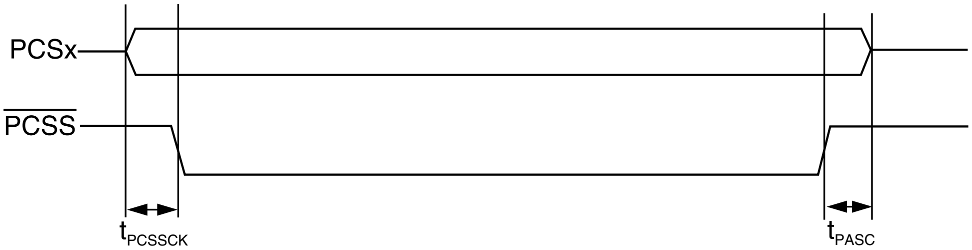


Figure 50-93. Peripheral Chip Select Strobe Timing

The delay between the assertion of the PCS signals and the assertion of $\overline{\text{PCSS}}$ is selected by the PCSSCK field in the CTAR based on the following formula:

$$t_{\text{PCSSCK}} = \frac{1}{f_{\text{SYS}}} \times \text{PCSSCK}$$

At the end of the transfer the delay between $\overline{\text{PCSS}}$ negation and PCS negation is selected by the PASC field in the CTAR based on the following formula:

$$t_{\text{PASC}} = \frac{1}{f_{\text{SYS}}} \times \text{PASC}$$

The following table shows an example of how to compute the t_{pcssck} delay.

Table 50-110. Peripheral Chip Select Strobe Assert Computation Example

f_{sys}	PCSSCK	Prescaler	Delay before Transfer
100 MHz	0b11	7	70.0 ns

The following table shows an example of how to compute the t_{pasc} delay.

Table 50-111. Peripheral Chip Select Strobe Negate Computation Example

f_{sys}	PASC	Prescaler	Delay after Transfer
100 MHz	0b11	7	70.0 ns

The $\overline{\text{PCSS}}$ signal is not supported when Continuous Serial Communication SCK mode are enabled.

NOTE

The clock frequency mentioned in the preceding tables is given as an example. Refer to the clocking chapter for the frequency used to drive this module in the device.

50.4.4 Transfer Formats

The SPI serial communication is controlled by the Serial Communications Clock (SCK) signal and the PCS signals. The SCK signal provided by the master device synchronizes shifting and sampling of the data on the SIN and SOUT pins. The PCS signals serve as enable signals for the slave devices.

In master mode, the CPOL and CPHA bits in the Clock and Transfer Attributes Registers (CTARn) select the polarity and phase of the serial clock, SCK.

- CPOL - Selects the idle state polarity of the SCK
- CPHA - Selects if the data on SOUT is valid before or on the first SCK edge

Even though the bus slave does not control the SCK signal, in slave mode these values must be identical to the master device settings to ensure proper transmission. In SPI slave mode, only CTAR0 is used.

The DSPI supports four different transfer formats:

- Classic SPI with CPHA=0
- Classic SPI with CPHA=1
- Modified Transfer format with CPHA = 0
- Modified Transfer format with CPHA = 1

A modified transfer format is supported to allow for high-speed communication with peripherals that require longer setup times. The DSPI can sample the incoming data later than halfway through the cycle to give the peripheral more setup time. The MTFE bit in the MCR selects between Classic SPI Format and Modified Transfer Format.

In the SPI Configurations, the DSPI provides the option of keeping the PCS signals asserted between frames. See Continuous Selection Format for details.

50.4.4.1 Classic SPI Transfer Format (CPHA = 0)

The transfer format shown in following figure is used to communicate with peripheral SPI slave devices where the first data bit is available on the first clock edge. In this format, the master and slave sample their SIN pins on the odd-numbered SCK edges and change the data on their SOUT pins on the even-numbered SCK edges.

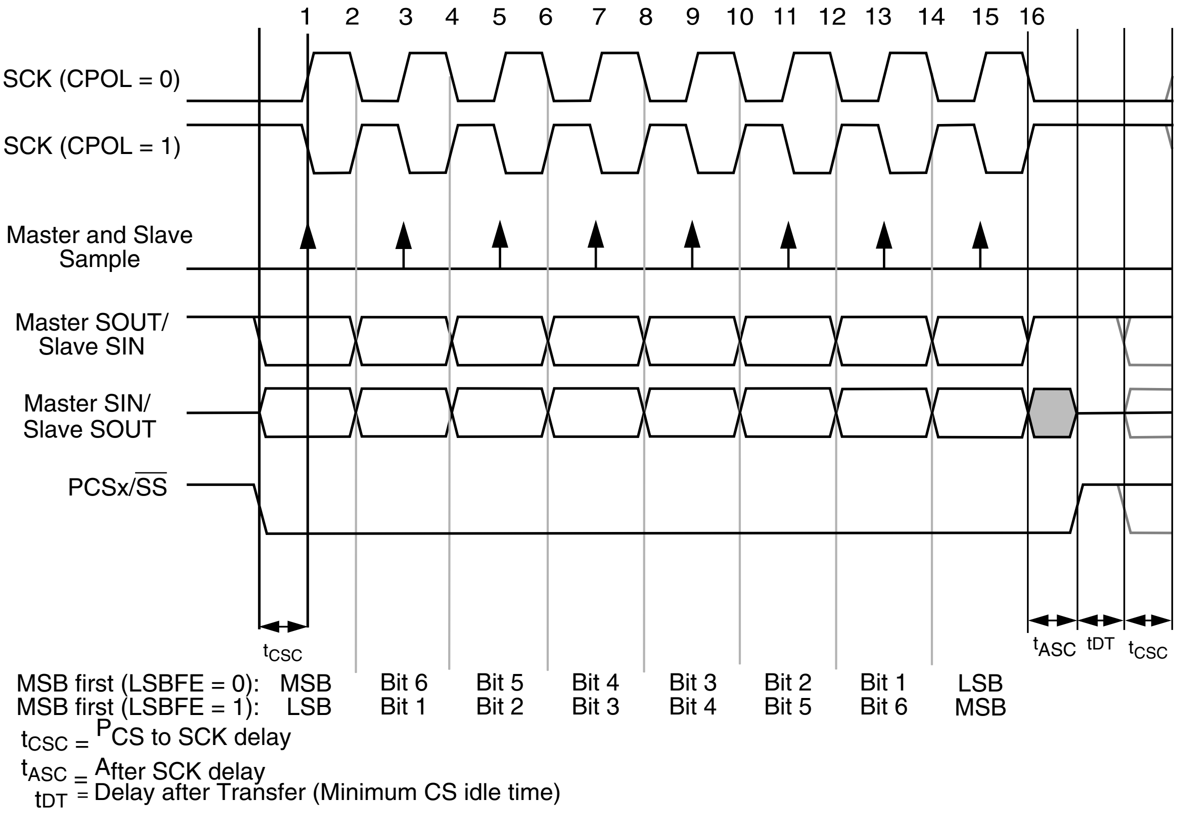

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
SCK (CPOL = 0)
SCK (CPOL = 1)
Master and Slave Sample
Master SOUT/ Slave SIN
Master SIN/ Slave SOUT
PCSx/$\overline{SS}$
t_{CSC}
t_{ASC}
tDT
t_{CSC}
MSB first (LSBFE = 0): MSB Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 LSB
MSB first (LSBFE = 1): LSB Bit 1 Bit 2 Bit 3 Bit 4 Bit 5 Bit 6 MSB
t_{CSC} = PCS to SCK delay
t_{ASC} = After SCK delay
tDT = Delay after Transfer (Minimum CS idle time)

Figure 50-94. DSPI Transfer Timing Diagram (MTFE=0, CPHA=0, FMSZ=8)

The master initiates the transfer by placing its first data bit on the SOUT pin and asserting the appropriate peripheral chip select signals to the slave device. The slave responds by placing its first data bit on its SOUT pin. After the tCSC delay elapses, the master outputs the first edge of SCK. The master and slave devices use this edge to sample the first input data bit on their serial data input signals. At the second edge of the SCK the master and slave devices place their second data bit on their serial data output signals. For the rest of the frame the master and the slave sample their SIN pins on the odd-numbered clock edges and changes the data on their SOUT pins on the even-numbered clock edges. After the last clock edge occurs a delay of t_{ASC} is inserted before the master negates the PCS signals. A delay of t_{DT} is inserted before a new frame transfer can be initiated by the master.

50.4.4.2 Classic SPI Transfer Format (CPHA = 1)

This transfer format shown in the following figure is used to communicate with peripheral SPI slave devices that require the first SCK edge before the first data bit becomes available on the slave SOUT pin. In this format the master and slave devices change the data on their SOUT pins on the odd-numbered SCK edges and sample the data on their SIN pins on the even-numbered SCK edges

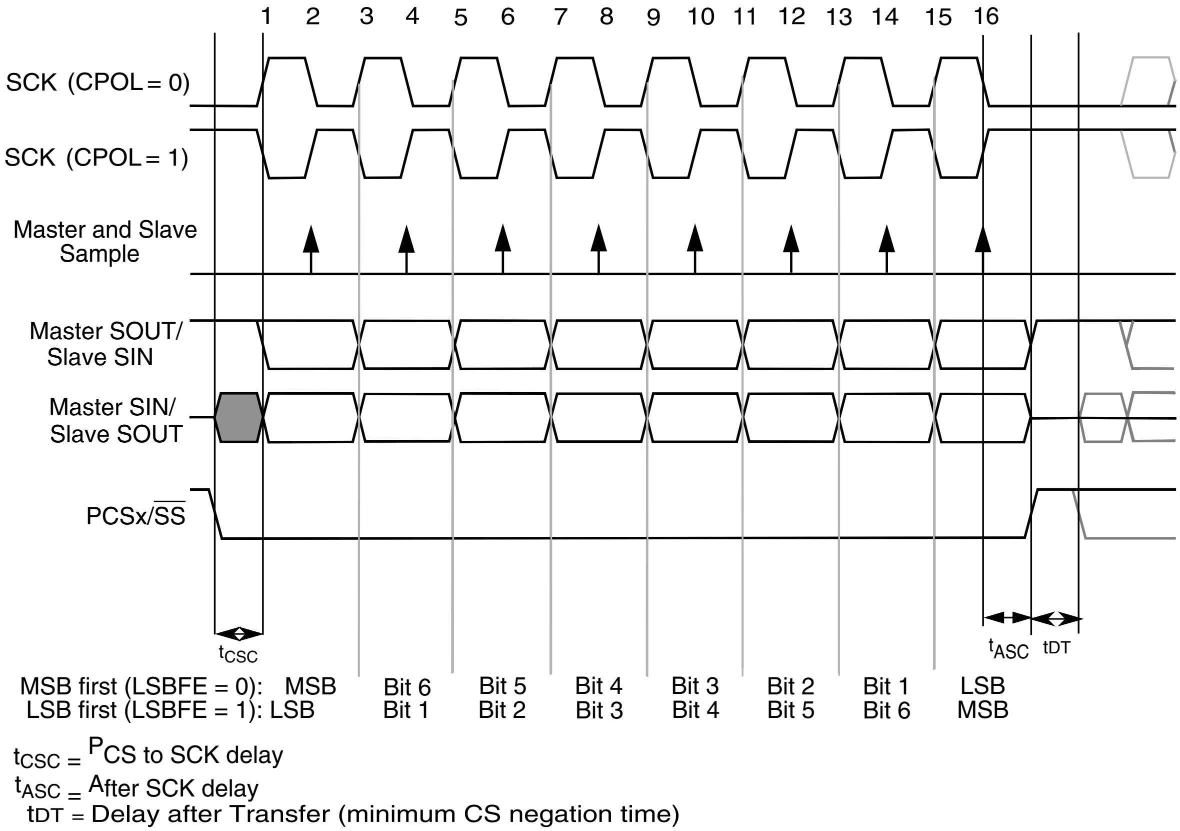


Figure 50-95. DSPI Transfer Timing Diagram (MTFE=0, CPHA=1, FMSZ=8)

The master initiates the transfer by asserting the PCS signal to the slave. After the t_{CSC} delay has elapsed, the master generates the first SCK edge and at the same time places valid data on the master SOUT pin . The slave responds to the first SCK edge by placing its first data bit on its slave SOUT pin.

At the second edge of the SCK the master and slave sample their SIN pins. For the rest of the frame the master and the slave change the data on their SOUT pins on the odd-numbered clock edges and sample their SIN pins on the even-numbered clock edges.

After the last clock edge occurs a delay of t_{ASC} is inserted before the master negates the PCS signal. A delay of t_{DT} is inserted before a new frame transfer can be initiated by the master.

50.4.4.3 Continuous Selection Format

Some peripherals must be deselected between every transfer. Other peripherals must remain selected between several sequential serial transfers. The Continuous Selection Format provides the flexibility to handle the following case. The Continuous Selection Format is enabled for the SPI Configuration by setting the CONT bit in the SPI command. The behavior of the PCS signals in the configurations is identical so only SPI Configuration will be described.

When the CONT bit = 0, the DSPI drives the asserted Chip Select signals to their idle states in between frames. The idle states of the Chip Select signals are selected by the PCSISn bits in the MCR. The following timing diagram is for two four-bit transfers with CPHA = 1 and CONT = 0.

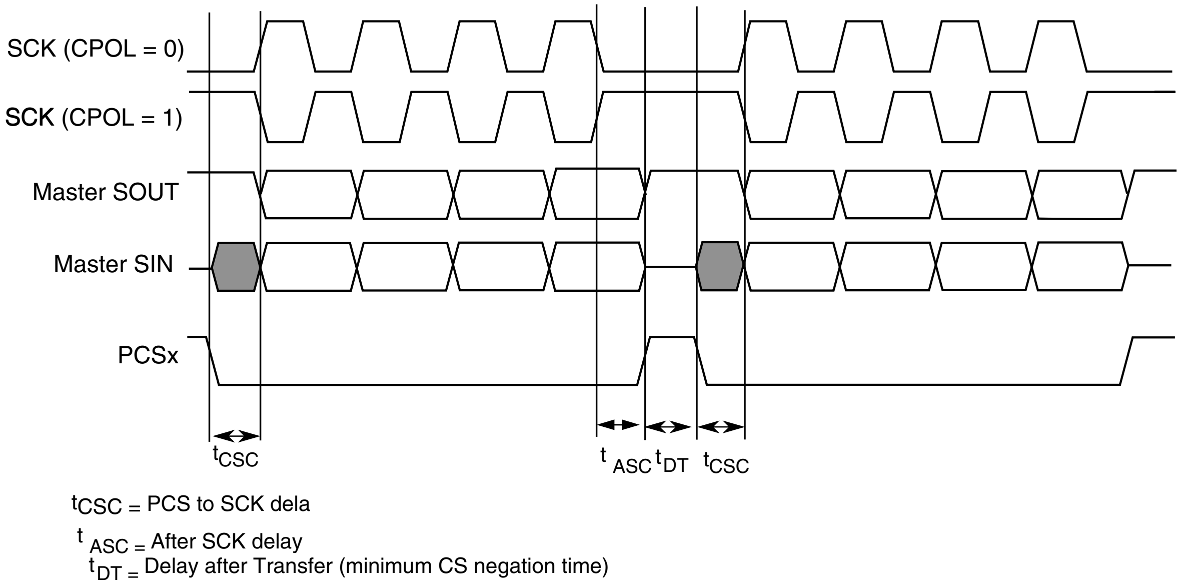


Figure 50-96. Example of Non-Continuous Format (CPHA=1, CONT=0)

When the CONT bit = 1, the PCS signal remains asserted for the duration of the two transfers. The Delay between Transfers (t_{DT}) is not inserted between the transfers. The following figure shows the timing diagram for two four-bit transfers with CPHA = 1 and CONT = 1.

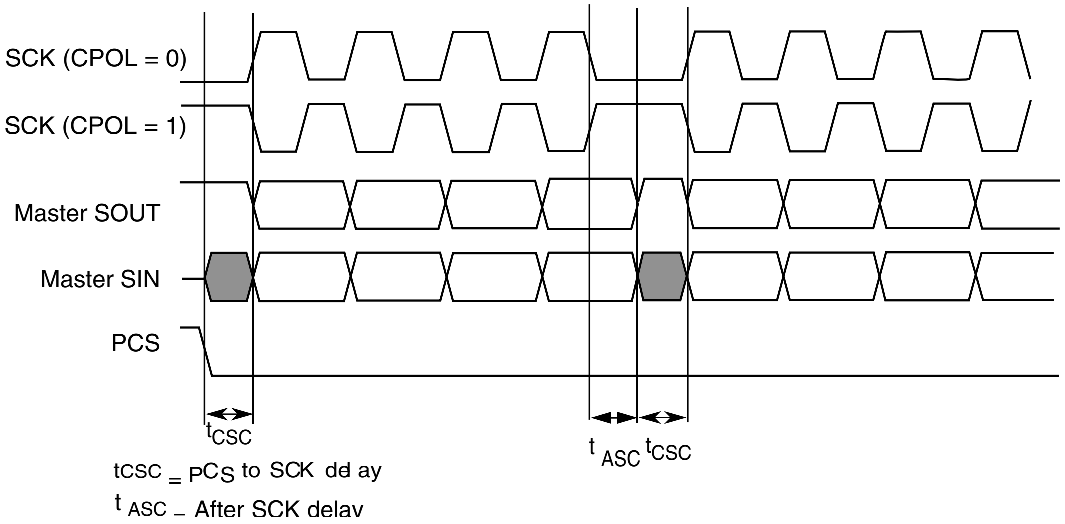


Figure 50-97. Example of Continuous Transfer (CPHA=1, CONT=1)

When using DSPI with continuous selection follow these rules:

- All transmit commands must have the same PCSn bits programming.
- The CTARs, selected by transmit commands, must be programmed with the same transfer attributes. Only FMSZ field can be programmed differently in these CTARs.
- When transmitting multiple frames in this mode, the user software must ensure that the last frame has the PUSHR[CONT] bit de-asserted (in master mode) and the user software must provide sufficient frames in the TX_FIFO to be sent out (in slave mode) and the master de-asserts the PCSn at end of transmission of last frame.
- The PUSHR[CONT] / DSICR0[DCONT] bits must be de-asserted before asserting MCR[HALT] bit (in master mode). This will make sure that the PCSn signals are de-asserted. Asserting MCR[HALT] bit during continuous transfer will cause the PCSn signals to remain asserted and hence Slave Device cannot transition from RUNNING to STOPPED state.

NOTE

User must fill the TXFIFO with the number of entries that will be concatenated together under one PCS assertion for both master and slave before the TXFIFO becomes empty.

When operating in slave mode, ensure that when the last-entry in the TXFIFO is completely transmitted (that is the corresponding TCF flag is asserted and TXFIFO is empty), the slave is deselected for any further serial communication; otherwise, an underflow error occurs.

50.4.5 Continuous Serial Communications Clock

The DSPI provides the option of generating a continuous SCK signal for slave peripherals that require a continuous clock.

Continuous SCK is enabled by setting the CONT_SCKE bit in the MCR. Enabling this bit generates the Continuous Serial Communications Clock regardless of the MCR[HALT] bit status.. Continuous SCK is valid in all configurations.

Continuous SCK is only supported for CPHA=1. Clearing CPHA is ignored if the CONT_SCKE bit is set. Continuous SCK is supported for Modified Transfer Format.

Clock and transfer attributes for the Continuous SCK mode are set according to the following rules:

- When the DSPI is in SPI configuration, CTAR0 is used initially. At the start of each SPI frame transfer, the CTAR specified by the CTAS for the frame is used.
- In all configurations, the currently selected CTAR remains in use until the start of a frame with a different CTAR specified, or the Continuous SCK mode is terminated.

It is recommended to keep the baud rate the same while using the Continuous SCK. Switching clock polarity between frames while using Continuous SCK can cause errors in the transfer. Continuous SCK operation is not guaranteed if the DSPI is put into the External Stop mode or Module Disable mode.

Enabling Continuous SCK disables the PCS to SCK delay and the Delay after Transfer (t_{DT}) is fixed to one SCK cycle. The following figure is the timing diagram for Continuous SCK format with Continuous Selection disabled.

NOTE

When in Continuous SCK mode, for the SPI transfer CTAR0 should always be used, and the TXFIFO must be cleared using the MCR[CLR_TXF] field before initiating transfer.

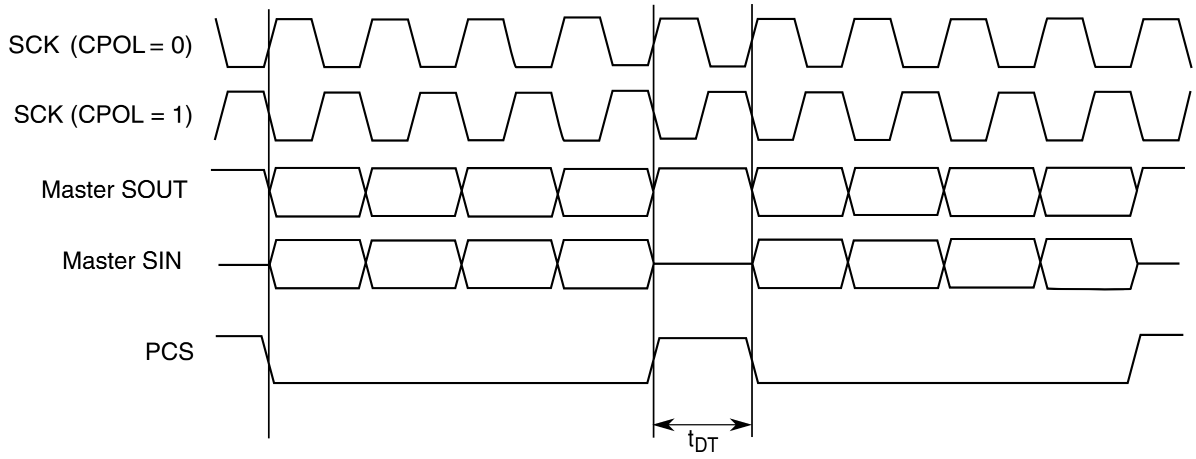


Figure 50-98. Continuous SCK Timing Diagram (CONT=0)

If the CONT bit in the TX FIFO entry is set, PCS remains asserted between the transfers. Under certain conditions, SCK can continue with PCS asserted, but with no data being shifted out of SOUT (SOUT pulled high). This can cause the slave to receive incorrect data. Those conditions include:

- Continuous SCK with CONT bit set, but no data in the transmit FIFO.
- Continuous SCK with CONT bit set and entering STOPPED state (refer to Start and Stop of DSPI Transfers).
- Continuous SCK with CONT bit set and entering Stop mode or Module Disable mode.

The following figure shows timing diagram for Continuous SCK format with Continuous Selection enabled.

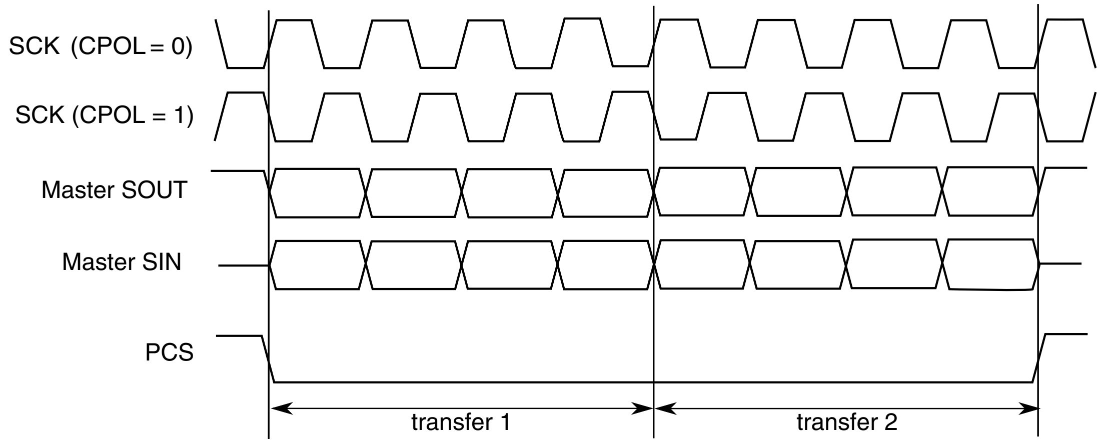


Figure 50-99. Continuous SCK Timing Diagram (CONT=1)

50.4.6 Slave Mode Operation Constraints

Slave mode logic shift register is buffered. This allows data streaming operation, when the DSPI is permanently selected and data is shifted in with a constant rate.

The transmit data is transferred at second SCK clock edge of the each frame to the shift register if the $\overline{\mathrm{SS}}$ signal is asserted and any time when transmit data is ready and $\overline{\mathrm{SS}}$ signal is negated.

Received data is transferred to the receive buffer at last SCK edge of each frame, defined by frame size programmed to the CTAR0/1 register. Then the data from the buffer is transferred to the RXFIFO or DDR register.

If the $\overline{\mathrm{SS}}$ negates before that last SCK edge, the data from shift register is lost.

This buffering scheme allows to operate slave clock with higher frequency than the system frequency. The clocks relationship is defined by the following equation. *FrameSize* is the value of the CTAR0/1[FMSZ] field plus one.

$$f_{\mathrm{SCK}} < f_{\mathrm{SYS}} \times \mathrm{FrameSize} \big/ 3$$

50.4.7 Interrupts/DMA Requests

The DSPI has several conditions that can only generate interrupt requests and two conditions that can generate interrupt or DMA requests. The following table lists these conditions.

Table 50-112. Interrupt and DMA Request Conditions

Condition	Flag	Interrupt	DMA
End of Queue (EOQ)	EOQF	Yes	
TX FIFO Fill	TFFF	Yes	Yes
Transfer Complete	TCF	Yes	
TX FIFO Underflow	TFUF	Yes	
RX FIFO Drain	RFDF	Yes	Yes
RX FIFO Overflow	RFOF	Yes	

Each condition has a flag bit in the DSPI Status Register (SR) and an Request Enable bit in the DSPI DMA/Interrupt Request Select and Enable Register (RSER). The TX FIFO Fill Flag (TFFF) and RX FIFO Drain Flag (RFDF) generate interrupt requests or DMA requests depending on the TFFF_DIRS and RFDF_DIRS bits in the RSER.

The DSPI module also provides a global interrupt request line, which is asserted when any of individual interrupt requests lines is asserted.

50.4.7.1 End of Queue Interrupt Request

The End of Queue Request indicates that the end of a transmit queue is reached. The End of Queue Request is generated when the EOQ bit in the executing SPI command is set and the EOQF_RE bit in the RSER is set.

NOTE

This interrupt request is generated when the last bit of the SPI frame with EOQ bit set is transmitted.

50.4.7.2 Transmit FIFO Fill Interrupt or DMA Request

The Transmit FIFO Fill Request indicates that the TX FIFO is not full. The Transmit FIFO Fill Request is generated when the number of entries in the TX FIFO is less than the maximum number of possible entries, and the TFFF_RE bit in the RSER is set. The TFFF_DIRS bit in the RSER selects whether a DMA request or an interrupt request is generated.

NOTE

TFFF flag clears automatically when DMA is used to fill TXFIFO.

To clear TFFF when not using DMA, follow these steps for every PUSH performed using CPU to fill TXFIFO:

1. Wait until TFFF = 1
2. Write data to PUSHR using CPU.
3. Clear TFFF by writing a 1 to its location. If FIFO is not full, this flag will not clear.

50.4.7.3 Transfer Complete Interrupt Request

The Transfer Complete Request indicates the end of the transfer of a serial frame. The Transfer Complete Request is generated at the end of each frame transfer when the TCF_RE bit is set in the RSER.

50.4.7.4 Transmit FIFO Underflow Interrupt Request

The Transmit FIFO Underflow Request indicates that an underflow condition in the TX FIFO has occurred. The transmit underflow condition is detected only for the DSPI, operating in slave mode and SPI configuration . The TFUF bit is set when the TX FIFO of a DSPI is empty, and a transfer is initiated from an external SPI master. If the TFUF bit is set while the TFUF_RE bit in the RSER is set, an interrupt request is generated.

50.4.7.5 Receive FIFO Drain Interrupt or DMA Request

The Receive FIFO Drain Request indicates that the RX FIFO is not empty. The Receive FIFO Drain Request is generated when the number of entries in the RX FIFO is not zero, and the RFDF_RE bit in the RSER is set. The RFDF_DIRS bit in the RSER selects whether a DMA request or an interrupt request is generated.

50.4.7.6 Receive FIFO Overflow Interrupt Request

The Receive FIFO Overflow Request indicates that an overflow condition in the RX FIFO has occurred. A Receive FIFO Overflow request is generated when RX FIFO and shift register are full and a transfer is initiated. The RFOF_RE bit in the RSER must be set for the interrupt request to be generated.

Depending on the state of the ROOE bit in the MCR, the data from the transfer that generated the overflow is either ignored or shifted in to the shift register. If the ROOE bit is set, the incoming data is shifted in to the shift register. If the ROOE bit is cleared, the incoming data is ignored.

50.4.8 Power Saving Features

The DSPI supports following power-saving strategies:

- External Stop mode
- Module Disable mode - Clock gating of non-memory mapped logic

50.4.8.1 Stop Mode (External Stop Mode)

The DSPI supports the stop mode protocol. When a request is made to enter external stop mode, the DSPI block acknowledges the request . If a serial transfer is in progress, the DSPI waits until it reaches the frame boundary before it is ready to have its clocks shut off .While the clocks are shut off, the DSPI memory-mapped logic is not accessible. The states of the interrupt and DMA request signals cannot be changed while in External Stop mode.

50.4.8.2 Module Disable Mode

Module disable mode is a block-specific mode that the DSPI can enter to save power. Host CPU can initiate the module disable mode by setting the MDIS bit in the MCR. The module disable mode can also be initiated by hardware. A power management block can initiate the module disable mode by asserting the DOZE mode signal while the DOZE bit in the MCR is set.

When the MDIS bit is set or the DOZE mode signal is asserted while the DOZE bit is set, the DSPI negates Clock Enable signal at the next frame boundary. If implemented, the Clock Enable signal can stop the clock to the non-memory mapped logic. When Clock Enable is negated, the DSPI is in a dormant state, but the memory mapped registers are still accessible. Certain read or write operations have a different effect when the DSPI is in the module disable mode. Reading the RX FIFO Pop Register does not change the state of the RX FIFO. Likewise, writing to the TX FIFO Push Register does not change the state of the TX FIFO. Clearing either of the FIFOs has no effect in the module disable mode. Changes to the DIS_TXF and DIS_RXF fields of the MCR have no effect in the module disable mode. In the module disable mode, all status bits and register flags in the DSPI return the correct values when read, but writing to them has no effect. Writing to the TCR during module disable mode has no effect. Interrupt and DMA request signals cannot be cleared while in the module disable mode.

50.5 Initialization/Application Information

This section describes how to initialize the DSPI module.

50.5.1 How to Manage DSPI Queues

The queues are not part of the DSPI, but the DSPI includes features in support of queue management. Queues are primarily supported in SPI Configuration.

1. When DSPI executes last command word from a queue, the EOQ bit in the command word is set to indicate to the DSPI that this is the last entry in the queue.
2. At the end of the transfer, corresponding to the command word with EOQ set is sampled, the EOQ flag (EOQF) in the SR is set.
3. The setting of the EOQF flag disables serial transmission and reception of data, putting the DSPI in the STOPPED state. The TXRXS bit is cleared to indicate the STOPPED state.
4. The DMA can continue to fill TX FIFO until it is full or step 5 occurs.
5. Disable DSPI DMA transfers by disabling the DMA enable request for the DMA channel assigned to TX FIFO and RX FIFO. This is done by clearing the corresponding DMA enable request bits in the DMA Controller.
6. Ensure all received data in RX FIFO has been transferred to memory receive queue by reading the RXCNT in SR or by checking RFDF in the SR after each read operation of the POPR.
7. Modify DMA descriptor of TX and RX channels for new queues
8. Flush TX FIFO by writing a '1' to the CLR_TXF bit in the MCR. Flush RX FIFO by writing a '1' to the CLR_RXF bit in the MCR.
9. Clear transfer count either by setting CTCNT bit in the command word of the first entry in the new queue or via CPU writing directly to SPI_TCNT field in the TCR.
10. Enable DMA channel by enabling the DMA enable request for the DMA channel assigned to the DSPI TX FIFO, and RX FIFO by setting the corresponding DMA set enable request bit.
11. Enable serial transmission and serial reception of data by clearing the EOQF bit.

50.5.2 Switching Master and Slave Mode

When changing modes in the DSPI, follow the steps below to guarantee proper operation.

1. Halt the DSPI by setting MCR[HALT].
2. Clear the transmit and receive FIFOs by writing a 1 to the CLR_TXF and CLR_RXF bits in MCR.
3. Set the appropriate mode in MCR[MSTR] and enable the DSPI by clearing MCR[HALT].

50.5.3 Baud Rate Settings

The following table shows the baud rate that is generated based on the combination of the baud rate prescaler PBR and the baud rate scaler BR in the CTAR registers. The values calculated assume a 100 MHz system frequency and the double baud rate DBR bit is clear.

NOTE

> The clock frequency mentioned above is given as an example in this chapter. Refer to the clocking chapter for the frequency used to drive this module in the device.

Table 50-113. Baud Rate Values (bps)

		Baud Rate Divider Prescaler Values			
		2	3	5	7
Baud Rate Scaler Values	2	25.0M	16.7M	10.0M	7.14M
	4	12.5M	8.33M	5.00M	3.57M
	6	8.33M	5.56M	3.33M	2.38M
	8	6.25M	4.17M	2.50M	1.79M
	16	3.12M	2.08M	1.25M	893k
	32	1.56M	1.04M	625k	446k
	64	781k	521k	312k	223k
	128	391k	260k	156k	112k
	256	195k	130k	78.1k	55.8k
	512	97.7k	65.1k	39.1k	27.9k
	1024	48.8k	32.6k	19.5k	14.0k
	2048	24.4k	16.3k	9.77k	6.98k
	4096	12.2k	8.14k	4.88k	3.49k
	8192	6.10k	4.07k	2.44k	1.74k
	16384	3.05k	2.04k	1.22k	872
	32768	1.53k	1.02k	610	436

50.5.4 Delay Settings

The following table shows the values for the Delay after Transfer (t_{DT}) and CS to SCK Delay (T_{CSC}) that can be generated based on the prescaler values and the scaler values set in the CTAR registers. The values calculated assume a 100 MHz system frequency.

NOTE

The clock frequency mentioned above is given as an example in this chapter. Refer to the clocking chapter for the frequency used to drive this module in the device.

Table 50-114. Delay Values

		Delay Prescaler Values			
		1	3	5	7
Delay Scaler Values	2	20.0 ns	60.0 ns	100.0 ns	140.0 ns
	4	40.0 ns	120.0 ns	200.0 ns	280.0 ns
	8	80.0 ns	240.0 ns	400.0 ns	560.0 ns
	16	160.0 ns	480.0 ns	800.0 ns	1.1 μs
	32	320.0 ns	960.0 ns	1.6 μs	2.2 μs
	64	640.0 ns	1.9 μs	3.2 μs	4.5 μs
	128	1.3 μs	3.8 μs	6.4 μs	9.0 μs
	256	2.6 μs	7.7 μs	12.8 μs	17.9 μs
	512	5.1 μs	15.4 μs	25.6 μs	35.8 μs
	1024	10.2 μs	30.7 μs	51.2 μs	71.7 μs
	2048	20.5 μs	61.4 μs	102.4 μs	143.4 μs
	4096	41.0 μs	122.9 μs	204.8 μs	286.7 μs
	8192	81.9 μs	245.8 μs	409.6 μs	573.4 μs
	16384	163.8 μs	491.5 μs	819.2 μs	1.1 ms
	32768	327.7 μs	983.0 μs	1.6 ms	2.3 ms
	65536	655.4 μs	2.0 ms	3.3 ms	4.6 ms

50.5.5 Calculation of FIFO Pointer Addresses

Complete visibility of the TX and RX FIFO contents is available through the FIFO registers, and valid entries can be identified through a memory mapped pointer and a memory mapped counter for each FIFO. The pointer to the first-in entry in each FIFO is memory mapped. For the TX FIFO the first-in pointer is the Transmit Next Pointer (TXNXTPTR). For the RX FIFO the first-in pointer is the Pop Next Pointer (POPNXTPTR). The following figure illustrates the concept of first-in and last-in FIFO entries along with the FIFO Counter. The TX FIFO is chosen for the illustration, but the concepts carry over to the RX FIFO. See Transmit First In First Out (TX FIFO) Buffering Mechanism and Receive First In First Out (RX FIFO) Buffering Mechanism for details on the FIFO operation.

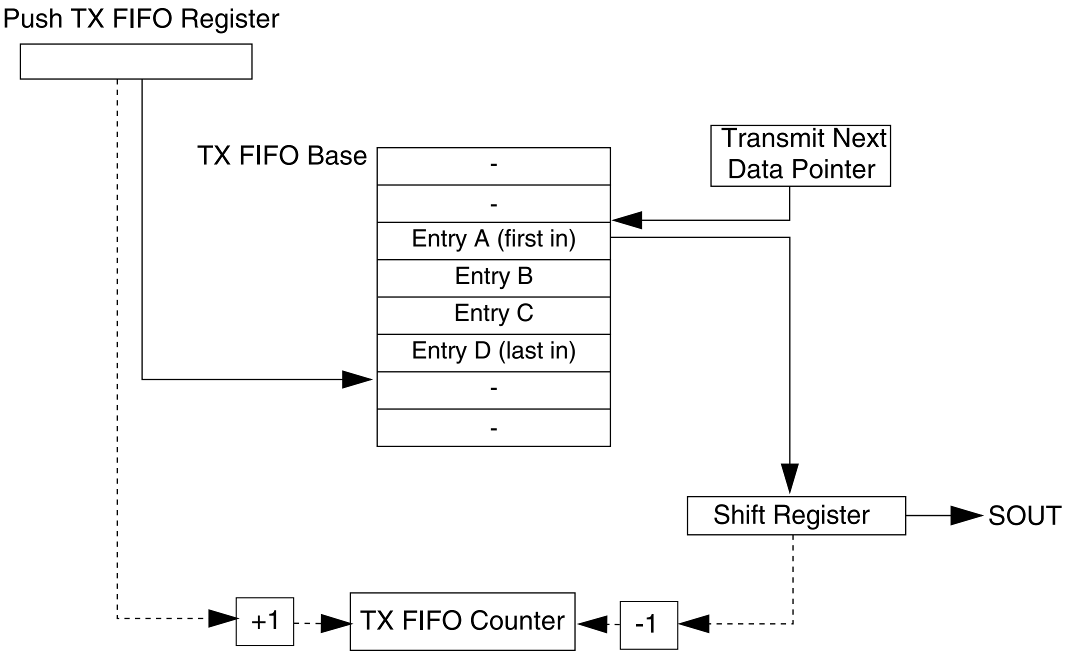


Figure 50-100. TX FIFO Pointers and Counter

50.5.5.1 Address Calculation for the First-in Entry and Last-in Entry in the TX FIFO

The memory address of the first-in entry in the TX FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{TXFIFOBase} + (4 \times \text{TXNXTPTR})$$

The memory address of the last-in entry in the TX FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{TXFIFOBase} + 4 \times (\text{TXCTR} + \text{TXNXTPTR} - 1)\text{mod}(\text{TXFIFOdepth})$$

TX FIFO Base - Base address of TX FIFO
TXCTR - TX FIFO Counter
TXNXTPTR - Transmit Next Pointer
TX FIFO Depth - Transmit FIFO depth, implementation specific

50.5.5.2 Address Calculation for the First-in Entry and Last-in Entry in the RX FIFO

The memory address of the first-in entry in the RX FIFO is computed by the following equation:

$$\text{First-in EntryAddress} = \text{RX FIFOBase} + (4 \times \text{POPNXTPTR})$$

The memory address of the last-in entry in the RX FIFO is computed by the following equation:

$$\text{Last-inEntryaddress} = \text{RX FIFO Base} + 4 \times (\text{RXCTR} + \text{POPNXTPTR} - 1)\text{mod}(\text{RXFIFOdepth})$$

RX FIFO Base - Base address of RX FIFO
RXCTR - RX FIFO counter
POPNXTPTR - Pop Next Pointer
RX FIFO Depth - Receive FIFO depth, implementation specific

Chapter 51
Inter-Integrated Circuit (I2C)

51.1 Introduction

NOTE

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

The inter-integrated circuit (I^2C, I2C, or IIC) module provides a method of communication between a number of devices. The interface is designed to operate up to 100 kbit/s with maximum bus loading and timing. The device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF. The I2C module also complies with the *System Management Bus (SMBus) Specification, version 2*.

51.1.1 Features

The I2C module has the following features:

- Compatible with *The I^2C-Bus Specification*
- Multimaster operation
- Software programmable for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- START and STOP signal generation and detection
- Repeated START signal generation and detection
- Acknowledge bit generation and detection
- Bus busy detection
- General call recognition

- 10-bit address extension
- Support for *System Management Bus (SMBus) Specification, version 2*
- Programmable glitch input filter
- Low power mode wakeup on slave address match
- Range slave address support
- DMA support

51.1.2 Modes of Operation

The I2C module's operation in various low power modes is as follows:

- Run mode: This is the basic mode of operation. To conserve power in this mode, disable the module.
- Wait mode: The module continues to operate when the core is in wait mode and can provide a wakeup interrupt.
- Stop mode: The module is inactive in stop mode for reduced power consumption, except that address matching is enabled in stop mode. The STOP instruction does not affect the I2C module's register states.

51.1.3 Block Diagram

The following figure is a functional block diagram of the I2C module.

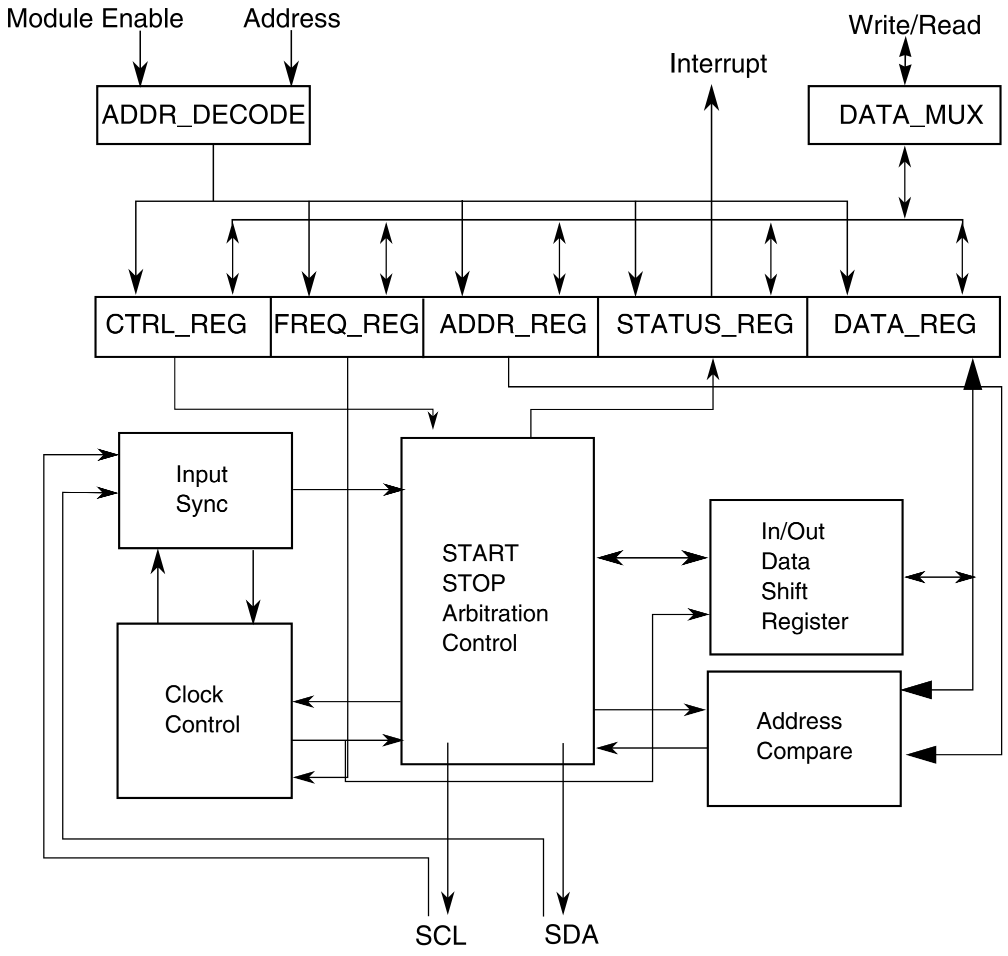


Figure 51-1. I2C Functional Block Diagram

51.2 I^{2}C Signal Descriptions

The signal properties of I^{2}C are shown in the following table.

Table 51-1. I^{2}C Signal Descriptions

Signal	Description	I/O
SCL	Bidirectional serial clock line of the I^{2}C system.	I/O
SDA	Bidirectional serial data line of the I^{2}C system.	I/O

51.3 Memory Map and Register Descriptions

This section describes in detail all I2C registers accessible to the end user.

I2C memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_6000	I2C Address Register 1 (I2C0_A1)	8	R/W	00h	51.3.1/ 1575
4006_6001	I2C Frequency Divider register (I2C0_F)	8	R/W	00h	51.3.2/ 1575
4006_6002	I2C Control Register 1 (I2C0_C1)	8	R/W	00h	51.3.3/ 1576
4006_6003	I2C Status Register (I2C0_S)	8	R/W	80h	51.3.4/ 1578
4006_6004	I2C Data I/O register (I2C0_D)	8	R/W	00h	51.3.5/ 1580
4006_6005	I2C Control Register 2 (I2C0_C2)	8	R/W	00h	51.3.6/ 1581
4006_6006	I2C Programmable Input Glitch Filter register (I2C0_FLT)	8	R/W	00h	51.3.7/ 1582
4006_6007	I2C Range Address register (I2C0_RA)	8	R/W	00h	51.3.8/ 1582
4006_6008	I2C SMBus Control and Status register (I2C0_SMB)	8	R/W	00h	51.3.9/ 1583
4006_6009	I2C Address Register 2 (I2C0_A2)	8	R/W	C2h	51.3.10/ 1584
4006_600A	I2C SCL Low Timeout Register High (I2C0_SLTH)	8	R/W	00h	51.3.11/ 1585
4006_600B	I2C SCL Low Timeout Register Low (I2C0_SLTL)	8	R/W	00h	51.3.12/ 1585
4006_7000	I2C Address Register 1 (I2C1_A1)	8	R/W	00h	51.3.1/ 1575
4006_7001	I2C Frequency Divider register (I2C1_F)	8	R/W	00h	51.3.2/ 1575
4006_7002	I2C Control Register 1 (I2C1_C1)	8	R/W	00h	51.3.3/ 1576
4006_7003	I2C Status Register (I2C1_S)	8	R/W	80h	51.3.4/ 1578
4006_7004	I2C Data I/O register (I2C1_D)	8	R/W	00h	51.3.5/ 1580
4006_7005	I2C Control Register 2 (I2C1_C2)	8	R/W	00h	51.3.6/ 1581
4006_7006	I2C Programmable Input Glitch Filter register (I2C1_FLT)	8	R/W	00h	51.3.7/ 1582
4006_7007	I2C Range Address register (I2C1_RA)	8	R/W	00h	51.3.8/ 1582

Table continues on the next page...

I2C memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_7008	I2C SMBus Control and Status register (I2C1_SMB)	8	R/W	00h	51.3.9/ 1583
4006_7009	I2C Address Register 2 (I2C1_A2)	8	R/W	C2h	51.3.10/ 1584
4006_700A	I2C SCL Low Timeout Register High (I2C1_SLTH)	8	R/W	00h	51.3.11/ 1585
4006_700B	I2C SCL Low Timeout Register Low (I2C1_SLTL)	8	R/W	00h	51.3.12/ 1585

51.3.1 I2C Address Register 1 (I2Cx_A1)

This register contains the slave address to be used by the I2C module.

Addresses: I2C0_A1 is 4006_6000h base + 0h offset = 4006_6000h
I2C1_A1 is 4006_7000h base + 0h offset = 4006_7000h

Bit	7	6	5	4	3	2	1	0
Read	AD[7:1]							0
Write								▨
Reset	0	0	0	0	0	0	0	0

I2Cx_A1 field descriptions

Field	Description
7–1 AD[7:1]	Address Contains the primary slave address used by the I2C module when it is addressed as a slave. This field is used in the 7-bit address scheme and the lower seven bits in the 10-bit address scheme.
0 Reserved	This read-only field is reserved and always has the value zero.

51.3.2 I2C Frequency Divider register (I2Cx_F)

Addresses: I2C0_F is 4006_6000h base + 1h offset = 4006_6001h
I2C1_F is 4006_7000h base + 1h offset = 4006_7001h

Bit	7	6	5	4	3	2	1	0
Read	MULT		ICR					
Write								
Reset	0	0	0	0	0	0	0	0

I2Cx_F field descriptions

Field	Description
7–6 MULT	The MULT bits define the multiplier factor mul. This factor is used along with the SCL divider to generate the I2C baud rate. 00 mul = 1 01 mul = 2 10 mul = 4 11 Reserved

5–0
ICR
	Clock rate
	Prescales the bus clock for bit rate selection. This field and the MULT field determine the I2C baud rate, the SDA hold time, the SCL start hold time, and the SCL stop hold time. For a list of values corresponding to each ICR setting, see I2C Divider and Hold Values.
	The SCL divider multiplied by multiplier factor (mul) determines the I2C baud rate.
	`I2C baud rate = bus speed (Hz)/(mul × SCL divider)`
	The SDA hold time is the delay from the falling edge of SCL (I2C clock) to the changing of SDA (I2C data).
	`SDA hold time = bus period (s) × mul × SDA hold value`
	The SCL start hold time is the delay from the falling edge of SDA (I2C data) while SCL is high (start condition) to the falling edge of SCL (I2C clock).
	`SCL start hold time = bus period (s) × mul × SCL start hold value`
	The SCL stop hold time is the delay from the rising edge of SCL (I2C clock) to the rising edge of SDA (I2C data) while SCL is high (stop condition).
	`SCL stop hold time = bus period (s) × mul × SCL stop hold value`

51.3.3 I2C Control Register 1 (I2Cx_C1)

Addresses: I2C0_C1 is 4006_6000h base + 2h offset = 4006_6002h
I2C1_C1 is 4006_7000h base + 2h offset = 4006_7002h

Bit	7	6	5	4	3	2	1	0
Read	IICEN	IICIE	MST	TX	TXAK	0	WUEN	DMAEN
Write						RSTA		
Reset	0	0	0	0	0	0	0	0

I2Cx_C1 field descriptions

Field	Description
7 IICEN	I2C enable Enables I2C module operation.

Table continues on the next page...

I2Cx_C1 field descriptions (continued)

Field	Description
	0 Disabled 1 Enabled
6 IICIE	I2C interrupt enable Enables I2C interrupt requests. 0 Disabled 1 Enabled
5 MST	Master mode select When the MST bit is changed from a 0 to a 1, a START signal is generated on the bus and master mode is selected. When this bit changes from a 1 to a 0, a STOP signal is generated and the mode of operation changes from master to slave. 0 Slave mode 1 Master mode
4 TX	Transmit mode select Selects the direction of master and slave transfers. In master mode this bit must be set according to the type of transfer required. Therefore, for address cycles, this bit is always set. When addressed as a slave this bit must be set by software according to the SRW bit in the status register. 0 Receive 1 Transmit
3 TXAK	Transmit acknowledge enable Specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers. The value of the FACK bit affects NACK/ACK generation. 0 An acknowledge signal is sent to the bus on the following (if FACK is cleared) or current (if FACK is set) receiving byte. 1 No acknowledge signal is sent to the bus on the following (if FACK is cleared) or current (if FACK is set) receiving data byte. **NOTE:** SCL is held low until TXAK is written.
2 RSTA	Repeat START Writing a one to this bit generates a repeated START condition provided it is the current master. This bit will always be read as zero. Attempting a repeat at the wrong time results in loss of arbitration.
1 WUEN	Wakeup enable The I2C module can wake the MCU from low power mode with no peripheral bus running when slave address matching occurs. 0 Normal operation. No interrupt generated when address matching in low power mode. 1 Enables the wakeup function in low power mode.
0 DMAEN	DMA enable The DMAEN bit enables or disables the DMA function.

Table continues on the next page...

I2Cx_C1 field descriptions (continued)

Field	Description
	0 All DMA signalling disabled. 1 DMA transfer is enabled and the following conditions trigger the DMA request: • While FACK = 0, a data byte is received, either address or data is transmitted. (ACK/NACK automatic) • While FACK = 0, the first byte received matches the A1 register or is general call address. If any address matching occurs, IAAS and TCF are set. If the direction of transfer is known from master to slave, then it is not required to check the SRW. With this assumption, DMA can also be used in this case. In other cases, if the master reads data from the slave, then it is required to rewrite the C1 register operation. With this assumption, DMA cannot be used. When FACK = 1, an address or a data byte is transmitted.

51.3.4 I2C Status Register (I2Cx_S)

Addresses: I2C0_S is 4006_6000h base + 3h offset = 4006_6003h

I2C1_S is 4006_7000h base + 3h offset = 4006_7003h

Bit	7	6	5	4	3	2	1	0
Read	TCF	IAAS	BUSY	ARBL	RAM	SRW	IICIF	RXAK
Write				w1c			w1c	
Reset	1	0	0	0	0	0	0	0

I2Cx_S field descriptions

Field	Description
7 TCF	Transfer complete flag This bit sets on the completion of a byte and acknowledge bit transfer. This bit is valid only during or immediately following a transfer to or from the I2C module. The TCF bit is cleared by reading the I2C data register in receive mode or by writing to the I2C data register in transmit mode. 0 Transfer in progress 1 Transfer complete
6 IAAS	Addressed as a slave This bit is set by one of the following conditions: • The calling address matches the programmed slave primary address in the A1 register or range address in the RA register (which must be set to a nonzero value). • GCAEN is set and a general call is received. • SIICAEN is set and the calling address matches the second programmed slave address. • ALERTEN is set and an SMBus alert response address is received • RMEN is set and an address is received that is within the range between the values of the A1 and RA registers. This bit sets before the ACK bit. The CPU must check the SRW bit and set TX/RX accordingly. Writing the C1 register with any value clears this bit.

Table continues on the next page...

I2Cx_S field descriptions (continued)

Field	Description
	0 Not addressed 1 Addressed as a slave
5 BUSY	Bus busy Indicates the status of the bus regardless of slave or master mode. This bit is set when a START signal is detected and cleared when a STOP signal is detected. 0 Bus is idle 1 Bus is busy
4 ARBL	Arbitration lost This bit is set by hardware when the arbitration procedure is lost. The ARBL bit must be cleared by software, by writing a one to it. 0 Standard bus operation. 1 Loss of arbitration.
3 RAM	Range address match This bit is set by any of the following conditions: • Any nonzero calling address is received that matches the address in the RA register. • The RMEN bit is set and the calling address is within the range of values of the A1 and RA registers. Writing the C1 register with any value clears this bit. 0 Not addressed 1 Addressed as a slave
2 SRW	Slave read/write When addressed as a slave, SRW indicates the value of the $R/\overline{W}$ command bit of the calling address sent to the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IICIF	Interrupt flag This bit sets when an interrupt is pending. This bit must be cleared by software or by writing a 1 to it in the interrupt routine. One of the following events can set this bit: • One byte transfer including ACK/NACK bit completes if FACK = 0 • One byte transfer excluding ACK/NACK bit completes if FACK = 1. An ACK or NACK is sent on the bus by writing 0 or 1 to TXAK after this bit is set in receive mode • Match of slave address to calling address including primary slave address, range slave address, alert response address, second slave address, or general call address. • Arbitration lost • In SMBus mode, any timeouts except SCL and SDA high timeouts 0 No interrupt pending 1 Interrupt pending
0 RXAK	Receive acknowledge

Table continues on the next page...

I2Cx_S field descriptions (continued)

Field	Description
	0 Acknowledge signal was received after the completion of one byte of data transmission on the bus 1 No acknowledge signal detected

51.3.5 I2C Data I/O register (I2Cx_D)

Addresses: I2C0_D is 4006_6000h base + 4h offset = 4006_6004h
I2C1_D is 4006_7000h base + 4h offset = 4006_7004h

Bit	7	6	5	4	3	2	1	0
Read Write	DATA							
Reset	0	0	0	0	0	0	0	0

I2Cx_D field descriptions

Field	Description
7–0 DATA	Data In master transmit mode, when data is written to this register, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates receiving of the next byte of data. **NOTE:** When making the transition out of master receive mode, switch the I2C mode before reading the Data register to prevent an inadvertent initiation of a master receive data transfer. In slave mode, the same functions are available after an address match occurs. The C1[TX] bit must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For example, if the I2C module is configured for master transmit but a master receive is desired, reading the Data register does not initiate the receive. Reading the Data register returns the last byte received while the I2C module is configured in master receive or slave receive mode. The Data register does not reflect every byte that is transmitted on the I2C bus, and neither can software verify that a byte has been written to the Data register correctly by reading it back. In master transmit mode, the first byte of data written to the Data register following assertion of MST (start bit) or assertion of RSTA (repeated start bit) is used for the address transfer and must consist of the calling address (in bits 7-1) concatenated with the required R/$\overline{W}$ bit (in position bit 0).

51.3.6 I2C Control Register 2 (I2Cx_C2)

Addresses: I2C0_C2 is 4006_6000h base + 5h offset = 4006_6005h
I2C1_C2 is 4006_7000h base + 5h offset = 4006_7005h

Bit	7	6	5	4	3	2	1	0
Read Write	GCAEN	ADEXT	HDRS	SBRC	RMEN	AD[10:8]		
Reset	0	0	0	0	0	0	0	0

I2Cx_C2 field descriptions

Field	Description
7 GCAEN	General call address enable Enables general call address. 0 Disabled 1 Enabled
6 ADEXT	Address extension Controls the number of bits used for the slave address. 0 7-bit address scheme 1 10-bit address scheme
5 HDRS	High drive select Controls the drive capability of the I2C pads. 0 Normal drive mode 1 High drive mode
4 SBRC	Slave baud rate control Enables independent slave mode baud rate at max frequency. This forces clock stretching on SCL in very fast I2C modes. 0 The slave baud rate follows the master baud rate and clock stretching may occur 1 Slave baud rate is independent of the master baud rate
3 RMEN	Range address matching enable This bit controls slave address matching for addresses between the values of the A1 and RA registers. When this bit is set, a slave address match occurs for any address greater than the value of the A1 register and less than or equal to the value of the RA register. 0 Range mode disabled. No address match occurs for an address within the range of values of the A1 and RA registers. 1 Range mode enabled. Address matching occurs when a slave receives an address within the range of values of the A1 and RA registers.
2–0 AD[10:8]	Slave address Contains the upper three bits of the slave address in the 10-bit address scheme. This field is valid only when the ADEXT bit is set.

51.3.7 I2C Programmable Input Glitch Filter register (I2Cx_FLT)

Addresses: I2C0_FLT is 4006_6000h base + 6h offset = 4006_6006h
I2C1_FLT is 4006_7000h base + 6h offset = 4006_7006h

Bit	7	6	5	4	3	2	1	0
Read	0	0		FLT				
Write	▨	▨	▨					
Reset	0	0	0	0	0	0	0	0

I2Cx_FLT field descriptions

Field	Description
7 Reserved	This read-only field is reserved and always has the value zero.
6–5 Reserved	This read-only field is reserved and always has the value zero.
4–0 FLT	I2C programmable filter factor Controls the width of the glitch, in terms of bus clock cycles, that the filter must absorb. For any glitch whose size is less than or equal to this width setting, the filter does not allow the glitch to pass. 00h No filter/bypass 01-1Fh Filter glitches up to width of *n* bus clock cycles, where *n*=1-31d

51.3.8 I2C Range Address register (I2Cx_RA)

Addresses: I2C0_RA is 4006_6000h base + 7h offset = 4006_6007h
I2C1_RA is 4006_7000h base + 7h offset = 4006_7007h

Bit	7	6	5	4	3	2	1	0
Read	RAD							0
Write								▨
Reset	0	0	0	0	0	0	0	0

I2Cx_RA field descriptions

Field	Description
7–1 RAD	Range slave address This field contains the slave address to be used by the I2C module. The field is used in the 7-bit address scheme. Any nonzero write enables this register. This register's use is similar to that of the A1 register, but in addition this register can be considered a maximum boundary in range matching mode.
0 Reserved	This read-only field is reserved and always has the value zero.

51.3.9 I2C SMBus Control and Status register (I2Cx_SMB)

NOTE

When the SCL and SDA signals are held high for a length of time greater than the high timeout period, the SHTF1 flag sets. Before reaching this threshold, while the system is detecting how long these signals are being held high, a master assumes that the bus is free. However, the SHTF1 bit rises in the bus transmission process with the idle bus state.

NOTE

When the TCKSEL bit is set, there is no meaning to monitor the SHTF1 bit because the bus speed is too high to match the protocol of SMBus.

Addresses: I2C0_SMB is 4006_6000h base + 8h offset = 4006_6008h

I2C1_SMB is 4006_7000h base + 8h offset = 4006_7008h

Bit	7	6	5	4	3	2	1	0
Read	FACK	ALERTEN	SIICAEN	TCKSEL	SLTF	SHTF1	SHTF2	SHTF2IE
Write					w1c		w1c	
Reset	0	0	0	0	0	0	0	0

I2Cx_SMB field descriptions

Field	Description
7 FACK	Fast NACK/ACK enable For SMBus packet error checking, the CPU must be able to issue an ACK or NACK according to the result of receiving data byte. 0 An ACK or NACK is sent on the following receiving data byte 1 Writing 0 to TXAK after receiving a data byte generates an ACK. Writing 1 to TXAK after receiving a data byte generates a NACK.
6 ALERTEN	SMBus alert response address enable Enables or disables SMBus alert response address matching. **NOTE:** After the host responds to a device that used the alert response address, you must use software to put the device's address on the bus. The alert protocol is described in the SMBus specification. 0 SMBus alert response address matching is disabled 1 SMBus alert response address matching is enabled
5 SIICAEN	Second I2C address enable Enables or disables SMBus device default address.

Table continues on the next page...

I2Cx_SMB field descriptions (continued)

Field	Description
	0 I2C address register 2 matching is disabled 1 I2C address register 2 matching is enabled
4 TCKSEL	Timeout counter clock select Selects the clock source of the timeout counter. 0 Timeout counter counts at the frequency of the bus clock / 64 1 Timeout counter counts at the frequency of the bus clock
3 SLTF	SCL low timeout flag This bit is set when the SLT register (consisting of the SLTH and SLTL registers) is loaded with a non-zero value (LoValue) and an SCL low timeout occurs. Software clears this bit by writing a logic 1 to it. **NOTE:** The low timeout function is disabled when the SLT register's value is zero. 0 No low timeout occurs 1 Low timeout occurs
2 SHTF1	SCL high timeout flag 1 This read-only bit sets when SCL and SDA are held high more than clock × LoValue / 512, which indicates the bus is free. This bit is cleared automatically. 0 No SCL high and SDA high timeout occurs 1 SCL high and SDA high timeout occurs
1 SHTF2	SCL high timeout flag 2 This bit sets when SCL is held high and SDA is held low more than clock × LoValue/512. Software clears this bit by writing a 1 to it. 0 No SCL high and SDA low timeout occurs 1 SCL high and SDA low timeout occurs
0 SHTF2IE	SHTF2 interrupt enable Enables SCL high and SDA low timeout interrupt. 0 SHTF2 interrupt is disabled 1 SHTF2 interrupt is enabled

51.3.10 I2C Address Register 2 (I2Cx_A2)

Addresses: I2C0_A2 is 4006_6000h base + 9h offset = 4006_6009h
I2C1_A2 is 4006_7000h base + 9h offset = 4006_7009h

Bit	7	6	5	4	3	2	1	0
Read	SAD							0
Write								▨
Reset	1	1	0	0	0	0	1	0

I2Cx_A2 field descriptions

Field	Description
7–1 SAD	SMBus address Contains the slave address used by the SMBus. This field is used on the device default address or other related addresses.
0 Reserved	This read-only field is reserved and always has the value zero.

51.3.11 I2C SCL Low Timeout Register High (I2Cx_SLTH)

Addresses: I2C0_SLTH is 4006_6000h base + Ah offset = 4006_600Ah
I2C1_SLTH is 4006_7000h base + Ah offset = 4006_700Ah

Bit	7	6	5	4	3	2	1	0
Read Write	SSLT[15:8]							
Reset	0	0	0	0	0	0	0	0

I2Cx_SLTH field descriptions

Field	Description
7–0 SSLT[15:8]	Most significant byte of SCL low timeout value that determines the timeout period of SCL low.

51.3.12 I2C SCL Low Timeout Register Low (I2Cx_SLTL)

Addresses: I2C0_SLTL is 4006_6000h base + Bh offset = 4006_600Bh
I2C1_SLTL is 4006_7000h base + Bh offset = 4006_700Bh

Bit	7	6	5	4	3	2	1	0
Read Write	SSLT[7:0]							
Reset	0	0	0	0	0	0	0	0

I2Cx_SLTL field descriptions

Field	Description
7–0 SSLT[7:0]	Least significant byte of SCL low timeout value that determines the timeout period of SCL low.

51.4 Functional Description

This section provides a comprehensive functional description of the I2C module.

51.4.1 I2C Protocol

The I2C bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfers. All devices connected to it must have open drain or open collector outputs. A logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors depends on the system.

Normally, a standard instance of communication is composed of four parts:

1. START signal
2. Slave address transmission
3. Data transfer
4. STOP signal

The STOP signal should not be confused with the CPU STOP instruction. The following figure illustrates I2C bus system communication.

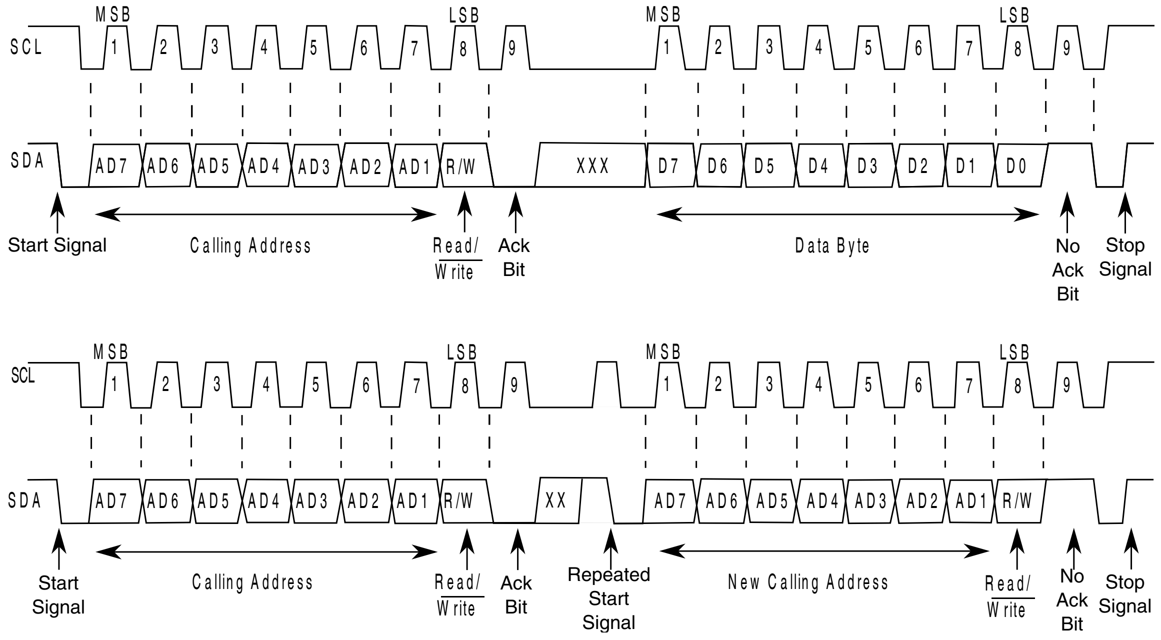


Figure 51-38. I2C Bus Transmission Signals

51.4.1.1 START Signal

The bus is free when no master device is engaging the bus (both SCL and SDA are high). When the bus is free, a master may initiate communication by sending a START signal. A START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer might contain several bytes of data) and brings all slaves out of their idle states.

51.4.1.2 Slave Address Transmission

Immediately after the START signal, the first byte of a data transfer is the slave address transmitted by the master. This address is a 7-bit calling address followed by an $R/\overline{W}$ bit. The $R/\overline{W}$ bit tells the slave the desired direction of data transfer.

- 1 = Read transfer: The slave transmits data to the master
- 0 = Write transfer: The master transmits data to the slave

Only the slave with a calling address that matches the one transmitted by the master responds by sending an acknowledge bit. The slave sends the acknowledge bit by pulling SDA low at the ninth clock.

No two slaves in the system can have the same address. If the I2C module is the master, it must not transmit an address that is equal to its own slave address. The I2C module cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the I2C module reverts to slave mode and operates correctly even if it is being addressed by another master.

51.4.1.3 Data Transfers

When successful slave addressing is achieved, data transfer can proceed on a byte-by-byte basis in the direction specified by the $R/\overline{W}$ bit sent by the calling master.

All transfers that follow an address cycle are referred to as data transfers, even if they carry subaddress information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low. Data must be held stable while SCL is high. There is one clock pulse on SCL for each data bit, and the MSB is transferred first. Each data byte is followed by a ninth (acknowledge) bit, which is signaled from the receiving device by pulling SDA low at the ninth clock. In summary, one complete data transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master in the ninth bit, the slave must leave SDA high. The master interprets the failed acknowledgement as an unsuccessful data transfer.

If the master receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets it as an end to data transfer and releases the SDA line.

In the case of a failed acknowledgement by either the slave or master, the data transfer is aborted and the master does one of two things:

- Relinquishes the bus by generating a STOP signal.
- Commences a new call by generating a repeated START signal.

51.4.1.4 STOP Signal

The master can terminate the communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of SDA while SCL is asserted.

The master can generate a STOP signal even if the slave has generated an acknowledgement, at which point the slave must release the bus.

51.4.1.5 Repeated START Signal

The master may generate a START signal followed by a calling command without generating a STOP signal first. This action is called a repeated START. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

51.4.1.6 Arbitration Procedure

The I2C bus is a true multimaster bus that allows more than one master to be connected on it.

If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock. The bus clock's low period is equal to the longest clock low period, and the high period is equal to the shortest one among the masters.

The relative priority of the contending masters is determined by a data arbitration procedure. A bus master loses arbitration if it transmits logic level 1 while another master transmits logic level 0. The losing masters immediately switch to slave receive mode and stop driving SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets a status bit to indicate the loss of arbitration.

51.4.1.7 Clock Synchronization

Because wire AND logic is performed on SCL, a high-to-low transition on SCL affects all devices connected on the bus. The devices start counting their low period and, after a device's clock has gone low, that device holds SCL low until the clock reaches its high state. However, the change of low to high in this device clock might not change the state of SCL if another device clock is still within its low period. Therefore, the synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see the following diagram). When all applicable devices have counted off their low period, the synchronized clock SCL is released and pulled high. Afterward there is no difference between the device clocks and the state of SCL, and all devices start counting their high periods. The first device to complete its high period pulls SCL low again.

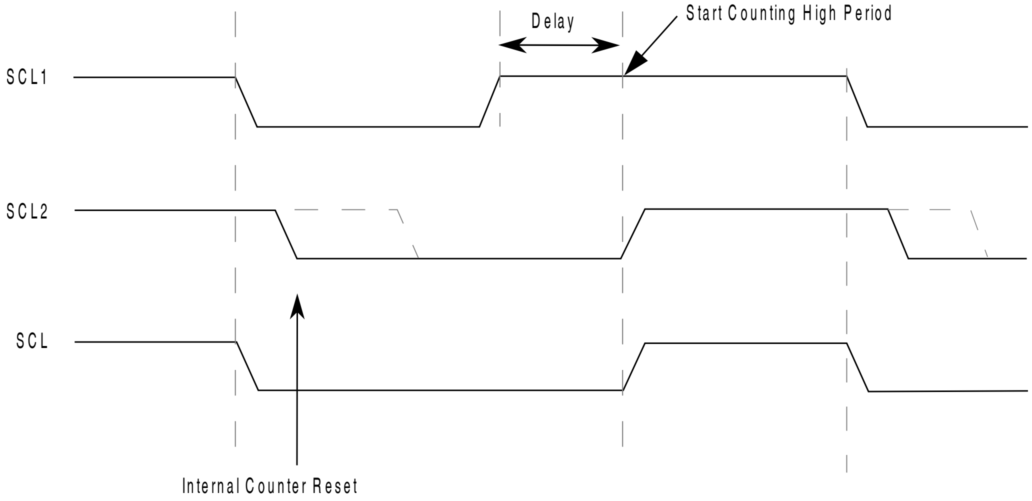


Figure 51-39. I2C Clock Synchronization

51.4.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. A slave device may hold SCL low after completing a single byte transfer (9 bits). In this case, it halts the bus clock and forces the master clock into wait states until the slave releases SCL.

51.4.1.9 Clock Stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master drives SCL low, a slave can drive SCL low for the required period and then release it. If the slave's SCL low period is greater than the master's SCL low period, the resulting SCL bus signal's low period is stretched.

51.4.1.10 I2C Divider and Hold Values

Table 51-41. I2C Divider and Hold Values

ICR (hex)	SCL Divider	SDA Hold Value	SCL Hold (Start) Value	SCL Hold (Stop) Value	ICR (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (Start) Value	SCL Hold (Stop) Value
00	20	7	6	11	20	160	17	78	81
01	22	7	7	12	21	192	17	94	97
02	24	8	8	13	22	224	33	110	113
03	26	8	9	14	23	256	33	126	129
04	28	9	10	15	24	288	49	142	145
05	30	9	11	16	25	320	49	158	161
06	34	10	13	18	26	384	65	190	193
07	40	10	16	21	27	480	65	238	241
08	28	7	10	15	28	320	33	158	161
09	32	7	12	17	29	384	33	190	193
0A	36	9	14	19	2A	448	65	222	225
0B	40	9	16	21	2B	512	65	254	257
0C	44	11	18	23	2C	576	97	286	289
0D	48	11	20	25	2D	640	97	318	321
0E	56	13	24	29	2E	768	129	382	385
0F	68	13	30	35	2F	960	129	478	481
10	48	9	18	25	30	640	65	318	321
11	56	9	22	29	31	768	65	382	385
12	64	13	26	33	32	896	129	446	449
13	72	13	30	37	33	1024	129	510	513
14	80	17	34	41	34	1152	193	574	577
15	88	17	38	45	35	1280	193	638	641
16	104	21	46	53	36	1536	257	766	769
17	128	21	58	65	37	1920	257	958	961
18	80	9	38	41	38	1280	129	638	641

Table continues on the next page...

Table 51-41. I2C Divider and Hold Values (continued)

ICR (hex)	SCL Divider	SDA Hold Value	SCL Hold (Start) Value	SCL Hold (Stop) Value		ICR (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (Start) Value	SCL Hold (Stop) Value
19	96	9	46	49		39	1536	129	766	769
1A	112	17	54	57		3A	1792	257	894	897
1B	128	17	62	65		3B	2048	257	1022	1025
1C	144	25	70	73		3C	2304	385	1150	1153
1D	160	25	78	81		3D	2560	385	1278	1281
1E	192	33	94	97		3E	3072	513	1534	1537
1F	240	33	118	121		3F	3840	513	1918	1921

51.4.2 10-bit Address

For 10-bit addressing, 0x11110 is used for the first 5 bits of the first address byte. Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing.

51.4.2.1 Master-Transmitter Addresses a Slave-Receiver

The transfer direction is not changed. When a 10-bit address follows a START condition, each slave compares the first seven bits of the first byte of the slave address (11110XX) with its own address and tests whether the eighth bit (R/$\overline{\text{W}}$ direction bit) is 0. It is possible that more than one device finds a match and generates an acknowledge (A1). Each slave that finds a match compares the eight bits of the second byte of the slave address with its own address, but only one slave finds a match and generate an acknowledge (A2). The matching slave remains addressed by the master until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

Table 51-42. Master-Transmitter Addresses Slave-Receiver with a 10-bit Address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{\text{W}}$ 0	A1	Slave address second byte AD[8:1]	A2	Data	A	...	Data	A/A	P

After the master-transmitter has sent the first byte of the 10-bit address, the slave-receiver sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

51.4.2.2 Master-Receiver Addresses a Slave-Transmitter

The transfer direction is changed after the second R/$\overline{W}$ bit. Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated START condition (Sr), a matching slave remembers that it was addressed before. This slave then checks whether the first seven bits of the first byte of the slave address following Sr are the same as they were after the START condition (S), and it tests whether the eighth (R/$\overline{W}$) bit is 1. If there is a match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3. The slave-transmitter remains addressed until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

After a repeated START condition (Sr), all other slave devices also compare the first seven bits of the first byte of the slave address with their own addresses and test the eighth (R/$\overline{W}$) bit. However, none of them are addressed because R/$\overline{W}$ = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

Table 51-43. Master-Receiver Addresses a Slave-Transmitter with a 10-bit Address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Sr	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 1	A3	Data	A	...	Data	A	P

After the master-receiver has sent the first byte of the 10-bit address, the slave-transmitter sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

51.4.3 Address Matching

All received addresses can be requested in 7-bit or 10-bit address format. The Address Register 1, which contains the I2C primary slave address, always participates in the address matching process. If the GCAEN bit is set, general call participates the address matching process. If the ALERTEN bit is set, alert response participates the address matching process. If the SIICAEN bit is set, the Address Register 2 participates in the address matching process.If the Range Address register is programmed to a nonzero value, the range address itself participates in the address matching process. If the RMEN bit is set, any address within the range of values of the Address Register 1 and the Range Address register participates in the address matching process. The Range Address register must be programmed to a value greater than the value of the Address Register 1.

When the I2C module responds to one of these addresses, it acts as a slave-receiver and the IAAS bit is set after the address cycle. Software must read the Data register after the first byte transfer to determine that the address is matched.

51.4.4 System Management Bus Specification

SMBus provides a control bus for system and power management related tasks. A system can use SMBus to pass messages to and from devices instead of tripping individual control lines. Removing the individual control lines reduces pin count. Accepting messages ensures future expandability. With the system management bus, a device can provide manufacturer information, tell the system what its model/part number is, save its state for a suspend event, report different types of errors, accept control parameters, and return its status.

51.4.4.1 Timeouts

The $T_{TIMEOUT,MIN}$ parameter allows a master or slave to conclude that a defective device is holding the clock low indefinitely or a master is intentionally trying to drive devices off the bus. It is highly recommended that a slave device release the bus (stop driving the bus and let SCL and SDA float high) when it detects any single clock held low longer than $T_{TIMEOUT,MIN}$. Devices that have detected this condition must reset their communication and be able to receive a new START condition within the timeframe of $T_{TIMEOUT,MAX}$.

SMBus defines a clock low timeout, $T_{TIMEOUT}$, of 35 ms, specifies $T_{LOW:SEXT}$ as the cumulative clock low extend time for a slave device, and specifies $T_{LOW:MEXT}$ as the cumulative clock low extend time for a master device.

51.4.4.1.1 SCL Low Timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than a timeout value condition. Devices that have detected the timeout condition must reset the communication. When the I2C module is an active master, if it detects that SMBCLK low has exceeded the value of $T_{TIMEOUT,MIN}$, it must generate a stop condition within or after the current data byte in the transfer process. When the I2C module is a slave, if it detects the $T_{TIMEOUT,MIN}$ condition, it resets its communication and is then able to receive a new START condition.

51.4.4.1.2 SCL High Timeout

When the I2C module has determined that the SMBCLK and SMBDAT signals have been high for at least $T_{HIGH:MAX}$, it assumes that the bus is idle. A HIGH timeout can occur in two ways:

1. HIGH timeout detected after a STOP condition appears on the bus
2. HIGH timeout detected after a START condition, but before a STOP condition appears on the bus

Any master detecting either scenario can assume the bus is free when SHTF1 rises. A HIGH timeout occurs in scenario 2 if a master ever detects that both the BUSY bit is high and SHTF1 is high.

When the SMBDAT signal is low and the SMBCLK signal is high for a period of time, the other kind of timeout occurs. The time period must be defined in software. SHTF2 is used as the flag when the time limit is reached. This flag is also an interrupt resource, so it also triggers IICIF.

51.4.4.1.3 CSMBCLK TIMEOUT MEXT and CSMBCLK TIMEOUT SEXT

The following figure illustrates the definition of the timeout intervals $T_{LOW:SEXT}$ and $T_{LOW:MEXT}$. When in master mode, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:MEXT}$ within a byte, where each byte is defined as START-to-ACK, ACK-to-ACK, or ACK-to-STOP. When CSMBCLK TIMEOUT MEXT occurs, SMBus MEXT rises and also triggers the SLTF.

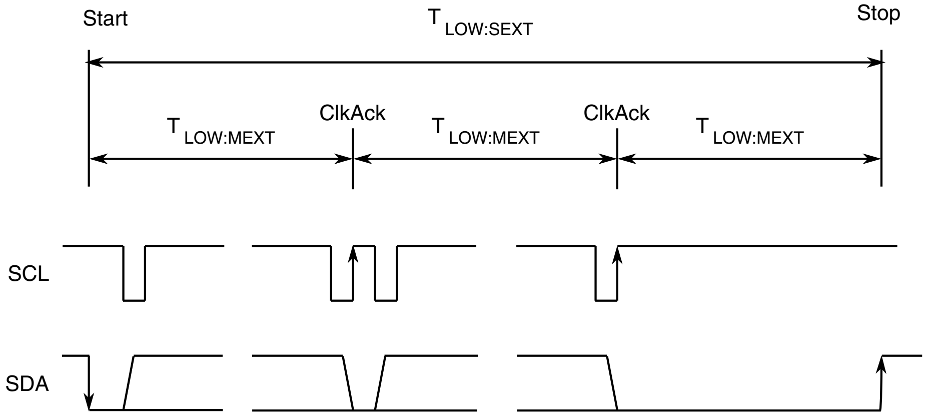


Figure 51-40. Timeout measurement intervals

A master is allowed to abort the transaction in progress to any slave that violates the $T_{LOW:SEXT}$ or $T_{TIMEOUT,MIN}$ specifications. To abort the transaction, the master issues a STOP condition at the conclusion of the byte transfer in progress. When a slave, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:SEXT}$ during any message from the initial START to the STOP. When CSMBCLK TIMEOUT SEXT occurs, SEXT rises and also triggers SLTF.

NOTE

CSMBCLK TIMEOUT SEXT and CSMBCLK TIMEOUT MEXT are optional functions that are implemented in the second step.

51.4.4.2 FAST ACK and NACK

To improve reliability and communication robustness, implementation of packet error checking (PEC) by SMBus devices is optional for SMBus devices but required for devices participating in and only during the address resolution protocol (ARP) process. The PEC is a CRC-8 error checking byte, calculated on all the message bytes. The PEC is appended to the message by the device that supplied the last data byte. If the PEC is present but not correct, a NACK is issued by the receiver. Otherwise an ACK is issued. In order to calculate the CRC-8 by software, this module can hold the SCL line low after receiving the eighth SCL (8th bit) if this byte is a data byte. So software can determine whether an ACK or NACK should be sent to the bus by setting or clearing the TXAK bit if the FACK (fast ACK/NACK enable) bit is enabled.

SMBus requires a device always to acknowledge its own address, as a mechanism to detect the presence of a removable device (such as a battery or docking station) on the bus. In addition to indicating a slave device busy condition, SMBus uses the NACK mechanism to indicate the reception of an invalid command or invalid data. Because such a condition may occur on the last byte of the transfer, SMBus devices are required to have the ability to generate the not acknowledge after the transfer of each byte and before the completion of the transaction. This requirement is important because SMBus does not provide any other resend signaling. This difference in the use of the NACK signaling has implications on the specific implementation of the SMBus port, especially in devices that handle critical system data such as the SMBus host and the SBS components.

NOTE

In the last byte of master receive slave transmit mode, the master must send a NACK to the bus, so FACK must be switched off before the last byte transmits.

51.4.5 Resets

The I2C module is disabled after a reset. The I2C module cannot cause a core reset.

51.4.6 Interrupts

The I2C module generates an interrupt when any of the events in the following table occur, provided that the IICIE bit is set. The interrupt is driven by the IICIF bit (of the I2C Status Register) and masked with the IICIE bit (of the I2C Control Register 1). The IICIF bit must be cleared (by software) by writing 1 to it in the interrupt routine. The SMBus timeouts interrupt is driven by SLTF and masked with the IICIE bit. The SLTF bit must be cleared by software by writing 1 to it in the interrupt routine. You can determine the interrupt type by reading the Status Register.

NOTE

In master receive mode, the FACK bit must be set to zero before the last byte transfer.

Table 51-44. Interrupt Summary

Interrupt Source	Status	Flag	Local Enable
Complete 1-byte transfer	TCF	IICIF	IICIE
Match of received calling address	IAAS	IICIF	IICIE
Arbitration lost	ARBL	IICIF	IICIE
SMBus SCL low timeout	SLTF	IICIF	IICIE
SMBus SCL high SDA low timeout	SHTF2	IICIF	IICIE & SHTF2IE
Wakeup from stop or wait mode	IAAS	IICIF	IICIE & WUEN

51.4.6.1 Byte Transfer Interrupt

The transfer complete flag (TCF) bit is set at the falling edge of the ninth clock to indicate the completion of a byte and acknowledgement transfer. When FACK is enabled, TCF is then set at the falling edge of 8th clock to indicate the completion of byte.

51.4.6.2 Address Detect Interrupt

When the calling address matches the programmed slave address (I2C Address Register) or when the GCAEN bit is set and a general call is received, the IAAS bit in the Status Register is set. The CPU is interrupted, provided the IICIE bit is set. The CPU must check the SRW bit and set its Tx mode accordingly.

51.4.6.3 Exit from Low-Power/Stop Modes

The slave receive input detect circuit and address matching feature are still active on low power modes (wait and stop). An asynchronous input matching slave address or general call address brings the CPU out of low power/stop mode if the interrupt is not masked. Therefore, TCF and IAAS both can trigger this interrupt.

51.4.6.4 Arbitration Lost Interrupt

The I2C is a true multimaster bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, the relative priority of the contending masters is determined by a data arbitration procedure. The I2C module asserts the arbitration-lost interrupt when it loses the data arbitration process and the ARBL bit in the Status Register is set.

Arbitration is lost in the following circumstances:

1. SDA is sampled as low when the master drives high during an address or data transmit cycle.
2. SDA is sampled as low when the master drives high during the acknowledge bit of a data receive cycle.
3. A START cycle is attempted when the bus is busy.
4. A repeated START cycle is requested in slave mode.
5. A STOP condition is detected when the master did not request it.

The ARBL bit must be cleared (by software) by writing 1 to it.

51.4.6.5 Timeout Interrupt in SMBus

When the IICIE bit is set, the I2C module asserts a timeout interrupt (outputs SLTF and SHTF2) upon detection of any of the mentioned timeout conditions, with one exception. The SCL high and SDA high TIMEOUT mechanism must not be used to influence the timeout interrupt output, because this timeout indicates an idle condition on the bus. SHTF1 rises when it matches the SCL high and SDA high TIMEOUT and falls automatically just to indicate the bus status. The SHTF2's timeout period is the same as that of SHTF1, which is short compared to that of SLTF, so another control bit, SHTF2IE, is added to enable or disable it.

51.4.7 Programmable Input Glitch Filter

An I2C glitch filter has been added outside legacy I2C modules but within the I2C package. This filter can absorb glitches on the I2C clock and data lines for the I2C module. The width of the glitch to absorb can be specified in terms of the number of (half) bus clock cycles. A single Programmable Input Glitch Filter control register is provided. Effectively, any down-up-down or up-down-up transition on the data line that occurs within the number of clock cycles programmed in this register is ignored by the I2C module. The programmer must specify the size of the glitch (in terms of bus clock cycles) for the filter to absorb and not pass.

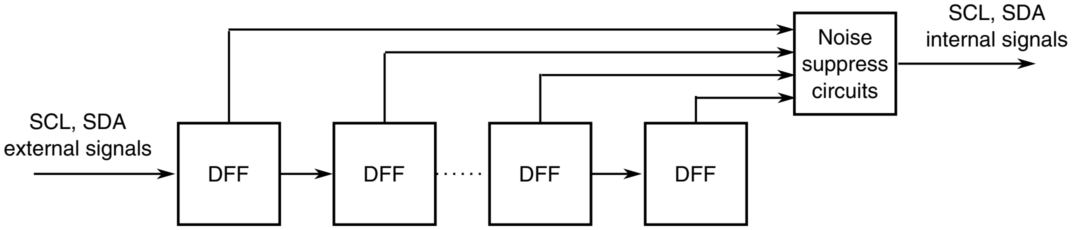


Figure 51-41. Programmable input glitch filter diagram

51.4.8 Address Matching Wakeup

When a primary, range, or general call address match occurs when the I2C module is in slave receive mode, the MCU wakes from low power mode with no peripheral bus running. After the address matching IAAS bit is set, an interrupt is sent at the end of address matching to wake the core. The IAAS bit must be cleared after the clock recovery.

NOTE

> After the system recovers and is in run mode, restart the I2C module if necessary. The SCL line is not held low until the I2C module resets after address matching. The main purpose of this feature is to wake the MCU from stop mode. Data sent on the bus that is the same as a target device address might also wake the target MCU.

51.4.9 DMA Support

If the DMAEN bit is cleared and the IICIE bit is set, an interrupt condition generates an interrupt request. If the DMAEN bit is set and the IICIE bit is set, an interrupt condition generates a DMA request instead. DMA requests are generated by the transfer complete flag (TCF).

If the DMAEN bit is set, the only arbitration lost is to another I2C module (error), and SCL low timeouts (error) generate CPU interrupts. All other events initiate a DMA transfer.

NOTE

> Before the last byte of master receive mode, TXAK must be set to send a NACK after the last byte's transfer. Therefore, the DMA must be disabled before the last byte's transfer.

NOTE

> In 10-bit address mode transmission, the addresses to send occupy 2-3 bytes. During this transfer period, the DMA must be disabled because the C1 register is written to send a repeat start or to change the transfer direction.

51.5 Initialization/Application Information

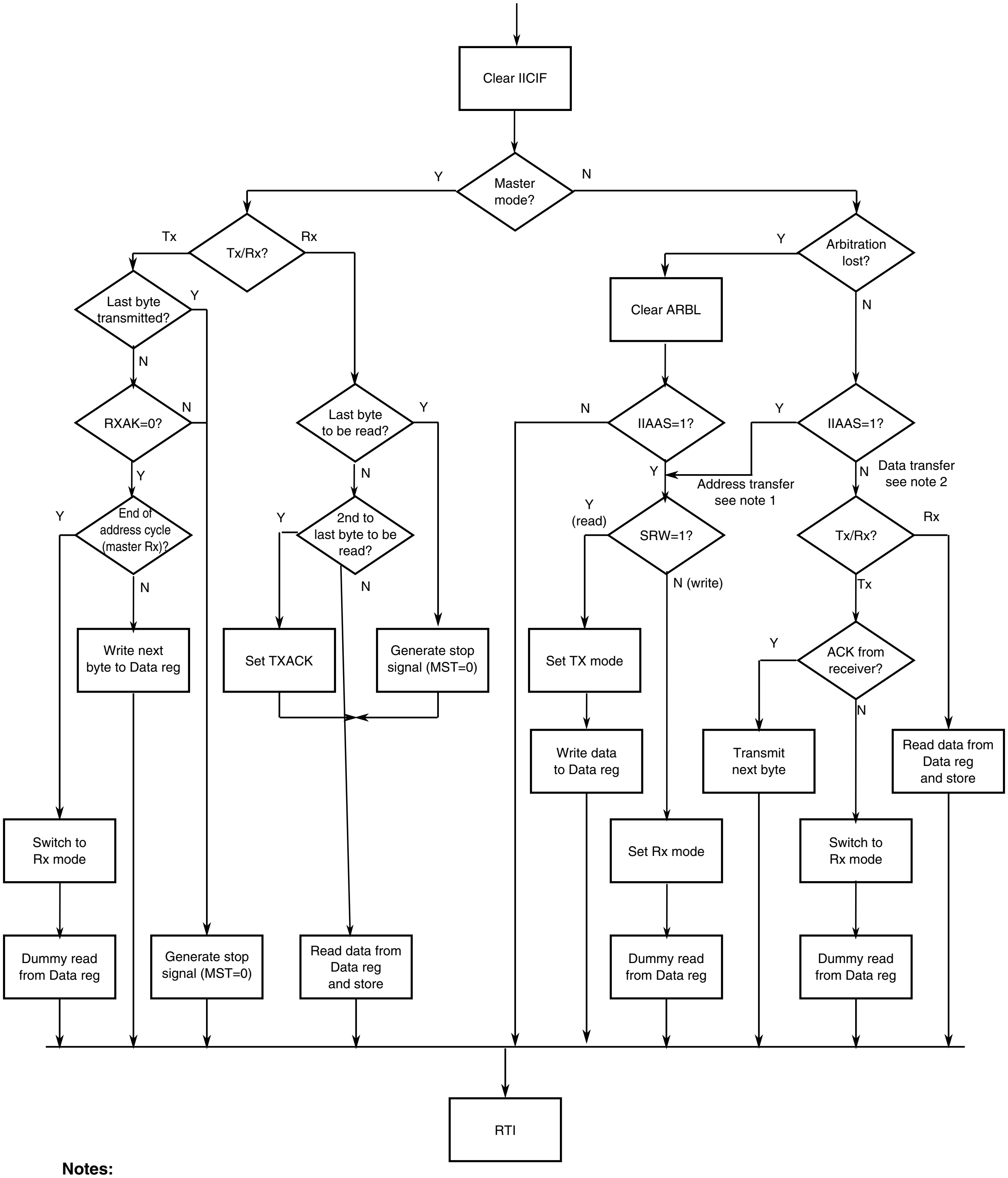
Module Initialization (Slave)

1. Write: Control Register 2
   - to enable or disable general call
   - to select 10-bit or 7-bit addressing mode
2. Write: Address Register 1 to set the slave address
3. Write: Control Register 1 to enable the I2C module and interrupts
4. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
5. Initialize RAM variables used to achieve the routine shown in the following figure

Module Initialization (Master)

1. Write: Frequency Divider register to set the I2C baud rate (example provided in this chapter)
2. Write: Control Register 1 to enable the I2C module and interrupts
3. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
4. Initialize RAM variables used to achieve the routine shown in the following figure
5. Write: Control Register 1 to enable TX
6. Write: Control Register 1 to enable MST (master mode)
7. Write: Data register with the address of the target slave (the LSB of this byte determines whether the communication is master receive or transmit)

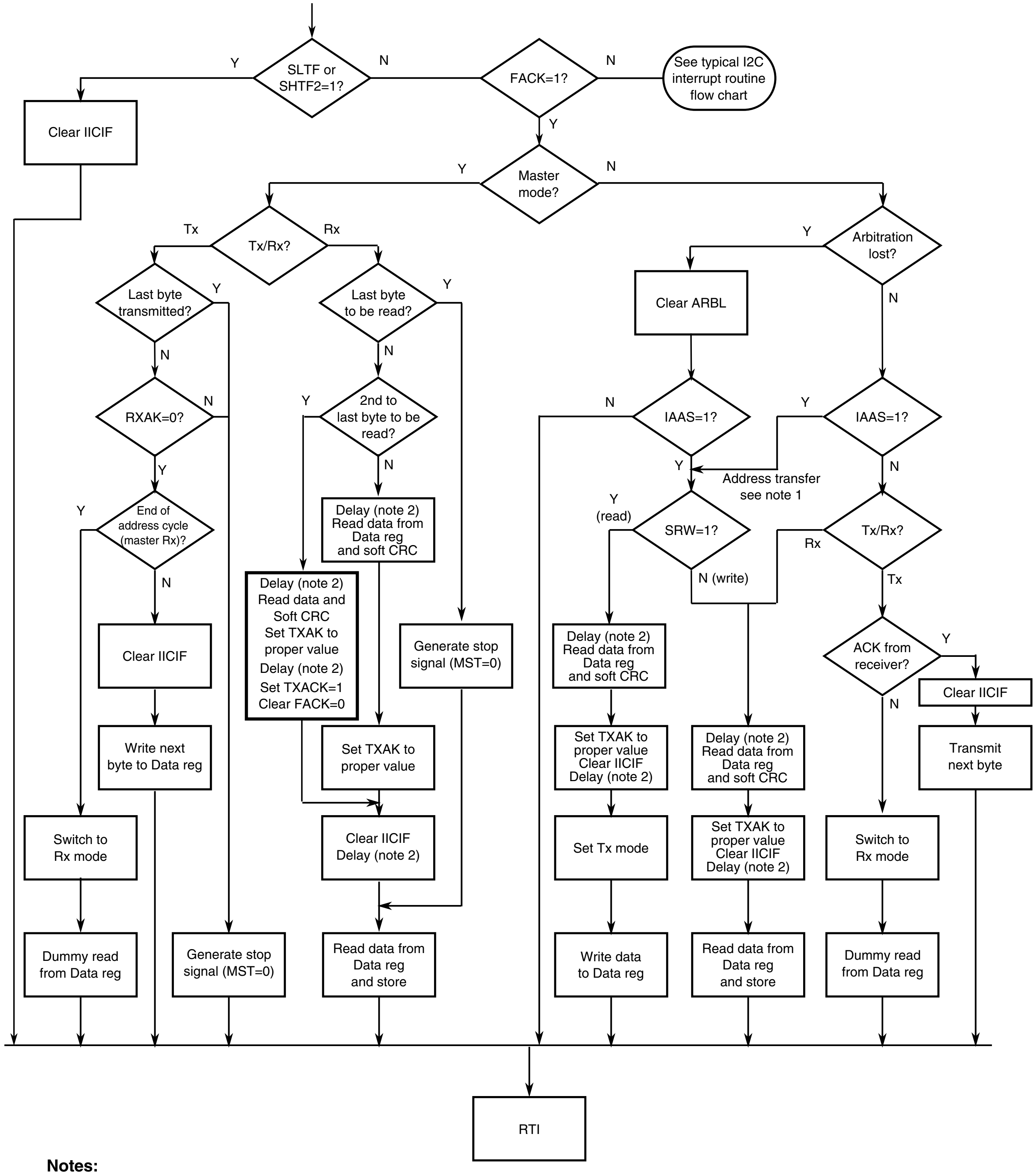
The routine shown in the following figure can handle both master and slave I2C operations. For slave operation, an incoming I2C message that contains the proper address begins I2C communication. For master operation, communication must be initiated by writing the Data register.



Notes:

1. If general call is enabled, check to determine if the received address is a general call address (0x00). If the received address is a general call address, the general call must be handled by user software.
2. When 10-bit addressing addresses a slave, the slave sees an interrupt following the first byte of the extended address. Ensure that for this interrupt, the contents of the Data register are ignored and not treated as a valid data transfer.

Figure 51-42. Typical I2C Interrupt Routine


SLTF or SHTF2=1?
Y
N
Clear IICIF
FACK=1?
N
See typical I2C interrupt routine flow chart
Y
Master mode?
Y
N
Tx/Rx?
Tx
Rx
Arbitration lost?
Y
N
Last byte transmitted?
Y
N
Last byte to be read?
Y
N
Clear ARBL
RXAK=0?
N
Y
2nd to last byte to be read?
Y
N
IAAS=1?
N
Y
IAAS=1?
Y
N
Address transfer see note 1
End of address cycle (master Rx)?
Y
N
Delay (note 2) Read data from Data reg and soft CRC
SRW=1?
Y (read)
N (write)
Tx/Rx?
Rx
Tx
Delay (note 2) Read data and Soft CRC Set TXAK to proper value Delay (note 2) Set TXACK=1 Clear FACK=0
Generate stop signal (MST=0)
Delay (note 2) Read data from Data reg and soft CRC
ACK from receiver?
Y
N
Clear IICIF
Clear IICIF
Write next byte to Data reg
Set TXAK to proper value
Set TXAK to proper value Clear IICIF Delay (note 2)
Delay (note 2) Read data from Data reg and soft CRC
Transmit next byte
Switch to Rx mode
Clear IICIF Delay (note 2)
Set Tx mode
Set TXAK to proper value Clear IICIF Delay (note 2)
Switch to Rx mode
Dummy read from Data reg
Generate stop signal (MST=0)
Read data from Data reg and store
Write data to Data reg
Read data from Data reg and store
Dummy read from Data reg
RTI

Notes:

1. If general call or SIICAEN is enabled, check to determine if the received address is a general call address (0x00) or an SMBus device default address. In either case, they must be handled by user software.
2. In receive mode, one bit time delay may be needed before the first and second data reading.

Figure 51-43. Typical I2C SMBus Interrupt Routine

Chapter 52
Universal Asynchronous Receiver/Transmitter (UART)

52.1 Introduction

NOTE

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

The UART allows asynchronous serial communication with peripheral devices and CPUs.

52.1.1 Features

The UART includes the following features:

- Full-duplex operation
- Standard mark/space non-return-to-zero (NRZ) format
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse width
- 13-bit baud rate selection with /32 fractional divide, based on the module clock frequency
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Programmable transmitter output polarity
- Programmable receive input polarity
- 13-bit break character option

- 11-bit break character detection option
- Independent FIFO structure for transmit and receive
- Two receiver wakeup methods:
    - Idle line wakeup
    - Address mark wakeup
- Address match feature in the receiver to reduce address mark wakeup ISR overhead
- Ability to select MSB or LSB to be first bit on wire
- Hardware flow control support for request to send (RTS) and clear to send (CTS) signals
- Support for ISO 7816 protocol to interface with SIM cards and smart cards
    - Support for T=0 and T=1 protocols
    - Automatic retransmission of NACK'd packets with programmable retry threshold
    - Support for 11 and 12 ETU transfers
    - Detection of initial packet and automated transfer parameter programming
    - Interrupt-driven operation with seven ISO-7816 specific interrupts:
        - Wait time violated
        - Character wait time violated
        - Block wait time violated
        - Initial frame detected
        - Transmit error threshold exceeded
        - Receive error threshold exceeded
        - Guard time violated
- Support for CEA709.1-B protocol used in building automation and home networking systems
    - Automatic clock resynchronization
    - Support for collision detection
- Interrupt-driven operation with 12 flags, not specific to ISO-7816 support

  - Transmitter data buffer at or below watermark
  - Transmission complete
  - Receiver data buffer at or above watermark
  - Idle receiver input
  - Receiver data buffer overrun
  - Receiver data buffer underflow
  - Transmit data buffer overflow
  - Noise error
  - Framing error
  - Parity error
  - Active edge on receive pin
  - LIN break detect
- Receiver framing error detection
- Hardware parity generation and checking
- 1/16 bit-time noise detection
- DMA interface

52.1.2 Modes of operation

The UART functions in the same way in all the normal modes.

It has the following two low power modes:

- Wait mode
- Stop mode

52.1.2.1 Run mode

This is the normal mode of operation.

52.1.2.2 Wait mode

UART operation in the Wait mode depends on the state of the C1[UARTSWAI] field.

- If the C1[UARTSWAI] field is cleared, and the CPU is in Wait mode, the UART operates normally.
- If the C1[UARTSWAI] field is set, and the CPU is in Wait mode, the UART clock generation ceases and the UART module enters a power conservation state.

The C1[UARTSWAI] field does not initiate any power down or power up procedures for the ISO-7816 smartcard interface.

Setting C1[UARTSWAI] does not affect the state of the C2[RE] or C2[TE].

If C1[UARTSWAI] is set, any ongoing transmission or reception stops at the Wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the CPU out of Wait mode. Bringing the CPU out of Wait mode by reset aborts any ongoing transmission or reception and resets the UART.

52.1.2.3 Stop mode

The UART is inactive during Stop mode for reduced power consumption. The STOP instruction does not affect the UART register states, but the UART module clock is disabled. The UART operation resumes after an external interrupt brings the CPU out of Stop mode. Bringing the CPU out of Stop mode by reset aborts any ongoing transmission or reception and resets the UART. Entering or leaving Stop mode does not initiate any power down or power up procedures for the ISO-7816 smartcard interface.

52.2 UART signal descriptions

The UART signals are shown in the following table.

Table 52-1. UART signal descriptions

Signal	Description	I/O
$\overline{\text{CTS}}$	Clear to send	I
$\overline{\text{RTS}}$	Request to send	O
RXD	Receive data	I
TXD	Transmit data	O
$\overline{\text{Collision}}$	Collision detect	I

52.2.1 Detailed signal descriptions

The detailed signal descriptions of the UART are shown in the following table.

Table 52-2. UART—Detailed signal descriptions

Signal	I/O	Description	
$\overline{\text{CTS}}$	I	Clear to send. Indicates whether the UART can start transmitting data when flow control is enabled.	
		State meaning	Asserted—Data transmission can start. Negated—Data transmission cannot start.
		Timing	Assertion—When transmitting device's $\overline{\text{RTS}}$ asserts. Negation—When transmitting device's $\overline{\text{RTS}}$ deasserts.
$\overline{\text{RTS}}$	O	Request to send. When driven by the receiver, indicates whether the UART is ready to receive data. When driven by the transmitter, can enable an external transceiver during transmission.	
		State Meaning	Asserted—When driven by the receiver, ready to receive data. When driven by the transmitter, enable the external transmitter. Negated—When driven by the receiver, not ready to receive data. When driven by the transmitter, disable the external transmitter.
		Timing	Assertion—Can occur at any time; can assert asynchronously to the other input signals. Negation—Can occur at any time; can deassert asynchronously to the other input signals.
RXD	I	Receive data. Serial data input to receiver.	
		State meaning	Whether RXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Sampled at a frequency determined by the module clock divided by the baud rate.
TXD	O	Transmit data. Serial data output from transmitter.	
		State meaning	Whether TXD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Driven at the beginning or within a bit time according to the bit encoding method along with other configuration settings. Otherwise, transmissions are independent of reception timing.
$\overline{\text{Collision}}$	I	Collision Detect. Indicates if a collision is detected during Data Transmission.	
		State Meaning	Asserted—Indicates a collision detection. UARTxCPW determines the length of this pulse for valid collision detection. Negated—No collision detected.
		Timing	Asserts asynchronously to other input signals.

52.3 Memory map and registers

This section provides a detailed description of all memory and registers.

Accessing reserved addresses within the memory map results in a transfer error. None of the contents of the implemented addresses are modified as a result of that access.

Only byte accesses are supported.

UART memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A000	UART Baud Rate Registers: High (UART0_BDH)	8	R/W	00h	52.3.1/ 1622
4006_A001	UART Baud Rate Registers: Low (UART0_BDL)	8	R/W	04h	52.3.2/ 1624
4006_A002	UART Control Register 1 (UART0_C1)	8	R/W	00h	52.3.3/ 1625
4006_A003	UART Control Register 2 (UART0_C2)	8	R/W	00h	52.3.4/ 1627
4006_A004	UART Status Register 1 (UART0_S1)	8	R	C0h	52.3.5/ 1629
4006_A005	UART Status Register 2 (UART0_S2)	8	R/W	00h	52.3.6/ 1632
4006_A006	UART Control Register 3 (UART0_C3)	8	R/W	00h	52.3.7/ 1634
4006_A007	UART Data Register (UART0_D)	8	R/W	00h	52.3.8/ 1635
4006_A008	UART Match Address Registers 1 (UART0_MA1)	8	R/W	00h	52.3.9/ 1637
4006_A009	UART Match Address Registers 2 (UART0_MA2)	8	R/W	00h	52.3.10/ 1637
4006_A00A	UART Control Register 4 (UART0_C4)	8	R/W	00h	52.3.11/ 1638
4006_A00B	UART Control Register 5 (UART0_C5)	8	R/W	00h	52.3.12/ 1639
4006_A00C	UART Extended Data Register (UART0_ED)	8	R	00h	52.3.13/ 1640
4006_A00D	UART Modem Register (UART0_MODEM)	8	R/W	00h	52.3.14/ 1641

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A00E	UART Infrared Register (UART0_IR)	8	R/W	00h	52.3.15/ 1642
4006_A010	UART FIFO Parameters (UART0_PFIFO)	8	R/W	See section	52.3.16/ 1643
4006_A011	UART FIFO Control Register (UART0_CFIFO)	8	R/W	00h	52.3.17/ 1645
4006_A012	UART FIFO Status Register (UART0_SFIFO)	8	R/W	C0h	52.3.18/ 1646
4006_A013	UART FIFO Transmit Watermark (UART0_TWFIFO)	8	R/W	00h	52.3.19/ 1647
4006_A014	UART FIFO Transmit Count (UART0_TCFIFO)	8	R	00h	52.3.20/ 1648
4006_A015	UART FIFO Receive Watermark (UART0_RWFIFO)	8	R/W	01h	52.3.21/ 1648
4006_A016	UART FIFO Receive Count (UART0_RCFIFO)	8	R	00h	52.3.22/ 1649
4006_A018	UART 7816 Control Register (UART0_C7816)	8	R/W	00h	52.3.23/ 1650
4006_A019	UART 7816 Interrupt Enable Register (UART0_IE7816)	8	R/W	00h	52.3.24/ 1652
4006_A01A	UART 7816 Interrupt Status Register (UART0_IS7816)	8	R/W	00h	52.3.25/ 1653
4006_A01B	UART 7816 Wait Parameter Register (UART0_WP7816T0)	8	R/W	0Ah	52.3.26/ 1655
4006_A01B	UART 7816 Wait Parameter Register (UART0_WP7816T1)	8	R/W	0Ah	52.3.27/ 1656
4006_A01C	UART 7816 Wait N Register (UART0_WN7816)	8	R/W	00h	52.3.28/ 1657
4006_A01D	UART 7816 Wait FD Register (UART0_WF7816)	8	R/W	01h	52.3.29/ 1657
4006_A01E	UART 7816 Error Threshold Register (UART0_ET7816)	8	R/W	00h	52.3.30/ 1658
4006_A01F	UART 7816 Transmit Length Register (UART0_TL7816)	8	R/W	00h	52.3.31/ 1659
4006_A021	UART CEA709.1-B Control Register 6 (UART0_C6)	8	R/W	00h	52.3.32/ 1659
4006_A022	UART CEA709.1-B Packet Cycle Time Counter High (UART0_PCTH)	8	R/W	00h	52.3.33/ 1660
4006_A023	UART CEA709.1-B Packet Cycle Time Counter Low (UART0_PCTL)	8	R/W	00h	52.3.34/ 1661

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_A024	UART CEA709.1-B Beta1 Timer (UART0_B1T)	8	R/W	00h	52.3.35/ 1662
4006_A025	UART CEA709.1-B Secondary Delay Timer High (UART0_SDTH)	8	R/W	00h	52.3.36/ 1662
4006_A026	UART CEA709.1-B Secondary Delay Timer Low (UART0_SDTL)	8	R/W	00h	52.3.37/ 1663
4006_A027	UART CEA709.1-B Preamble (UART0_PRE)	8	R/W	00h	52.3.38/ 1663
4006_A028	UART CEA709.1-B Transmit Packet Length (UART0_TPL)	8	R/W	00h	52.3.39/ 1664
4006_A029	UART CEA709.1-B Interrupt Enable Register (UART0_IE)	8	R/W	00h	52.3.40/ 1664
4006_A02A	UART CEA709.1-B WBASE (UART0_WB)	8	R/W	00h	52.3.41/ 1666
4006_A02B	UART CEA709.1-B Status Register (UART0_S3)	8	R/W	00h	52.3.42/ 1666
4006_A02C	UART CEA709.1-B Status Register (UART0_S4)	8	R/W	00h	52.3.43/ 1668
4006_A02D	UART CEA709.1-B Received Packet Length (UART0_RPL)	8	R	00h	52.3.44/ 1669
4006_A02E	UART CEA709.1-B Received Preamble Length (UART0_RPREL)	8	R	00h	52.3.45/ 1670
4006_A02F	UART CEA709.1-B Collision Pulse Width (UART0_CPW)	8	R/W	00h	52.3.46/ 1670
4006_A030	UART CEA709.1-B Receive Indeterminate Time (UART0_RIDT)	8	R/W	00h	52.3.47/ 1671
4006_A031	UART CEA709.1-B Transmit Indeterminate Time (UART0_TIDT)	8	R/W	00h	52.3.48/ 1671
4006_B000	UART Baud Rate Registers: High (UART1_BDH)	8	R/W	00h	52.3.1/ 1622
4006_B001	UART Baud Rate Registers: Low (UART1_BDL)	8	R/W	04h	52.3.2/ 1624
4006_B002	UART Control Register 1 (UART1_C1)	8	R/W	00h	52.3.3/ 1625
4006_B003	UART Control Register 2 (UART1_C2)	8	R/W	00h	52.3.4/ 1627
4006_B004	UART Status Register 1 (UART1_S1)	8	R	C0h	52.3.5/ 1629
4006_B005	UART Status Register 2 (UART1_S2)	8	R/W	00h	52.3.6/ 1632

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_B006	UART Control Register 3 (UART1_C3)	8	R/W	00h	52.3.7/ 1634
4006_B007	UART Data Register (UART1_D)	8	R/W	00h	52.3.8/ 1635
4006_B008	UART Match Address Registers 1 (UART1_MA1)	8	R/W	00h	52.3.9/ 1637
4006_B009	UART Match Address Registers 2 (UART1_MA2)	8	R/W	00h	52.3.10/ 1637
4006_B00A	UART Control Register 4 (UART1_C4)	8	R/W	00h	52.3.11/ 1638
4006_B00B	UART Control Register 5 (UART1_C5)	8	R/W	00h	52.3.12/ 1639
4006_B00C	UART Extended Data Register (UART1_ED)	8	R	00h	52.3.13/ 1640
4006_B00D	UART Modem Register (UART1_MODEM)	8	R/W	00h	52.3.14/ 1641
4006_B00E	UART Infrared Register (UART1_IR)	8	R/W	00h	52.3.15/ 1642
4006_B010	UART FIFO Parameters (UART1_PFIFO)	8	R/W	See section	52.3.16/ 1643
4006_B011	UART FIFO Control Register (UART1_CFIFO)	8	R/W	00h	52.3.17/ 1645
4006_B012	UART FIFO Status Register (UART1_SFIFO)	8	R/W	C0h	52.3.18/ 1646
4006_B013	UART FIFO Transmit Watermark (UART1_TWFIFO)	8	R/W	00h	52.3.19/ 1647
4006_B014	UART FIFO Transmit Count (UART1_TCFIFO)	8	R	00h	52.3.20/ 1648
4006_B015	UART FIFO Receive Watermark (UART1_RWFIFO)	8	R/W	01h	52.3.21/ 1648
4006_B016	UART FIFO Receive Count (UART1_RCFIFO)	8	R	00h	52.3.22/ 1649
4006_B018	UART 7816 Control Register (UART1_C7816)	8	R/W	00h	52.3.23/ 1650
4006_B019	UART 7816 Interrupt Enable Register (UART1_IE7816)	8	R/W	00h	52.3.24/ 1652
4006_B01A	UART 7816 Interrupt Status Register (UART1_IS7816)	8	R/W	00h	52.3.25/ 1653
4006_B01B	UART 7816 Wait Parameter Register (UART1_WP7816T0)	8	R/W	0Ah	52.3.26/ 1655

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_B01B	UART 7816 Wait Parameter Register (UART1_WP7816T1)	8	R/W	0Ah	52.3.27/ 1656
4006_B01C	UART 7816 Wait N Register (UART1_WN7816)	8	R/W	00h	52.3.28/ 1657
4006_B01D	UART 7816 Wait FD Register (UART1_WF7816)	8	R/W	01h	52.3.29/ 1657
4006_B01E	UART 7816 Error Threshold Register (UART1_ET7816)	8	R/W	00h	52.3.30/ 1658
4006_B01F	UART 7816 Transmit Length Register (UART1_TL7816)	8	R/W	00h	52.3.31/ 1659
4006_B021	UART CEA709.1-B Control Register 6 (UART1_C6)	8	R/W	00h	52.3.32/ 1659
4006_B022	UART CEA709.1-B Packet Cycle Time Counter High (UART1_PCTH)	8	R/W	00h	52.3.33/ 1660
4006_B023	UART CEA709.1-B Packet Cycle Time Counter Low (UART1_PCTL)	8	R/W	00h	52.3.34/ 1661
4006_B024	UART CEA709.1-B Beta1 Timer (UART1_B1T)	8	R/W	00h	52.3.35/ 1662
4006_B025	UART CEA709.1-B Secondary Delay Timer High (UART1_SDTH)	8	R/W	00h	52.3.36/ 1662
4006_B026	UART CEA709.1-B Secondary Delay Timer Low (UART1_SDTL)	8	R/W	00h	52.3.37/ 1663
4006_B027	UART CEA709.1-B Preamble (UART1_PRE)	8	R/W	00h	52.3.38/ 1663
4006_B028	UART CEA709.1-B Transmit Packet Length (UART1_TPL)	8	R/W	00h	52.3.39/ 1664
4006_B029	UART CEA709.1-B Interrupt Enable Register (UART1_IE)	8	R/W	00h	52.3.40/ 1664
4006_B02A	UART CEA709.1-B WBASE (UART1_WB)	8	R/W	00h	52.3.41/ 1666
4006_B02B	UART CEA709.1-B Status Register (UART1_S3)	8	R/W	00h	52.3.42/ 1666
4006_B02C	UART CEA709.1-B Status Register (UART1_S4)	8	R/W	00h	52.3.43/ 1668
4006_B02D	UART CEA709.1-B Received Packet Length (UART1_RPL)	8	R	00h	52.3.44/ 1669
4006_B02E	UART CEA709.1-B Received Preamble Length (UART1_RPREL)	8	R	00h	52.3.45/ 1670
4006_B02F	UART CEA709.1-B Collision Pulse Width (UART1_CPW)	8	R/W	00h	52.3.46/ 1670

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_B030	UART CEA709.1-B Receive Indeterminate Time (UART1_RIDT)	8	R/W	00h	52.3.47/ 1671
4006_B031	UART CEA709.1-B Transmit Indeterminate Time (UART1_TIDT)	8	R/W	00h	52.3.48/ 1671
4006_C000	UART Baud Rate Registers: High (UART2_BDH)	8	R/W	00h	52.3.1/ 1622
4006_C001	UART Baud Rate Registers: Low (UART2_BDL)	8	R/W	04h	52.3.2/ 1624
4006_C002	UART Control Register 1 (UART2_C1)	8	R/W	00h	52.3.3/ 1625
4006_C003	UART Control Register 2 (UART2_C2)	8	R/W	00h	52.3.4/ 1627
4006_C004	UART Status Register 1 (UART2_S1)	8	R	C0h	52.3.5/ 1629
4006_C005	UART Status Register 2 (UART2_S2)	8	R/W	00h	52.3.6/ 1632
4006_C006	UART Control Register 3 (UART2_C3)	8	R/W	00h	52.3.7/ 1634
4006_C007	UART Data Register (UART2_D)	8	R/W	00h	52.3.8/ 1635
4006_C008	UART Match Address Registers 1 (UART2_MA1)	8	R/W	00h	52.3.9/ 1637
4006_C009	UART Match Address Registers 2 (UART2_MA2)	8	R/W	00h	52.3.10/ 1637
4006_C00A	UART Control Register 4 (UART2_C4)	8	R/W	00h	52.3.11/ 1638
4006_C00B	UART Control Register 5 (UART2_C5)	8	R/W	00h	52.3.12/ 1639
4006_C00C	UART Extended Data Register (UART2_ED)	8	R	00h	52.3.13/ 1640
4006_C00D	UART Modem Register (UART2_MODEM)	8	R/W	00h	52.3.14/ 1641
4006_C00E	UART Infrared Register (UART2_IR)	8	R/W	00h	52.3.15/ 1642
4006_C010	UART FIFO Parameters (UART2_PFIFO)	8	R/W	See section	52.3.16/ 1643
4006_C011	UART FIFO Control Register (UART2_CFIFO)	8	R/W	00h	52.3.17/ 1645
4006_C012	UART FIFO Status Register (UART2_SFIFO)	8	R/W	C0h	52.3.18/ 1646

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_C013	UART FIFO Transmit Watermark (UART2_TWFIFO)	8	R/W	00h	52.3.19/ 1647
4006_C014	UART FIFO Transmit Count (UART2_TCFIFO)	8	R	00h	52.3.20/ 1648
4006_C015	UART FIFO Receive Watermark (UART2_RWFIFO)	8	R/W	01h	52.3.21/ 1648
4006_C016	UART FIFO Receive Count (UART2_RCFIFO)	8	R	00h	52.3.22/ 1649
4006_C018	UART 7816 Control Register (UART2_C7816)	8	R/W	00h	52.3.23/ 1650
4006_C019	UART 7816 Interrupt Enable Register (UART2_IE7816)	8	R/W	00h	52.3.24/ 1652
4006_C01A	UART 7816 Interrupt Status Register (UART2_IS7816)	8	R/W	00h	52.3.25/ 1653
4006_C01B	UART 7816 Wait Parameter Register (UART2_WP7816T0)	8	R/W	0Ah	52.3.26/ 1655
4006_C01B	UART 7816 Wait Parameter Register (UART2_WP7816T1)	8	R/W	0Ah	52.3.27/ 1656
4006_C01C	UART 7816 Wait N Register (UART2_WN7816)	8	R/W	00h	52.3.28/ 1657
4006_C01D	UART 7816 Wait FD Register (UART2_WF7816)	8	R/W	01h	52.3.29/ 1657
4006_C01E	UART 7816 Error Threshold Register (UART2_ET7816)	8	R/W	00h	52.3.30/ 1658
4006_C01F	UART 7816 Transmit Length Register (UART2_TL7816)	8	R/W	00h	52.3.31/ 1659
4006_C021	UART CEA709.1-B Control Register 6 (UART2_C6)	8	R/W	00h	52.3.32/ 1659
4006_C022	UART CEA709.1-B Packet Cycle Time Counter High (UART2_PCTH)	8	R/W	00h	52.3.33/ 1660
4006_C023	UART CEA709.1-B Packet Cycle Time Counter Low (UART2_PCTL)	8	R/W	00h	52.3.34/ 1661
4006_C024	UART CEA709.1-B Beta1 Timer (UART2_B1T)	8	R/W	00h	52.3.35/ 1662
4006_C025	UART CEA709.1-B Secondary Delay Timer High (UART2_SDTH)	8	R/W	00h	52.3.36/ 1662
4006_C026	UART CEA709.1-B Secondary Delay Timer Low (UART2_SDTL)	8	R/W	00h	52.3.37/ 1663
4006_C027	UART CEA709.1-B Preamble (UART2_PRE)	8	R/W	00h	52.3.38/ 1663

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_C028	UART CEA709.1-B Transmit Packet Length (UART2_TPL)	8	R/W	00h	52.3.39/ 1664
4006_C029	UART CEA709.1-B Interrupt Enable Register (UART2_IE)	8	R/W	00h	52.3.40/ 1664
4006_C02A	UART CEA709.1-B WBASE (UART2_WB)	8	R/W	00h	52.3.41/ 1666
4006_C02B	UART CEA709.1-B Status Register (UART2_S3)	8	R/W	00h	52.3.42/ 1666
4006_C02C	UART CEA709.1-B Status Register (UART2_S4)	8	R/W	00h	52.3.43/ 1668
4006_C02D	UART CEA709.1-B Received Packet Length (UART2_RPL)	8	R	00h	52.3.44/ 1669
4006_C02E	UART CEA709.1-B Received Preamble Length (UART2_RPREL)	8	R	00h	52.3.45/ 1670
4006_C02F	UART CEA709.1-B Collision Pulse Width (UART2_CPW)	8	R/W	00h	52.3.46/ 1670
4006_C030	UART CEA709.1-B Receive Indeterminate Time (UART2_RIDT)	8	R/W	00h	52.3.47/ 1671
4006_C031	UART CEA709.1-B Transmit Indeterminate Time (UART2_TIDT)	8	R/W	00h	52.3.48/ 1671
4006_D000	UART Baud Rate Registers: High (UART3_BDH)	8	R/W	00h	52.3.1/ 1622
4006_D001	UART Baud Rate Registers: Low (UART3_BDL)	8	R/W	04h	52.3.2/ 1624
4006_D002	UART Control Register 1 (UART3_C1)	8	R/W	00h	52.3.3/ 1625
4006_D003	UART Control Register 2 (UART3_C2)	8	R/W	00h	52.3.4/ 1627
4006_D004	UART Status Register 1 (UART3_S1)	8	R	C0h	52.3.5/ 1629
4006_D005	UART Status Register 2 (UART3_S2)	8	R/W	00h	52.3.6/ 1632
4006_D006	UART Control Register 3 (UART3_C3)	8	R/W	00h	52.3.7/ 1634
4006_D007	UART Data Register (UART3_D)	8	R/W	00h	52.3.8/ 1635
4006_D008	UART Match Address Registers 1 (UART3_MA1)	8	R/W	00h	52.3.9/ 1637
4006_D009	UART Match Address Registers 2 (UART3_MA2)	8	R/W	00h	52.3.10/ 1637

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_D00A	UART Control Register 4 (UART3_C4)	8	R/W	00h	52.3.11/ 1638
4006_D00B	UART Control Register 5 (UART3_C5)	8	R/W	00h	52.3.12/ 1639
4006_D00C	UART Extended Data Register (UART3_ED)	8	R	00h	52.3.13/ 1640
4006_D00D	UART Modem Register (UART3_MODEM)	8	R/W	00h	52.3.14/ 1641
4006_D00E	UART Infrared Register (UART3_IR)	8	R/W	00h	52.3.15/ 1642
4006_D010	UART FIFO Parameters (UART3_PFIFO)	8	R/W	See section	52.3.16/ 1643
4006_D011	UART FIFO Control Register (UART3_CFIFO)	8	R/W	00h	52.3.17/ 1645
4006_D012	UART FIFO Status Register (UART3_SFIFO)	8	R/W	C0h	52.3.18/ 1646
4006_D013	UART FIFO Transmit Watermark (UART3_TWFIFO)	8	R/W	00h	52.3.19/ 1647
4006_D014	UART FIFO Transmit Count (UART3_TCFIFO)	8	R	00h	52.3.20/ 1648
4006_D015	UART FIFO Receive Watermark (UART3_RWFIFO)	8	R/W	01h	52.3.21/ 1648
4006_D016	UART FIFO Receive Count (UART3_RCFIFO)	8	R	00h	52.3.22/ 1649
4006_D018	UART 7816 Control Register (UART3_C7816)	8	R/W	00h	52.3.23/ 1650
4006_D019	UART 7816 Interrupt Enable Register (UART3_IE7816)	8	R/W	00h	52.3.24/ 1652
4006_D01A	UART 7816 Interrupt Status Register (UART3_IS7816)	8	R/W	00h	52.3.25/ 1653
4006_D01B	UART 7816 Wait Parameter Register (UART3_WP7816T0)	8	R/W	0Ah	52.3.26/ 1655
4006_D01B	UART 7816 Wait Parameter Register (UART3_WP7816T1)	8	R/W	0Ah	52.3.27/ 1656
4006_D01C	UART 7816 Wait N Register (UART3_WN7816)	8	R/W	00h	52.3.28/ 1657
4006_D01D	UART 7816 Wait FD Register (UART3_WF7816)	8	R/W	01h	52.3.29/ 1657
4006_D01E	UART 7816 Error Threshold Register (UART3_ET7816)	8	R/W	00h	52.3.30/ 1658

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4006_D01F	UART 7816 Transmit Length Register (UART3_TL7816)	8	R/W	00h	52.3.31/ 1659
4006_D021	UART CEA709.1-B Control Register 6 (UART3_C6)	8	R/W	00h	52.3.32/ 1659
4006_D022	UART CEA709.1-B Packet Cycle Time Counter High (UART3_PCTH)	8	R/W	00h	52.3.33/ 1660
4006_D023	UART CEA709.1-B Packet Cycle Time Counter Low (UART3_PCTL)	8	R/W	00h	52.3.34/ 1661
4006_D024	UART CEA709.1-B Beta1 Timer (UART3_B1T)	8	R/W	00h	52.3.35/ 1662
4006_D025	UART CEA709.1-B Secondary Delay Timer High (UART3_SDTH)	8	R/W	00h	52.3.36/ 1662
4006_D026	UART CEA709.1-B Secondary Delay Timer Low (UART3_SDTL)	8	R/W	00h	52.3.37/ 1663
4006_D027	UART CEA709.1-B Preamble (UART3_PRE)	8	R/W	00h	52.3.38/ 1663
4006_D028	UART CEA709.1-B Transmit Packet Length (UART3_TPL)	8	R/W	00h	52.3.39/ 1664
4006_D029	UART CEA709.1-B Interrupt Enable Register (UART3_IE)	8	R/W	00h	52.3.40/ 1664
4006_D02A	UART CEA709.1-B WBASE (UART3_WB)	8	R/W	00h	52.3.41/ 1666
4006_D02B	UART CEA709.1-B Status Register (UART3_S3)	8	R/W	00h	52.3.42/ 1666
4006_D02C	UART CEA709.1-B Status Register (UART3_S4)	8	R/W	00h	52.3.43/ 1668
4006_D02D	UART CEA709.1-B Received Packet Length (UART3_RPL)	8	R	00h	52.3.44/ 1669
4006_D02E	UART CEA709.1-B Received Preamble Length (UART3_RPREL)	8	R	00h	52.3.45/ 1670
4006_D02F	UART CEA709.1-B Collision Pulse Width (UART3_CPW)	8	R/W	00h	52.3.46/ 1670
4006_D030	UART CEA709.1-B Receive Indeterminate Time (UART3_RIDT)	8	R/W	00h	52.3.47/ 1671
4006_D031	UART CEA709.1-B Transmit Indeterminate Time (UART3_TIDT)	8	R/W	00h	52.3.48/ 1671
400E_A000	UART Baud Rate Registers: High (UART4_BDH)	8	R/W	00h	52.3.1/ 1622
400E_A001	UART Baud Rate Registers: Low (UART4_BDL)	8	R/W	04h	52.3.2/ 1624

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400E_A002	UART Control Register 1 (UART4_C1)	8	R/W	00h	52.3.3/ 1625
400E_A003	UART Control Register 2 (UART4_C2)	8	R/W	00h	52.3.4/ 1627
400E_A004	UART Status Register 1 (UART4_S1)	8	R	C0h	52.3.5/ 1629
400E_A005	UART Status Register 2 (UART4_S2)	8	R/W	00h	52.3.6/ 1632
400E_A006	UART Control Register 3 (UART4_C3)	8	R/W	00h	52.3.7/ 1634
400E_A007	UART Data Register (UART4_D)	8	R/W	00h	52.3.8/ 1635
400E_A008	UART Match Address Registers 1 (UART4_MA1)	8	R/W	00h	52.3.9/ 1637
400E_A009	UART Match Address Registers 2 (UART4_MA2)	8	R/W	00h	52.3.10/ 1637
400E_A00A	UART Control Register 4 (UART4_C4)	8	R/W	00h	52.3.11/ 1638
400E_A00B	UART Control Register 5 (UART4_C5)	8	R/W	00h	52.3.12/ 1639
400E_A00C	UART Extended Data Register (UART4_ED)	8	R	00h	52.3.13/ 1640
400E_A00D	UART Modem Register (UART4_MODEM)	8	R/W	00h	52.3.14/ 1641
400E_A00E	UART Infrared Register (UART4_IR)	8	R/W	00h	52.3.15/ 1642
400E_A010	UART FIFO Parameters (UART4_PFIFO)	8	R/W	See section	52.3.16/ 1643
400E_A011	UART FIFO Control Register (UART4_CFIFO)	8	R/W	00h	52.3.17/ 1645
400E_A012	UART FIFO Status Register (UART4_SFIFO)	8	R/W	C0h	52.3.18/ 1646
400E_A013	UART FIFO Transmit Watermark (UART4_TWFIFO)	8	R/W	00h	52.3.19/ 1647
400E_A014	UART FIFO Transmit Count (UART4_TCFIFO)	8	R	00h	52.3.20/ 1648
400E_A015	UART FIFO Receive Watermark (UART4_RWFIFO)	8	R/W	01h	52.3.21/ 1648
400E_A016	UART FIFO Receive Count (UART4_RCFIFO)	8	R	00h	52.3.22/ 1649

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400E_A018	UART 7816 Control Register (UART4_C7816)	8	R/W	00h	52.3.23/ 1650
400E_A019	UART 7816 Interrupt Enable Register (UART4_IE7816)	8	R/W	00h	52.3.24/ 1652
400E_A01A	UART 7816 Interrupt Status Register (UART4_IS7816)	8	R/W	00h	52.3.25/ 1653
400E_A01B	UART 7816 Wait Parameter Register (UART4_WP7816T0)	8	R/W	0Ah	52.3.26/ 1655
400E_A01B	UART 7816 Wait Parameter Register (UART4_WP7816T1)	8	R/W	0Ah	52.3.27/ 1656
400E_A01C	UART 7816 Wait N Register (UART4_WN7816)	8	R/W	00h	52.3.28/ 1657
400E_A01D	UART 7816 Wait FD Register (UART4_WF7816)	8	R/W	01h	52.3.29/ 1657
400E_A01E	UART 7816 Error Threshold Register (UART4_ET7816)	8	R/W	00h	52.3.30/ 1658
400E_A01F	UART 7816 Transmit Length Register (UART4_TL7816)	8	R/W	00h	52.3.31/ 1659
400E_A021	UART CEA709.1-B Control Register 6 (UART4_C6)	8	R/W	00h	52.3.32/ 1659
400E_A022	UART CEA709.1-B Packet Cycle Time Counter High (UART4_PCTH)	8	R/W	00h	52.3.33/ 1660
400E_A023	UART CEA709.1-B Packet Cycle Time Counter Low (UART4_PCTL)	8	R/W	00h	52.3.34/ 1661
400E_A024	UART CEA709.1-B Beta1 Timer (UART4_B1T)	8	R/W	00h	52.3.35/ 1662
400E_A025	UART CEA709.1-B Secondary Delay Timer High (UART4_SDTH)	8	R/W	00h	52.3.36/ 1662
400E_A026	UART CEA709.1-B Secondary Delay Timer Low (UART4_SDTL)	8	R/W	00h	52.3.37/ 1663
400E_A027	UART CEA709.1-B Preamble (UART4_PRE)	8	R/W	00h	52.3.38/ 1663
400E_A028	UART CEA709.1-B Transmit Packet Length (UART4_TPL)	8	R/W	00h	52.3.39/ 1664
400E_A029	UART CEA709.1-B Interrupt Enable Register (UART4_IE)	8	R/W	00h	52.3.40/ 1664
400E_A02A	UART CEA709.1-B WBASE (UART4_WB)	8	R/W	00h	52.3.41/ 1666
400E_A02B	UART CEA709.1-B Status Register (UART4_S3)	8	R/W	00h	52.3.42/ 1666

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400E_A02C	UART CEA709.1-B Status Register (UART4_S4)	8	R/W	00h	52.3.43/ 1668
400E_A02D	UART CEA709.1-B Received Packet Length (UART4_RPL)	8	R	00h	52.3.44/ 1669
400E_A02E	UART CEA709.1-B Received Preamble Length (UART4_RPREL)	8	R	00h	52.3.45/ 1670
400E_A02F	UART CEA709.1-B Collision Pulse Width (UART4_CPW)	8	R/W	00h	52.3.46/ 1670
400E_A030	UART CEA709.1-B Receive Indeterminate Time (UART4_RIDT)	8	R/W	00h	52.3.47/ 1671
400E_A031	UART CEA709.1-B Transmit Indeterminate Time (UART4_TIDT)	8	R/W	00h	52.3.48/ 1671
400E_B000	UART Baud Rate Registers: High (UART5_BDH)	8	R/W	00h	52.3.1/ 1622
400E_B001	UART Baud Rate Registers: Low (UART5_BDL)	8	R/W	04h	52.3.2/ 1624
400E_B002	UART Control Register 1 (UART5_C1)	8	R/W	00h	52.3.3/ 1625
400E_B003	UART Control Register 2 (UART5_C2)	8	R/W	00h	52.3.4/ 1627
400E_B004	UART Status Register 1 (UART5_S1)	8	R	C0h	52.3.5/ 1629
400E_B005	UART Status Register 2 (UART5_S2)	8	R/W	00h	52.3.6/ 1632
400E_B006	UART Control Register 3 (UART5_C3)	8	R/W	00h	52.3.7/ 1634
400E_B007	UART Data Register (UART5_D)	8	R/W	00h	52.3.8/ 1635
400E_B008	UART Match Address Registers 1 (UART5_MA1)	8	R/W	00h	52.3.9/ 1637
400E_B009	UART Match Address Registers 2 (UART5_MA2)	8	R/W	00h	52.3.10/ 1637
400E_B00A	UART Control Register 4 (UART5_C4)	8	R/W	00h	52.3.11/ 1638
400E_B00B	UART Control Register 5 (UART5_C5)	8	R/W	00h	52.3.12/ 1639
400E_B00C	UART Extended Data Register (UART5_ED)	8	R	00h	52.3.13/ 1640
400E_B00D	UART Modem Register (UART5_MODEM)	8	R/W	00h	52.3.14/ 1641

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400E_B00E	UART Infrared Register (UART5_IR)	8	R/W	00h	52.3.15/ 1642
400E_B010	UART FIFO Parameters (UART5_PFIFO)	8	R/W	See section	52.3.16/ 1643
400E_B011	UART FIFO Control Register (UART5_CFIFO)	8	R/W	00h	52.3.17/ 1645
400E_B012	UART FIFO Status Register (UART5_SFIFO)	8	R/W	C0h	52.3.18/ 1646
400E_B013	UART FIFO Transmit Watermark (UART5_TWFIFO)	8	R/W	00h	52.3.19/ 1647
400E_B014	UART FIFO Transmit Count (UART5_TCFIFO)	8	R	00h	52.3.20/ 1648
400E_B015	UART FIFO Receive Watermark (UART5_RWFIFO)	8	R/W	01h	52.3.21/ 1648
400E_B016	UART FIFO Receive Count (UART5_RCFIFO)	8	R	00h	52.3.22/ 1649
400E_B018	UART 7816 Control Register (UART5_C7816)	8	R/W	00h	52.3.23/ 1650
400E_B019	UART 7816 Interrupt Enable Register (UART5_IE7816)	8	R/W	00h	52.3.24/ 1652
400E_B01A	UART 7816 Interrupt Status Register (UART5_IS7816)	8	R/W	00h	52.3.25/ 1653
400E_B01B	UART 7816 Wait Parameter Register (UART5_WP7816T0)	8	R/W	0Ah	52.3.26/ 1655
400E_B01B	UART 7816 Wait Parameter Register (UART5_WP7816T1)	8	R/W	0Ah	52.3.27/ 1656
400E_B01C	UART 7816 Wait N Register (UART5_WN7816)	8	R/W	00h	52.3.28/ 1657
400E_B01D	UART 7816 Wait FD Register (UART5_WF7816)	8	R/W	01h	52.3.29/ 1657
400E_B01E	UART 7816 Error Threshold Register (UART5_ET7816)	8	R/W	00h	52.3.30/ 1658
400E_B01F	UART 7816 Transmit Length Register (UART5_TL7816)	8	R/W	00h	52.3.31/ 1659
400E_B021	UART CEA709.1-B Control Register 6 (UART5_C6)	8	R/W	00h	52.3.32/ 1659
400E_B022	UART CEA709.1-B Packet Cycle Time Counter High (UART5_PCTH)	8	R/W	00h	52.3.33/ 1660
400E_B023	UART CEA709.1-B Packet Cycle Time Counter Low (UART5_PCTL)	8	R/W	00h	52.3.34/ 1661

Table continues on the next page...

UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400E_B024	UART CEA709.1-B Beta1 Timer (UART5_B1T)	8	R/W	00h	52.3.35/ 1662
400E_B025	UART CEA709.1-B Secondary Delay Timer High (UART5_SDTH)	8	R/W	00h	52.3.36/ 1662
400E_B026	UART CEA709.1-B Secondary Delay Timer Low (UART5_SDTL)	8	R/W	00h	52.3.37/ 1663
400E_B027	UART CEA709.1-B Preamble (UART5_PRE)	8	R/W	00h	52.3.38/ 1663
400E_B028	UART CEA709.1-B Transmit Packet Length (UART5_TPL)	8	R/W	00h	52.3.39/ 1664
400E_B029	UART CEA709.1-B Interrupt Enable Register (UART5_IE)	8	R/W	00h	52.3.40/ 1664
400E_B02A	UART CEA709.1-B WBASE (UART5_WB)	8	R/W	00h	52.3.41/ 1666
400E_B02B	UART CEA709.1-B Status Register (UART5_S3)	8	R/W	00h	52.3.42/ 1666
400E_B02C	UART CEA709.1-B Status Register (UART5_S4)	8	R/W	00h	52.3.43/ 1668
400E_B02D	UART CEA709.1-B Received Packet Length (UART5_RPL)	8	R	00h	52.3.44/ 1669
400E_B02E	UART CEA709.1-B Received Preamble Length (UART5_RPREL)	8	R	00h	52.3.45/ 1670
400E_B02F	UART CEA709.1-B Collision Pulse Width (UART5_CPW)	8	R/W	00h	52.3.46/ 1670
400E_B030	UART CEA709.1-B Receive Indeterminate Time (UART5_RIDT)	8	R/W	00h	52.3.47/ 1671
400E_B031	UART CEA709.1-B Transmit Indeterminate Time (UART5_TIDT)	8	R/W	00h	52.3.48/ 1671

52.3.1 UART Baud Rate Registers: High (UARTx_BDH)

This register, along with the BDL register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting (SBR[12:0]), first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written.

BDL is reset to a non-zero value, but after reset, the baud rate generator remains disabled until the first time the receiver or transmitter is enabled, that is, when C2[RE] or C2[TE] are set.

Addresses: UART0_BDH is 4006_A000h base + 0h offset = 4006_A000h
UART1_BDH is 4006_B000h base + 0h offset = 4006_B000h
UART2_BDH is 4006_C000h base + 0h offset = 4006_C000h
UART3_BDH is 4006_D000h base + 0h offset = 4006_D000h
UART4_BDH is 400E_A000h base + 0h offset = 400E_A000h
UART5_BDH is 400E_B000h base + 0h offset = 400E_B000h

Bit	7	6	5	4	3	2	1	0
Read	LBKDIE	RXEDGIE	0	SBR				
Write			▨					
Reset	0	0	0	0	0	0	0	0

UARTx_BDH field descriptions

Field	Description
7 LBKDIE	LIN Break Detect Interrupt Enable LBKDIE enables the LIN break detect flag, LBKDIF, to generate interrupt requests based on the state of LBKDDMAS. 0 LBKDIF interrupt requests disabled. 1 LBKDIF interrupt requests enabled.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable RXEDGIE enables the Receive input active edge, RXEDGIF, to generate interrupt requests. 0 Hardware interrupts from RXEDGIF disabled (use polling). 1 RXEDGIF interrupt request enabled.
5 Reserved	This read-only field is reserved and always has the value zero.
4–0 SBR	UART Baud Rate Bits The baud rate for the UART is determined by these 13 bits. See Baud rate generation for details. **NOTE:** The baud rate generator is disabled until the C2[TE] bit or the C2[RE] bit is set for the first time after reset.The baud rate generator is disabled when SBR = 0. **NOTE:** Writing to BDH has no effect without writing to BDL, since writing to BDH puts the data in a temporary location until BDL is written.

52.3.2 UART Baud Rate Registers: Low (UARTx_BDL)

This register, along with the BDH register, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting (SBR[12:0]), first write to BDH to buffer the high half of the new value and then write to BDL. The working value in BDH does not change until BDL is written. BDL is reset to a non-zero value, but after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled (C2[RE] or C2[TE] bits are set)

Addresses: UART0_BDL is 4006_A000h base + 1h offset = 4006_A001h
UART1_BDL is 4006_B000h base + 1h offset = 4006_B001h
UART2_BDL is 4006_C000h base + 1h offset = 4006_C001h
UART3_BDL is 4006_D000h base + 1h offset = 4006_D001h
UART4_BDL is 400E_A000h base + 1h offset = 400E_A001h
UART5_BDL is 400E_B000h base + 1h offset = 400E_B001h

Bit	7	6	5	4	3	2	1	0
Read Write	SBR							
Reset	0	0	0	0	0	1	0	0

UARTx_BDL field descriptions

Field	Description
7–0 SBR	UART Baud Rate Bits The baud rate for the UART is determined by these 13 bits. See Baud rate generation for details **NOTE:** The baud rate generator is disabled until the C2[TE] bit or the C2[RE] bit is set for the first time after reset.The baud rate generator is disabled when SBR = 0. **NOTE:** Writing to BDH has no effect without writing to BDL, since writing to BDH puts the data in a temporary location until BDL is written. **NOTE:** When the 1/32 narrow pulse width is selected for infrared (IrDA), the baud rate bits must be even, the least significant bit is 0. Refer to MODEM register.

52.3.3 UART Control Register 1 (UARTx_C1)

This read/write register controls various optional features of the UART system.

Addresses: UART0_C1 is 4006_A000h base + 2h offset = 4006_A002h
UART1_C1 is 4006_B000h base + 2h offset = 4006_B002h
UART2_C1 is 4006_C000h base + 2h offset = 4006_C002h
UART3_C1 is 4006_D000h base + 2h offset = 4006_D002h
UART4_C1 is 400E_A000h base + 2h offset = 400E_A002h
UART5_C1 is 400E_B000h base + 2h offset = 400E_B002h

Bit	7	6	5	4	3	2	1	0
Read Write	LOOPS	UARTSWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

UARTx_C1 field descriptions

Field	Description
7 LOOPS	Loop Mode Select When LOOPS is set, the RxD pin is disconnected from the UART and the transmitter output is internally connected to the receiver input.The transmitter and the receiver must be enabled to use the loop function. 0 Normal operation. 1 Loop mode where transmitter output is internally connected to receiver input. The receiver input is determined by the RSRC bit.
6 UARTSWAI	UART Stops in Wait Mode 0 UART clock continues to run in wait mode. 1 UART clock freezes while CPU is in wait mode.
5 RSRC	Receiver Source Select This bit has no meaning or effect unless the LOOPS bit is set. When LOOPS is set, the RSRC bit determines the source for the receiver shift register input. 0 Selects internal loop back mode and receiver input is internally connected to transmitter output. 1 Single-wire UART mode where the receiver input is connected to the transmit pin input signal.
4 M	9-bit or 8-bit Mode Select This bit must be set when 7816E is set/enabled. 0 Normal - start + 8 data bits (MSB/LSB first as determined by MSBF) + stop. 1 Use - start + 9 data bits (MSB/LSB first as determined by MSBF) + stop.
3 WAKE	Receiver Wakeup Method Select WAKE determines which condition wakes the UART: address mark in the most significant bit position of a received data character or an idle condition on the receive pin input signal.

Table continues on the next page...

UARTx_C1 field descriptions (continued)

Field	Description
	0 Idle-line wakeup. 1 Address-mark wakeup.
2 ILT	Idle Line Type Select ILT determines when the receiver starts counting logic 1s as idle character bits. The counting 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 the case where UART is programmed with ILT = 1, a logic of 1'b0 is automatically shifted after a received stop bit thus resetting the idle count. **NOTE:** In the case where UART is programmed for IDLE line wakeup (RWU = 1 and WAKE = 0), ILT has no effect on when the receiver starts counting logic 1s as idle character bits. In idle line wakeup an idle character is recognized at anytime the receiver sees 10, 11, or 12 1s depending on the M, PE, and C4[M10] bits. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables the parity function. When parity is enabled, parity function inserts a parity bit in the bit position immediately preceding the stop bit. This bit must be set when 7816E is set/enabled. 0 Parity function disabled. 1 Parity function enabled.
0 PT	Parity Type PT determines whether the UART generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit.This bit must be cleared when 7816E is set/enabled. 0 Even parity. 1 Odd parity.

52.3.4 UART Control Register 2 (UARTx_C2)

This register can be read or written at any time.

Addresses: UART0_C2 is 4006_A000h base + 3h offset = 4006_A003h
UART1_C2 is 4006_B000h base + 3h offset = 4006_B003h
UART2_C2 is 4006_C000h base + 3h offset = 4006_C003h
UART3_C2 is 4006_D000h base + 3h offset = 4006_D003h
UART4_C2 is 400E_A000h base + 3h offset = 400E_A003h
UART5_C2 is 400E_B000h base + 3h offset = 400E_B003h

Bit	7	6	5	4	3	2	1	0
Read	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_C2 field descriptions

Field	Description
7 TIE	Transmitter Interrupt or DMA Transfer Enable. TIE enables the S1[TDRE] flag, to generate interrupt requests or DMA transfer requests, based on the state of C5[TDMAS]. **NOTE:** If C2[TIE] and C5[TDMAS] are both set, then TCIE must be cleared, and D[D] must not be written outside of servicing of a DMA request. 0 TDRE interrupt and DMA transfer requests disabled. 1 TDRE interrupt or DMA transfer requests enabled.
6 TCIE	Transmission Complete Interrupt Enable TCIE enables the transmission complete flag, S1[TC], to generate interrupt requests. 0 TC interrupt requests disabled. 1 TC interrupt requests enabled.
5 RIE	Receiver Full Interrupt or DMA Transfer Enable RIE enables the S1[RDRF] flag, to generate interrupt requests or DMA transfer requests, based on the state of C5[RDMAS]. 0 RDRF interrupt and DMA transfer requests disabled. 1 RDRF interrupt or DMA transfer requests enabled
4 ILIE	Idle Line Interrupt Enable ILIE enables the idle line flag, S1[IDLE], to generate interrupt requests, based on the state of C5[ILDMAS]. 0 IDLE interrupt requests disabled. 1 IDLE interrupt requests enabled.

Table continues on the next page...

UARTx_C2 field descriptions (continued)

Field	Description
3 TE	Transmitter Enable TE enables the UART transmitter.The TE bit can be used to queue an idle preamble by clearing and then setting the TE bit. When 7816E is set/enabled and C7816[TTYPE] = 1, this bit is automatically cleared after the requested block has been transmitted. This condition is detected when TL7816[TLEN] = 0 and four additional characters have been transmitted. 0 Transmitter off. 1 Transmitter on.
2 RE	Receiver Enable RE enables the UART receiver. 0 Receiver off. 1 Receiver on.
1 RWU	Receiver Wakeup Control This bit can be set to place the UART receiver in a standby state. RWU automatically clears when an RWU event occurs (an IDLE event when C1[WAKE] is clear or an address match when C1[WAKE] is set). This bit must be cleared when 7816E is set. **NOTE:** RWU should only be set with C1[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by the S2[RAF] flag. If set to wake up an IDLE event and the channel is already idle, it is possible that the UART will discard data since data must be received (or a LIN break detect) after an IDLE is detected before IDLE is allowed to reasserted. 0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	Send Break Toggling SBK sends one break character (10, 11, or 12 logic 0s, if S2[BRK13] is cleared; 13 or 14 logic 0s, if S2[BRK13] is set). See Transmitting break characters for the number of logic 0s for the different configurations. Toggling implies clearing the SBK bit before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10, 11, or 12 bits, or 13 or 14 bits).This bit must be cleared when 7816E is set. 0 Normal transmitter operation. 1 Queue break character(s) to be sent.

52.3.5 UART Status Register 1 (UARTx_S1)

The S1 register provides inputs to the MCU for generation of UART interrupts or DMA requests. This register can also be polled by the MCU to check the status of these bits. To clear a flag, the status register should be read followed by a read or write (depending on interrupt flag type) to the UART Data Register. Other instructions can be executed between the two steps as long as it does not compromise the handling of I/O, but the order of operations is important for flag clearing. When a flag is configured to trigger a DMA request, assertion of the associated DMA done signal from the DMA controller, clears the flag.

NOTE

> If the condition that results in the assertion of the flag, interrupt or DMA request is not resolved prior to clearing the flag, the flag (and interrupt/DMA request) will reassert. For example, if the DMA or interrupt service routine failed to write sufficient data to the transmit buffer to raise it above the watermark level, the flag will reassert and generate another interrupt or DMA request.

NOTE

> Reading an empty data register to clear one of these flags causes the FIFO pointers to get out of alignment. A receive FIFO flush reinitializes the pointers.

Addresses: UART0_S1 is 4006_A000h base + 4h offset = 4006_A004h
UART1_S1 is 4006_B000h base + 4h offset = 4006_B004h
UART2_S1 is 4006_C000h base + 4h offset = 4006_C004h
UART3_S1 is 4006_D000h base + 4h offset = 4006_D004h
UART4_S1 is 400E_A000h base + 4h offset = 400E_A004h
UART5_S1 is 400E_B000h base + 4h offset = 400E_B004h

Bit	7	6	5	4	3	2	1	0
Read	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write	▨	▨	▨	▨	▨	▨	▨	▨
Reset	1	1	0	0	0	0	0	0

UARTx_S1 field descriptions

Field	Description
7 TDRE	Transmit Data Register Empty Flag TDRE will set when the number of datawords in the transmit buffer (D and C3[T8])is equal to or less than the number indicated by TWFIFO[TXWATER]. A character that is in the process of being transmitted is not included in the count. To clear TDRE, read S1 when TDRE is set and then write to the UART data

Table continues on the next page...

UARTx_S1 field descriptions (continued)

Field	Description
	register (D). For more efficient interrupt servicing all data except the final value to be written to the buffer should written to D/C3[T8]. Then S1 can be read before writing the final data value, resulting in the clearing of the TRDE flag. This is more efficient since the TDRE will reassert until the watermark has been exceeded so attempting to clear the TDRE every write will be ineffective until sufficient data has been written. 0 The amount of data in the transmit buffer is greater than the value indicated by TWFIFO[TXWATER]. 1 The amount of data in the transmit buffer is less than or equal to the value indicated by TWFIFO[TXWATER] at some point in time since the flag has been cleared.
6 TC	Transmit 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). When 7816E is set/enabled this bit is set after any NACK signal has been received but prior to any corresponding guard times expiring.When EN709 is set/enabled this flag is not set on transmit packet completion. TC is cleared by reading S1 with TC set and then doing one of the following: • Writing to the UART data register (D) to transmit new data • Queuing a preamble by clearing and then setting the C2[TE] bit. • Queuing a break character by writing 1 to SBK in C2 0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).
5 RDRF	Receive Data Register Full Flag RDRF is set when the number of datawords in the receive buffer is equal to or more than the number indicated by RWFIFO[RXWATER]. A dataword that is in the process of being received is not included in the count. RDRF is prevented from setting while S2[LBKDE] is set. Additionally, when S2[LBKDE] is set, datawords that are received will be stored in the receive buffer but will over-write each other. To clear RDRF, read S1 when RDRF is set and then read the UART data register (D). For more efficient interrupt and DMA operation all data except the final value is to be read from the buffer using D/C3[T8]/ED. The S1 should then be read and the final data value read, resulting in the clearing of the RDRF flag. Even if the RDRF flag is set, data will continue to be received until an overrun condition occurs. 0 The number of datawords in the receive buffer is less than the number indicated by RXWATER. 1 The number of datawords in the receive buffer is equal to or greater than the number indicated by RXWATER at some point in time since this flag was last cleared.
4 IDLE	Idle Line Flag IDLE is set when 10 consecutive logic 1s (if C1[M] = 0), 11 consecutive logic 1s (if C1[M] = 1 and C4[M10] = 0), or 12 consecutive logic 1s (if C1[M] = 1, C4[M10] = 1, and C1[PE] = 1) appear on the receiver input. After the IDLE flag is cleared, a frame must be received (although not necessarily stored in the data buffer, for example if C2[RWU] is set) or a LIN break character must set the S2[LBKDIF] flag before an idle condition can set the IDLE flag. To clear IDLE, read UART status S1 with IDLE set and then read D. Idle detection is not supported when 7816E or EN709 is set/enabled and hence this flag is ignored. **NOTE:** When the receiver wakeup bit (RWU) is set and WAKE is cleared, an idle line condition sets the IDLE flag if RWUID is set, else the IDLE flag does not get set. 0 Receiver input is either active now or has never become active since the IDLE flag was last cleared. 1 Receiver input has become idle or the flag has not been cleared since it last asserted.
3 OR	Receiver Overrun Flag

Table continues on the next page...

UARTx_S1 field descriptions (continued)

Field	Description
	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 UART data registers is not affected. If the OR flag is set, no data will be stored in the data buffer even if sufficient room exists. Additionally, while the OR flag is set the RDRF flag, and IDLE flags will be blocked from asserting, i.e. transition from an inactive to an active state. To clear OR, read S1 when OR is set and then read UART data register (D). If LBKDE is enabled and a LIN Break is detected, the OR bit will assert if the S2[LBKDIF] flag is not cleared before the next data character is received.See Overrun (OR) flag implications for more details regarding the operation of the OR bit. In 7816 mode, it is possible to configure a NACK to be returned by programing the C7816[ONACK] bit. 0 No overrun has occurred since the last time the flag was cleared. 1 Overrun has occurred or the overrun flag has not been cleared since the last overrun occured.
2 NF	Noise Flag NF is set when the UART detects noise on the receiver input. NF bit does not get set in the case of an overrun or while the LIN break detect feature is enabled (S2[LBKDE] = 1). When NF is set, it only indicates that a dataword has been received with noise since the last time it was cleared. There is no guarantee that the first dataword read from the receive buffer has noise or that there is only one dataword in the buffer that was received with noise unless the receive buffer has a depth of one. To clear NF, read S1 and then read the UART data register (D). When EN709 is set/enabled, noise flag is not set. 0 No noise detected since the last time this flag was cleared. If the receive buffer has a depth greater than 1 then there may be data in the receiver buffer that was received with noise. 1 At least one dataword was received with noise detected since the last time the flag was cleared.
1 FE	Framing Error Flag FE is set when a logic 0 is accepted as the stop bit. FE bit does not set in the case of an overrun or while the LIN break detect feature is enabled (S2[LBKDE] = 1). FE inhibits further data reception until it is cleared. To clear FE, read S1 with FE set and then read the UART data register (D). The last data in the receive buffer represents the data that was received with the frame error enabled. However, framing errors are not supported when 7816E is set/enabled. However, if this flag is set, data will still not be received in 7816 mode. Framing errors are not supported in 709 mode. 0 No framing error detected. 1 Framing error.
0 PF	Parity Error Flag PF is set when PE is set, S2[LBKDE] is disabled, and the parity of the received data does not match its parity bit. The PF is not set in the case of an overrun condition. When the PF bit is set it only indicates that a dataword was received with parity error since the last time it was cleared. There is no guarantee that the first dataword read from the receive buffer has a parity error or that there is only one dataword in the buffer that was received with a parity error unless the receive buffer was a depth of one. To clear PF, read S1 and then read the UART data register (D). Within the receive buffer structure the received dataword is tagged if it was received with a parity error. That information is available by reading the ED register prior to reading the D register. When EN709 is set/enabled parity error flag is not set. 0 No parity error has been detected since the last time this flag was cleared. If the receive buffer has a depth greater than 1 then there may be data in the receive buffer what was received with a parity error. 1 At least one dataword was received with a parity error since the last time this flag was cleared.

52.3.6 UART Status Register 2 (UARTx_S2)

The S2 register provides inputs to the MCU for generation of UART interrupts or DMA requests. Also, this register can be polled by the MCU to check the status of these bits. This register can be read or written at any time, with the exception of the MSBF and RXINV bits which should only be changed by the user between transmit and receive packets.

Addresses: UART0_S2 is 4006_A000h base + 5h offset = 4006_A005h
UART1_S2 is 4006_B000h base + 5h offset = 4006_B005h
UART2_S2 is 4006_C000h base + 5h offset = 4006_C005h
UART3_S2 is 4006_D000h base + 5h offset = 4006_D005h
UART4_S2 is 400E_A000h base + 5h offset = 400E_A005h
UART5_S2 is 400E_B000h base + 5h offset = 400E_B005h

Bit	7	6	5	4	3	2	1	0
Read	LBKDIF	RXEDGIF	MSBF	RXINV	RWUID	BRK13	LBKDE	RAF
Write								▨
Reset	0	0	0	0	0	0	0	0

UARTx_S2 field descriptions

Field	Description
7 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when LBKDE is set and a LIN break character is detected, when 11 consecutive logic 0s (if C1[M] = 0) or 12 consecutive logic 0s (if C1[M] = 1) appear on the receiver input. LBKDIF is set right after receiving the last LIN break character bit. LBKDIF is cleared by writing a 1 to it. 0 No LIN break character has been detected. 1 LIN break character has been detected.
6 RXEDGIF	RxD Pin Active Edge Interrupt Flag RXEDGIF is set when 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. See RXEDGIF description for additional details. **NOTE:** The active edge is only detected when in two wire mode and on receive data coming from the RxD pin. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
5 MSBF	Most Significant Bit 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 is automatically set when C7816[INIT] and C7816[ISO7816E] are enabled and an initial character is detected in T = 0 protocol mode. In EN709 mode, this bit affects the order of bits the same way as it does in normal mode.

Table continues on the next page...

UARTx_S2 field descriptions (continued)

Field	Description
	0 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. 1 MSB (bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of C1[M] and C1[PE]. Further, the first bit received after the start bit is identified as bit8, bit7 or bit6 depending on the setting of C1[M] and C1[PE].
4 RXINV	Receive Data Inversion Setting this bit, reverses the polarity of the received data input. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity.This bit is automatically set when C7816[INIT] and C7816[ISO7816E] are enabled and an initial character is detected in T = 0 protocol mode. In EN709 mode, this bit affects the polarity of bits the same as it does in normal mode. **NOTE:** Setting RXINV inverts the RxD input for: data bits, start and stop bits, break, and idle. When C7816[ISO7816E] is set/enabled then only the data bits and the parity bit are inverted. 0 Receive data is not inverted. 1 Receive data is inverted.
3 RWUID	Receive Wakeup Idle Detect When RWU is set and WAKE is cleared, this bit controls whether the idle character that wakes the receiver sets the S1[IDLE] bit.This bit must be cleared when C7816[ISO7816E] is set/enabled. 0 The S1[IDLE] bit is not set upon detection of an idle character. 1 The S1[IDLE] bit is set upon detection of an idle character.
2 BRK13	Break Transmit Character Length This bit determines whether the transmit break character is 10, 11, or 12 bits long, or 13 or 14 bits long. Refer to Transmitting break characters for the length of the break character for the different configurations. The detection of a framing error is not affected by this bit. 0 Break character is 10, 11, or 12 bits long. 1 Break character is 13 or 14 bits long.
1 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, the S1[RDRF], S1[NF], S1[FE], and S1[PF] flags are prevented from setting. When LBKDE is set, see Overrun operation. The LBKDE bit must be cleared when C7816[ISO7816E] is set. 0 Break character is detected at length of 10 bit times (C1[M] = 0), 11 (C1[M] = 1 and C4[M10] = 0), or 12 (C1[M] = 1, C4[M10] = 1, and S1[PE] = 1). 1 Break character is detected at length of 11 bits times (if C1[M] = 0 or 12 bits time (if C1[M] = 1).
0 RAF	Receiver Active Flag RAF is set when the UART receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character when C7816[ISO7816E] is cleared/disabled. When C7816[ISO7816E] is enabled the RAF is cleared if the C7816[TTYPE] = 0 expires or the C7816[TTYPE] = 1 expires. **NOTE:** In the case when C7816[ISO7816E] is set and C7816[TTYPE] = 0, it is possible to configure the guard time to be 12. However, in the event that a NACK is required to be transmitted the data

Table continues on the next page...

UARTx_S2 field descriptions (continued)

Field	Description
	transfer actually takes 13 ETU with the 13th ETU slot being a inactive buffer. Hence in this situation the RAF may deassert one ETU prior to actually being inactive. 0 UART receiver idle/inactive waiting for a start bit. 1 UART receiver active (RxD input not idle).

52.3.7 UART Control Register 3 (UARTx_C3)

Writing to R8 bit does not have any effect. The TXDIR and TXINV bits can only be changed between transmit and receive packets.

Addresses: UART0_C3 is 4006_A000h base + 6h offset = 4006_A006h
UART1_C3 is 4006_B000h base + 6h offset = 4006_B006h
UART2_C3 is 4006_C000h base + 6h offset = 4006_C006h
UART3_C3 is 4006_D000h base + 6h offset = 4006_D006h
UART4_C3 is 400E_A000h base + 6h offset = 400E_A006h
UART5_C3 is 400E_B000h base + 6h offset = 400E_B006h

Bit	7	6	5	4	3	2	1	0
Read	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_C3 field descriptions

Field	Description
7 R8	Received Bit 8 R8 is the ninth data bit received when the UART is configured for 9-bit data format (C1[M] = 1) or (C4[M10] = 1).
6 T8	Transmit Bit 8 T8 is the ninth data bit transmitted when the UART is configured for 9-bit data format (C1[M] = 1) or (C4[M10] = 1). **NOTE:** If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten. The same value is transmitted until T8 is rewritten.
5 TXDIR	Transmitter Pin Data Direction in Single-Wire mode This bit determines whether the TXD pin is used as an input or output in the single-wire mode of operation. This bit is relevant only to the single-wire mode. When C7816[ISO7816E] is set/enabled and C7816[TTYPE] = 1, this bit is automatically cleared after the requested block has been transmitted. This condition is detected when TL7816[TLEN] = 0 and 4 additional characters have been transmitted. Additionally, if C7816[ISO7816E] is set/enabled and C7816[TTYPE] = 0 and a NACK is being transmitted, the hardware will automatically override this bit as needed. In this situation TXDIR will not reflect the temporary state associated with the NACK.

Table continues on the next page...

UARTx_C3 field descriptions (continued)

Field	Description
	0 TXD pin is an input in single-wire mode. 1 TXD pin is an output in single-wire mode.
4 TXINV	Transmit Data Inversion. Setting this bit reverses the polarity of the transmitted data output. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity. This bit is automatically set when C7816[INIT] and C7816[ISO7816E] are enabled and an initial character is detected in T = 0 protocol mode. **NOTE:** Setting TXINV inverts all transmitted values, including idle, break, start, and stop bits. In loop mode, if TXINV is set, the receiver gets the transmit inversion bit when RXINV is disabled.When C7816[ISO7816E] is set/enabled then only the transmitted data bits and parity bit are inverted. 0 Transmit data is not inverted. 1 Transmit data is inverted.
3 ORIE	Overrun Error Interrupt Enable This bit enables the overrun error flag (S1[OR]) to generate interrupt requests. 0 OR interrupts are disabled. 1 OR interrupt requests are enabled.
2 NEIE	Noise Error Interrupt Enable This bit enables the noise flag (S1[NF]) to generate interrupt requests. 0 NF interrupt requests are disabled. 1 NF interrupt requests are enabled.
1 FEIE	Framing Error Interrupt Enable This bit enables the framing error flag (S1[FE]) to generate interrupt requests. 0 FE interrupt requests are disabled. 1 FE interrupt requests are enabled.
0 PEIE	Parity Error Interrupt Enable This bit enables the parity error flag (S1[PF]) to generate interrupt requests. 0 PF interrupt requests are disabled. 1 PF interrupt requests are enabled.

52.3.8 UART Data Register (UARTx_D)

This register is actually two separate registers. Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

NOTE

In 8-bit or 9-bit data format, only UART data register (D) needs to be accessed in order to clear the S1[RDRF] bit (assuming receiver buffer level is less than RWFIFO[RXWATER]). The C3 register only needs to be read (prior to the D register) if the ninth bit of data needs to be captured. Likewise the ED register only needs to be read (prior to the D register) if the additional flag data for the dataword needs to be captured.

NOTE

In the normal 8-bit mode (M bit cleared) if the parity is enabled, you get seven data bits and one parity bit. That one parity bit will be loaded into the D register. So if you care about only the data bits, you have to mask off the parity bit from the value you read out of this register.

NOTE

When transmitting in 9-bit data format and using 8-bit write instructions, write first to transmit bit 8 in UART control register 3 (C3[T8]), then D. A write to C3[T8] stores the data in a temporary register. If D register is written first then the new data on data bus is stored in D register, while the temporary value (written by last write to C3[T8]) gets stored in C3[T8] register.

Addresses: UART0_D is 4006_A000h base + 7h offset = 4006_A007h
UART1_D is 4006_B000h base + 7h offset = 4006_B007h
UART2_D is 4006_C000h base + 7h offset = 4006_C007h
UART3_D is 4006_D000h base + 7h offset = 4006_D007h
UART4_D is 400E_A000h base + 7h offset = 400E_A007h
UART5_D is 400E_B000h base + 7h offset = 400E_B007h

Bit	7	6	5	4	3	2	1	0
Read Write	RT							
Reset	0	0	0	0	0	0	0	0

UARTx_D field descriptions

Field	Description
7–0 RT	Reads return the contents of the read-only receive data register and writes go to the write-only transmit data register.

52.3.9 UART Match Address Registers 1 (UARTx_MA1)

The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[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. These registers can be read and written at anytime.

Addresses: UART0_MA1 is 4006_A000h base + 8h offset = 4006_A008h
UART1_MA1 is 4006_B000h base + 8h offset = 4006_B008h
UART2_MA1 is 4006_C000h base + 8h offset = 4006_C008h
UART3_MA1 is 4006_D000h base + 8h offset = 4006_D008h
UART4_MA1 is 400E_A000h base + 8h offset = 400E_A008h
UART5_MA1 is 400E_B000h base + 8h offset = 400E_B008h

Bit	7	6	5	4	3	2	1	0
Read Write	MA							
Reset	0	0	0	0	0	0	0	0

UARTx_MA1 field descriptions

Field	Description
7–0 MA	Match Address

52.3.10 UART Match Address Registers 2 (UARTx_MA2)

These registers can be read and written at anytime. The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[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.

Addresses: UART0_MA2 is 4006_A000h base + 9h offset = 4006_A009h
UART1_MA2 is 4006_B000h base + 9h offset = 4006_B009h
UART2_MA2 is 4006_C000h base + 9h offset = 4006_C009h
UART3_MA2 is 4006_D000h base + 9h offset = 4006_D009h
UART4_MA2 is 400E_A000h base + 9h offset = 400E_A009h
UART5_MA2 is 400E_B000h base + 9h offset = 400E_B009h

Bit	7	6	5	4	3	2	1	0
Read Write	MA							
Reset	0	0	0	0	0	0	0	0

UARTx_MA2 field descriptions

Field	Description
7–0 MA	Match Address

52.3.11 UART Control Register 4 (UARTx_C4)

Addresses: UART0_C4 is 4006_A000h base + Ah offset = 4006_A00Ah
UART1_C4 is 4006_B000h base + Ah offset = 4006_B00Ah
UART2_C4 is 4006_C000h base + Ah offset = 4006_C00Ah
UART3_C4 is 4006_D000h base + Ah offset = 4006_D00Ah
UART4_C4 is 400E_A000h base + Ah offset = 400E_A00Ah
UART5_C4 is 400E_B000h base + Ah offset = 400E_B00Ah

Bit	7	6	5	4	3	2	1	0
Read Write	MAEN1	MAEN2	M10	BRFA				
Reset	0	0	0	0	0	0	0	0

UARTx_C4 field descriptions

Field	Description
7 MAEN1	Match Address Mode Enable 1 Refer to Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN2 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA1 register. If no match occurs, the data is discarded. If match occurs, data is transferred to the data buffer.This bit must be cleared when C7816[ISO7816E] is set/enabled.
6 MAEN2	Match Address Mode Enable 2 Refer to Match address operation for more information. 0 All data received is transferred to the data buffer if MAEN1 is cleared. 1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA2 register. If no match occurs, the data is discarded. If match occurs, data is transferred to the data buffer.This bit must be cleared when C7816[ISO7816E] is set/enabled.
5 M10	10-bit Mode select The M10 bit causes a tenth, non-memory mapped bit to be part of the serial transmission. This tenth bit is generated and interpreted as a parity bit. The M10 bit does not affect the LIN send or detect break behavior. If M10 is set then both C1[M] and C1[PE] bits must also be set. This bit must be cleared when C7816[ISO7816E] is set/enabled. Refer to Data format (non ISO-7816) for more information.

Table continues on the next page...

UARTx_C4 field descriptions (continued)

Field	Description
	0 The parity bit is the ninth bit in the serial transmission. 1 The parity bit is the tenth bit in the serial transmission.
4–0 BRFA	Baud Rate Fine Adjust This bit field is used to add more timing resolution to the average baud frequency, in increments of 1/32. Refer to Baud rate generation for more information.

52.3.12 UART Control Register 5 (UARTx_C5)

Addresses: UART0_C5 is 4006_A000h base + Bh offset = 4006_A00Bh
UART1_C5 is 4006_B000h base + Bh offset = 4006_B00Bh
UART2_C5 is 4006_C000h base + Bh offset = 4006_C00Bh
UART3_C5 is 4006_D000h base + Bh offset = 4006_D00Bh
UART4_C5 is 400E_A000h base + Bh offset = 400E_A00Bh
UART5_C5 is 400E_B000h base + Bh offset = 400E_B00Bh

Bit	7	6	5	4	3	2	1	0
Read	TDMAS	0	RDMAS	0				
Write		▨		▨	▨	▨	▨	▨
Reset	0	0	0	0	0	0	0	0

UARTx_C5 field descriptions

Field	Description
7 TDMAS	Transmitter DMA Select TDMAS configures the transmit data register empty flag, S1[TDRE], to generate interrupt or DMA requests if C2[TIE] is set. **NOTE:** If C2[TIE] is cleared, TDRE DMA and TDRE interrupt request signals are not asserted when the TDRE flag is set, regardless of the state of TDMAS. **NOTE:** If C2[TIE] and TDMAS are both set, then C2[TCIE] must be cleared, and D register must not be written outside of servicing of a DMA request. 0 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE interrupt request signal is asserted to request interrupt service. 1 If C2[TIE] is set and the S1[TDRE] flag is set, the TDRE DMA request signal is asserted to request a DMA transfer.
6 Reserved	This read-only field is reserved and always has the value zero.
5 RDMAS	Receiver Full DMA Select RDMAS configures the receiver data register full flag, S1[RDRF], to generate interrupt or DMA requests if C2[RIE] is set.

Table continues on the next page...

UARTx_C5 field descriptions (continued)

Field	Description
	NOTE: If C2[RIE] is cleared, the RDRF DMA and RDFR interrupt request signals are not asserted when the S1[RDRF] flag is set, regardless of the state of RDMAS. 0 If C2[RIE] is set and the S1[RDRF] flag is set, the RDFR interrupt request signal is asserted to request interrupt service. 1 If C2[RIE] is set and the S1[RDRF] flag is set, the RDRF DMA request signal is asserted to request a DMA transfer.
4–0 Reserved	This read-only field is reserved and always has the value zero.

52.3.13 UART Extended Data Register (UARTx_ED)

This register contains additional information flags that are stored with a received dataword. This register may be read at any time but only contains valid data if there is a dataword in the receive FIFO.

NOTE

The data contained in this register represents additional information regarding the conditions on which a dataword was received. The importance of this data varies with application, and in some cases maybe completely optional. These fields automatically update to reflect the conditions of the next dataword whenever D is read.

NOTE

If the S1[NF] and S1[PF] flags have not been set since the last time the receive buffer was empty, the NOISY and PARITYE bits will be zero.

Addresses: UART0_ED is 4006_A000h base + Ch offset = 4006_A00Ch
UART1_ED is 4006_B000h base + Ch offset = 4006_B00Ch
UART2_ED is 4006_C000h base + Ch offset = 4006_C00Ch
UART3_ED is 4006_D000h base + Ch offset = 4006_D00Ch
UART4_ED is 400E_A000h base + Ch offset = 400E_A00Ch
UART5_ED is 400E_B000h base + Ch offset = 400E_B00Ch

Bit	7	6	5	4	3	2	1	0
Read	NOISY	PARITYE	0					
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_ED field descriptions

Field	Description
7 NOISY	The current received dataword contained in D and C3[R8] was received with noise. 0 The dataword was received without noise. 1 The data was received with noise.
6 PARITYE	The current received dataword contained in D and C3[R8] was received with a parity error. 0 The dataword was received without a parity error. 1 The dataword was received with a parity error.
5–0 Reserved	This read-only field is reserved and always has the value zero.

52.3.14 UART Modem Register (UARTx_MODEM)

The MODEM register controls options for setting the modem configuration.

NOTE

> RXRTSE, TXRTSPOL, TXRTSE and TXCTSE must all be cleared when C7816[ISO7816EN] is enabled. This will cause the RTS to deassert during ISO-7816 wait times. The ISO-7816 protocol does not make use of the RTS and CTS signals.

Addresses: UART0_MODEM is 4006_A000h base + Dh offset = 4006_A00Dh
UART1_MODEM is 4006_B000h base + Dh offset = 4006_B00Dh
UART2_MODEM is 4006_C000h base + Dh offset = 4006_C00Dh
UART3_MODEM is 4006_D000h base + Dh offset = 4006_D00Dh
UART4_MODEM is 400E_A000h base + Dh offset = 400E_A00Dh
UART5_MODEM is 400E_B000h base + Dh offset = 400E_B00Dh

Bit	7	6	5	4	3	2	1	0
Read	0				RXRTSE	TXRTSPOL	TXRTSE	TXCTSE
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_MODEM field descriptions

Field	Description
7–4 Reserved	This read-only field is reserved and always has the value zero.
3 RXRTSE	Receiver request-to-send enable Allows the RTS output to control the CTS input of the transmitting device to prevent receiver overrun. **NOTE:** Do not set both RXRTSE and TXRTSE.

Table continues on the next page...

UARTx_MODEM field descriptions (continued)

Field	Description
	0 The receiver has no effect on RTS. 1 RTS is deasserted if the number of characters in the receiver data register (FIFO) is equal to or greater than RWFIFO[RXWATER]. RTS is asserted when the number of characters in the receiver data register (FIFO) is less than RWFIFO[RXWATER].
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. 0 Transmitter RTS is active low. 1 Transmitter RTS is active high.
1 TXRTSE	Transmitter request-to-send enable Controls RTS before and after a transmission. 0 The transmitter has no effect on RTS. 1 When a character is placed into an empty transmitter data buffer(FIFO), RTS asserts one bit time before the start bit is transmitted. RTS deasserts one bit time after all characters in the transmitter data buffer(FIFO) 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. 0 CTS has no effect on the transmitter. 1 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.

52.3.15 UART Infrared Register (UARTx_IR)

The IR register controls options for setting the infrared configuration.

Addresses: UART0_IR is 4006_A000h base + Eh offset = 4006_A00Eh
UART1_IR is 4006_B000h base + Eh offset = 4006_B00Eh
UART2_IR is 4006_C000h base + Eh offset = 4006_C00Eh
UART3_IR is 4006_D000h base + Eh offset = 4006_D00Eh
UART4_IR is 400E_A000h base + Eh offset = 400E_A00Eh
UART5_IR is 400E_B000h base + Eh offset = 400E_B00Eh

Bit	7	6	5	4	3	2	1	0
Read	0					IREN	TNP	
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_IR field descriptions

Field	Description
7–3 Reserved	This read-only field is reserved and always has the value zero.
2 IREN	Infrared enable This bit enables/disables the infrared modulation/demodulation. 0 IR disabled. 1 IR enabled.
1–0 TNP	Transmitter narrow pulse These bits enable whether the UART transmits a 1/16, 3/16, 1/32 or 1/4 narrow pulse. 00 3/16. 01 1/16. 10 1/32. 11 1/4.

52.3.16 UART FIFO Parameters (UARTx_PFIFO)

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 should only be written when the C2[RE] and C2[TE] bits are cleared / not set and when the data buffer/FIFO is empty.

Addresses: UART0_PFIFO is 4006_A000h base + 10h offset = 4006_A010h
UART1_PFIFO is 4006_B000h base + 10h offset = 4006_B010h
UART2_PFIFO is 4006_C000h base + 10h offset = 4006_C010h
UART3_PFIFO is 4006_D000h base + 10h offset = 4006_D010h
UART4_PFIFO is 400E_A000h base + 10h offset = 400E_A010h
UART5_PFIFO is 400E_B000h base + 10h offset = 400E_B010h

Bit	7	6	5	4	3	2	1	0
Read	TXFE	TXFIFOSIZE			RXFE	RXFIFOSIZE		
Write								
Reset	0	*	*	*	0	*	*	*

* Notes:
- TXFIFOSIZE bitfield: The reset value depends on whether the specific UART instance supports the FIFO and on the size of that FIFO. See the Chip Configuration details for more information on the FIFO size supported for each UART instance.
- RXFIFOSIZE bitfield: The reset value depends on whether the specific UART instance supports the FIFO and on the size of that FIFO. See the Chip Configuration details for more information on the FIFO size supported for each UART instance.

UARTx_PFIFO field descriptions

Field	Description
7 TXFE	Transmit FIFO Enable When this bit is set the built in FIFO structure for the transmit buffer is enabled. The size of the FIFO structure is indicated by the TXFIFOSIZE field. If this bit is not set then the transmit buffer operates as a FIFO of depth one dataword regardless of the value in TXFIFOSIZE. Both C2[TE] and C2[RE] must be cleared prior to changing this bit. Additionally TXFLUSH and RXFLUSH commands should be issued immediately after changing this bit. 0 Transmit FIFO is not enabled. Buffer is depth 1. (Legacy support). 1 Transmit FIFO is enabled. Buffer is depth indicted 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. 000 Transmit FIFO/Buffer Depth = 1 Dataword. 001 Transmit FIFO/Buffer Depth = 4 Datawords. 010 Transmit FIFO/Buffer Depth = 8 Datawords. 011 Transmit FIFO/Buffer Depth = 16 Datawords. 100 Transmit FIFO/Buffer Depth = 32 Datawords. 101 Transmit FIFO/Buffer Depth = 64 Datawords. 110 Transmit FIFO/Buffer Depth = 128 Datawords. 111 Reserved.
3 RXFE	Receive FIFO Enable When this bit 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 bit is not set then the receive buffer operates as a FIFO of depth one dataword regardless of the value in RXFIFOSIZE. Both C2[TE] and C2[RE] must be cleared prior to changing this bit. Additionally TXFLUSH and RXFLUSH commands should be issued immediately after changing this bit. 0 Receive FIFO is not enabled. Buffer is depth 1. (Legacy support) 1 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. 000 Receive FIFO/Buffer Depth = 1 Dataword. 001 Receive FIFO/Buffer Depth = 4 Datawords. 010 Receive FIFO/Buffer Depth = 8 Datawords. 011 Receive FIFO/Buffer Depth = 16 Datawords. 100 Receive FIFO/Buffer Depth = 32 Datawords. 101 Receive FIFO/Buffer Depth = 64 Datawords. 110 Receive FIFO/Buffer Depth = 128 Datawords. 111 Reserved.

52.3.17 UART FIFO Control Register (UARTx_CFIFO)

This register provides the ability to program various control bits for FIFO operation. This register may be read or written at any time. Note that writing the TXFLUSH and RXFLUSH bits may result in data loss and requires careful action to prevent unintended / unpredictable behavior, hence it is recommended that TE and RE be cleared prior to flushing the corresponding FIFO.

Addresses: UART0_CFIFO is 4006_A000h base + 11h offset = 4006_A011h
UART1_CFIFO is 4006_B000h base + 11h offset = 4006_B011h
UART2_CFIFO is 4006_C000h base + 11h offset = 4006_C011h
UART3_CFIFO is 4006_D000h base + 11h offset = 4006_D011h
UART4_CFIFO is 400E_A000h base + 11h offset = 400E_A011h
UART5_CFIFO is 400E_B000h base + 11h offset = 400E_B011h

Bit	7	6	5	4	3	2	1	0
Read	0	0	0				TXOFE	RXUFE
Write	TXFLUSH	RXFLUSH						
Reset	0	0	0	0	0	0	0	0

UARTx_CFIFO field descriptions

Field	Description
7 TXFLUSH	Transmit FIFO/Buffer Flush Writing to this bit 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. 0 No flush operation occurs. 1 All data in the transmit FIFO/Buffer is cleared out.
6 RXFLUSH	Receive FIFO/Buffer Flush Writing to this bit 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. 0 No flush operation occurs. 1 All data in the receive FIFO/buffer is cleared out.
5–2 Reserved	This read-only field is reserved and always has the value zero.
1 TXOFE	Transmit FIFO Overflow Interrupt Enable When this bit is set the TXOF flag will generate an interrupt to the host. 0 TXOF flag does not generate an interrupt to the host. 1 TXOF flag generates an interrupt to the host.
0 RXUFE	Receive FIFO Underflow Interrupt Enable When this bit is set the RXUF flag will generate an interrupt to the host.

Table continues on the next page...

UARTx_CFIFO field descriptions (continued)

Field	Description
	0 RXUF flag does not generate an interrupt to the host. 1 RXUF flag generates an interrupt to the host.

52.3.18 UART FIFO Status Register (UARTx_SFIFO)

This register provides various status information regarding the transmit and receiver buffers/FIFOs, including interrupt information. This register may be written or read at anytime.

Addresses: UART0_SFIFO is 4006_A000h base + 12h offset = 4006_A012h
UART1_SFIFO is 4006_B000h base + 12h offset = 4006_B012h
UART2_SFIFO is 4006_C000h base + 12h offset = 4006_C012h
UART3_SFIFO is 4006_D000h base + 12h offset = 4006_D012h
UART4_SFIFO is 400E_A000h base + 12h offset = 400E_A012h
UART5_SFIFO is 400E_B000h base + 12h offset = 400E_B012h

Bit	7	6	5	4	3	2	1	0
Read	TXEMPT	RXEMPT	0				TXOF	RXUF
Write								
Reset	1	1	0	0	0	0	0	0

UARTx_SFIFO field descriptions

Field	Description
7 TXEMPT	Transmit Buffer/FIFO Empty This status bit asserts when there is no data in the Transmit FIFO/buffer. This bit does not take into account data that is in the transmit shift register. 0 Transmit buffer is not empty. 1 Transmit buffer is empty.
6 RXEMPT	Receive Buffer/FIFO Empty This status bit asserts when there is no data in the receive FIFO/Buffer. This bit does not take into account data that is in the receive shift register. 0 Receive buffer is not empty. 1 Receive buffer is empty.
5–2 Reserved	This read-only field is reserved and always has the value zero.
1 TXOF	Transmitter Buffer Overflow Flag This flag indicates that more data has been written to the transmit buffer than it can hold. This bit will assert regardless of the value of CFIFO[TXOFE]. However, an interrupt will only be issued to the host if the CFIFO[TXOFE] bit is set. This flag is cleared by writing a "1".

Table continues on the next page...

UARTx_SFIFO field descriptions (continued)

Field	Description
	0 No transmit buffer overflow has occurred since the last time the flag was cleared. 1 At least one transmit buffer overflow has occurred since the last time the flag was cleared.
0 RXUF	Receiver Buffer Underflow Flag This flag indicates that more data has been read from the receive buffer than was present. This bit will assert regardless of the value of CFIFO[RXUFE]. However, an interrupt will only be issued to the host if the CFIFO[RXUFE] bit is set. This flag is cleared by writing a "1". 0 No receive buffer underflow has occurred since the last time the flag was cleared. 1 At least one receive buffer underflow has occurred since the last time the flag was cleared.

52.3.19 UART FIFO Transmit Watermark (UARTx_TWFIFO)

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 should only be written when C2[TE] is not set. Changing the value of the watermark will not clear the S1[TDRE] flag.

Addresses: UART0_TWFIFO is 4006_A000h base + 13h offset = 4006_A013h
UART1_TWFIFO is 4006_B000h base + 13h offset = 4006_B013h
UART2_TWFIFO is 4006_C000h base + 13h offset = 4006_C013h
UART3_TWFIFO is 4006_D000h base + 13h offset = 4006_D013h
UART4_TWFIFO is 400E_A000h base + 13h offset = 400E_A013h
UART5_TWFIFO is 400E_B000h base + 13h offset = 400E_B013h

Bit	7	6	5	4	3	2	1	0
Read Write	TXWATER							
Reset	0	0	0	0	0	0	0	0

UARTx_TWFIFO field descriptions

Field	Description
7–0 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 then an interrupt via S1[TDRE] or a DMA request via C5[TDMAS] will be generated as determined by C5[TDMAS] and C2[TIE] fields. For proper operation the value in the TXWATER field must be set to be less than the size of the transmit buffer/FIFO size as indicated by PFIFO[TXFIFOSIZE] and PFIFO[TXFE].

52.3.20 UART FIFO Transmit Count (UARTx_TCFIFO)

This is a read only register that indicates how many datawords are currently in the transmit buffer/FIFO. It may be read at anytime.

Addresses: UART0_TCFIFO is 4006_A000h base + 14h offset = 4006_A014h
UART1_TCFIFO is 4006_B000h base + 14h offset = 4006_B014h
UART2_TCFIFO is 4006_C000h base + 14h offset = 4006_C014h
UART3_TCFIFO is 4006_D000h base + 14h offset = 4006_D014h
UART4_TCFIFO is 400E_A000h base + 14h offset = 400E_A014h
UART5_TCFIFO is 400E_B000h base + 14h offset = 400E_B014h

Bit	7	6	5	4	3	2	1	0
Read	TXCOUNT							
Write	▨							
Reset	0	0	0	0	0	0	0	0

UARTx_TCFIFO field descriptions

Field	Description
7–0 TXCOUNT	Transmit Counter The value in this register indicates the number of datawords that are in the transmit buffer/FIFO. If a dataword is in the process of being transmitted (i.e. in the transmit shift register) it is not included in the count. This value may be used in conjunction with the PFIFO[TXFIFOSIZE] field to calculate how much room is left in the transmit buffer/FIFO.

52.3.21 UART FIFO Receive Watermark (UARTx_RWFIFO)

This register provides the ability to set a programmable threshold for notification of needing to remove data from the receiver buffer/FIFO. This register may be read at any time but should only be written when C2[RE] is not asserted. Changing the value in this register will not clear the S1[RDRF] flag.

Addresses: UART0_RWFIFO is 4006_A000h base + 15h offset = 4006_A015h
UART1_RWFIFO is 4006_B000h base + 15h offset = 4006_B015h
UART2_RWFIFO is 4006_C000h base + 15h offset = 4006_C015h
UART3_RWFIFO is 4006_D000h base + 15h offset = 4006_D015h
UART4_RWFIFO is 400E_A000h base + 15h offset = 400E_A015h
UART5_RWFIFO is 400E_B000h base + 15h offset = 400E_B015h

Bit	7	6	5	4	3	2	1	0
Read	RXWATER							
Write								
Reset	0	0	0	0	0	0	0	1

UARTx_RWFIFO field descriptions

Field	Description
7–0 RXWATER	Receive Watermark When the number of datawords in the Receive FIFO/buffer is equal to or greater than the value in this register field the event is flagged. An interrupt via S1[RDRF] or a DMA request via C5[RDMAS] will be generated as determined by C5[RDMAS] and C2[RIE] fields. For proper operation the value in the RXWATER field must be set to be less than the size of the Receive buffer/FIFO size as indicated by PFIFO[RXFIFOSIZE] and PFIFO[RXFE] and greater than 0.

52.3.22 UART FIFO Receive Count (UARTx_RCFIFO)

This is a read only register that indicates how many datawords are currently in the receive buffer/FIFO. It may be read at anytime.

Addresses: UART0_RCFIFO is 4006_A000h base + 16h offset = 4006_A016h
UART1_RCFIFO is 4006_B000h base + 16h offset = 4006_B016h
UART2_RCFIFO is 4006_C000h base + 16h offset = 4006_C016h
UART3_RCFIFO is 4006_D000h base + 16h offset = 4006_D016h
UART4_RCFIFO is 400E_A000h base + 16h offset = 400E_A016h
UART5_RCFIFO is 400E_B000h base + 16h offset = 400E_B016h

Bit	7	6	5	4	3	2	1	0
Read	RXCOUNT							
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_RCFIFO field descriptions

Field	Description
7–0 RXCOUNT	Receive Counter The value in this register indicates the number of datawords that are in the receive buffer/FIFO. If a dataword is in the process of being received (i.e. in the receive shift register) it is not included in the count. This value may be used in conjunction with the PFIFO[RXFIFOSIZE] field to calculate how much room is left in the receive buffer/FIFO.

52.3.23 UART 7816 Control Register (UARTx_C7816)

The C7816 register is the primary control register for ISO-7816 specific functionality. This register is specific to 7816 functionality and the values in this register have no effect on UART operation and should be ignored if ISO_7816E is not set/enabled. This register may be read at anytime but values should only be changed when the ISO_7816E bit is not set.

Addresses: UART0_C7816 is 4006_A000h base + 18h offset = 4006_A018h
UART1_C7816 is 4006_B000h base + 18h offset = 4006_B018h
UART2_C7816 is 4006_C000h base + 18h offset = 4006_C018h
UART3_C7816 is 4006_D000h base + 18h offset = 4006_D018h
UART4_C7816 is 400E_A000h base + 18h offset = 400E_A018h
UART5_C7816 is 400E_B000h base + 18h offset = 400E_B018h

Bit	7	6	5	4	3	2	1	0
Read	0			ONACK	ANACK	INIT	TTYPE	ISO_7816E
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_C7816 field descriptions

Field	Description
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 ONACK	Generate NACK on Overflow When this bit is set, the receiver will automatically generate a NACK response if a receive buffer overrun occurs as indicated by the S1[OR] field. In many systems this will result in the transmitter resending the packet that overflowed until the retransmit threshold for that transmitter has been reached. A NACK is only generated if TTYPE=0. This bit operates independently of ANACK. See Overrun NACK considerations. 0 The received data does not generate a NACK when the receipt of the data results in an overflow event. 1 If the receiver buffer overflows, a NACK is automatically sent on a received character.
3 ANACK	Generate NACK on Error When this bit is set, the receiver will automatically generate a NACK response if a parity error occurs or if INIT is set and an invalid initial character is detected. A NACK is only generated if TTYPE = 0. If ANACK is set the UART will attempt to retransmit the data indefinitely. To stop retransmission attempts, clear C2[TE] or ISO_7816E and do not set until S1[TC] set C2[TE] again. 0 No NACK is automatically generated. 1 A NACK is automatically generated if a parity error is detected or if an invalid initial character is detected.
2 INIT	Detect Initial Character

Table continues on the next page...

UARTx_C7816 field descriptions (continued)

Field	Description
	When this bit is set, all received characters will be searched for a valid initial character. If an invalid initial character is identified then a NACK will be sent if ANACK is set. All received data is discarded and error flags blocked (S1[NF], S1[OR], S1[FE], S1[PF], IS7816[WT], IS7816[CWT], IS7816[BWT], IS7816[GTV]) until a valid initial character is detected. Upon detection of a valid initial character the configuration values S2[MSBF], C3[TXINV] and S2[RXINV] are automatically updated to reflect the initial character that was received. The actual INIT data value is not stored in the receive buffer. Additionally, upon detection of a valid initial character the IS7816[INITD] flag is set and an interrupt issued as programmed by the IE7816[INITDE] bit. When a valid initial character is detected the INIT bit is automatically cleared. This Initial Character Detect feature is only supported in T = 0 protocol mode. 0 Normal operating mode. Receiver does not seek to identify initial character. 1 Receiver searches for initial character.
1 TTYPE	Transfer Type This bit indicates the transfer protocol being used. Refer to ISO-7816 / smartcard support for more details. 0 T = 0 Per the ISO-7816 specification. 1 T = 1 Per the ISO-7816 specification.
0 ISO_7816E	ISO-7816 Functionality Enabled This bit indicates that the UART is operating according to the ISO-7816 protocol. **NOTE:** This bit should only be modified when no transmit or receive is occurring. If this bit is changed during a data transfer the data being transmitted or received may be transferred incorrectly. 0 ISO-7816 functionality is turned off / not enabled. 1 ISO-7816 functionality is turned on / enabled.

52.3.24 UART 7816 Interrupt Enable Register (UARTx_IE7816)

The IE7816 register controls which flags result in an interrupt being issued. This register is specific to 7816 functionality, the corresponding flags that drive the interrupts will not assert when 7816E is not set/enabled. However, these flags may remain set if they asserted while 7816E was set and not subsequently cleared. This register maybe read or written at anytime.

Addresses: UART0_IE7816 is 4006_A000h base + 19h offset = 4006_A019h
UART1_IE7816 is 4006_B000h base + 19h offset = 4006_B019h
UART2_IE7816 is 4006_C000h base + 19h offset = 4006_C019h
UART3_IE7816 is 4006_D000h base + 19h offset = 4006_D019h
UART4_IE7816 is 400E_A000h base + 19h offset = 400E_A019h
UART5_IE7816 is 400E_B000h base + 19h offset = 400E_B019h

Bit	7	6	5	4	3	2	1	0
Read	WTE	CWTE	BWTE	INITDE	0	GTVE	TXTE	RXTE
Write					▨			
Reset	0	0	0	0	0	0	0	0

UARTx_IE7816 field descriptions

Field	Description
7 WTE	Wait Timer Interrupt Enable 0 The assertion of the IS7816[WT] bit will not result in the generation of an interrupt. 1 The assertion of the IS7816[WT] bit will result in the generation of an interrupt.
6 CWTE	Character Wait Timer Interrupt Enable 0 The assertion of the IS7816[CWT] bit will not result in the generation of an interrupt. 1 The assertion of the IS7816[CWT] bit will result in the generation of an interrupt.
5 BWTE	Block Wait Timer Interrupt Enable 0 The assertion of the IS7816[BWT] bit will not result in the generation of an interrupt. 1 The assertion of the IS7816[BWT] bit will result in the generation of an interrupt.
4 INITDE	Initial Character Detected Interrupt Enable 0 The assertion of the IS7816[INITD] bit will not result in the generation of an interrupt. 1 The assertion of the IS7816[INITD] bit will result in the generation of an interrupt.
3 Reserved	This read-only field is reserved and always has the value zero.
2 GTVE	Guard Timer Violated Interrupt Enable 0 The assertion of the IS7816[GTV] bit will not result in the generation of an interrupt. 1 The assertion of the IS7816[GTV] bit will result in the generation of an interrupt.

Table continues on the next page...

UARTx_IE7816 field descriptions (continued)

Field	Description
1 TXTE	Transmit Threshold Exceeded Interrupt Enable 0 The assertion of the IS7816[TXT] bit will not result in the generation of an interrupt. 1 The assertion of the IS7816[TXT] bit will result in the generation of an interrupt.
0 RXTE	Receive Threshold Exceeded Interrupt Enable 0 The assertion of the IS7816[RXT] bit will not result in the generation of an interrupt. 1 The assertion of the IS7816[RXT] bit will result in the generation of an interrupt.

52.3.25 UART 7816 Interrupt Status Register (UARTx_IS7816)

The IS7816 register provides a mechanism to read and clear the interrupt flags. All flags/ interrupts are cleared by writing a "1" to the bit location. Writing a "0" has no effect. All bits are "sticky", meaning they only indicate that the flag condition occurred since the last time the bit was cleared not that the condition currently exists. The status flags are set regardless of if the corresponding bit in the IC7816 is set or cleared, the IC7816 only controls if a interrupt is issued to the host processor. This register is specific to 7816 functionality and the values in this register have no affect on UART operation and should be ignored if 7816E is not set/enabled. This register may be read or written at anytime.

Addresses: UART0_IS7816 is 4006_A000h base + 1Ah offset = 4006_A01Ah
UART1_IS7816 is 4006_B000h base + 1Ah offset = 4006_B01Ah
UART2_IS7816 is 4006_C000h base + 1Ah offset = 4006_C01Ah
UART3_IS7816 is 4006_D000h base + 1Ah offset = 4006_D01Ah
UART4_IS7816 is 400E_A000h base + 1Ah offset = 400E_A01Ah
UART5_IS7816 is 400E_B000h base + 1Ah offset = 400E_B01Ah

Bit	7	6	5	4	3	2	1	0
Read	WT	CWT	BWT	INITD	0	GTV	TXT	RXT
Write					▨			
Reset	0	0	0	0	0	0	0	0

UARTx_IS7816 field descriptions

Field	Description
7 WT	Wait Timer Interrupt This flag indicates that the wait time, the time between the leading edge of a character being transmitted and the leading edge of the next response character has exceeded the programed value. This flag only asserts when C7816[TTYPE] = 0. This interrupt is cleared by writing `1'. 0 Wait time (WT) has not been violated. 1 Wait time (WT) has been violated.

Table continues on the next page...

UARTx_IS7816 field descriptions (continued)

Field	Description
6 CWT	Character Wait Timer Interrupt This flag indicates that the character wait time, the time between the leading edges of two consecutive characters in a block has exceed the programed value. This flag only asserts when C7816[TTYPE] = 1. This interrupt is cleared by writing `1'. 0 Character wait time (CWT) has not been violated. 1 Character wait time (CWT) has been violated.
5 BWT	Block Wait Timer Interrupt This flag indicates that the block wait time, the time between the leading edge of first received character of a block and the leading edge of the last character the previously transmitted block. This flag only asserts when C7816[TTYPE] = 1.This interrupt is cleared by writing '1'. 0 Block wait time (BWT) has not been violated. 1 Block wait tTime (BWT) has been violated.
4 INITD	Initial Character Detected Interrupt This flag indicates that a valid initial character was received. This interrupt is cleared by writing `1'. 0 A valid initial character has not been received. 1 A valid initial character has been received.
3 Reserved	This read-only field is reserved and always has the value zero.
2 GTV	Guard Timer Violated Interrupt This flag indicates that one or more of the character guard time, block guard time or guard time were violated. This interrupt is cleared by writing `1'. 0 A guard time (GT, CGT or BGT) has not been violated. 1 A guard time (GT, CGT or BGT) has been violated.
1 TXT	Transmit Threshold Exceeded Interrupt This flag indicates that the transmit NACK threshold has been exceeded as indicated by the ET7816[TXTHRESHOLD] field. Regardless if this flag is set, the UART will continue to retransmit indefinitely. This flag only asserts when C7816[TTYPE] = 0. If 7816E is cleared/disabled, ANACK is cleared/disabled, C2[TE] is cleared/disabled, C7816[TTYPE] = 1 or packet is transferred without receiving a NACK the internal NACK detection counter is cleared and the count restarts from zero on the next received NACK. This interrupt is cleared by writing `1'. 0 The number of retries and corresponding NACKS does not exceed the value in the ET7816[TXTHRESHOLD] field. 1 The number of retries and corresponding NACKS exceeds the value in the ET7816[TXTHRESHOLD] field.
0 RXT	Receive Threshold Exceeded Interrupt This flag indicates that there were more than ET7816[RXTHRESHOLD] consecutive NACKS generated in response to parity errors on received data. This flag requires ANACK to be set. Additionally, this flag only asserts when C7816[TTYPE] = 0. Clearing this bit also resets the counter keeping track of consecutive NACKS. The UART will continue to attempt to receive data regardless of if this flag is set. If 7816E is cleared/disabled, RE is cleared/disabled, C7816[TTYPE] = 1 or packet is received without needing to

Table continues on the next page...

UARTx_IS7816 field descriptions (continued)

Field	Description
	issue a NACK, the internal NACK detection counter is cleared and the count restarts from zero on the next transmitted NACK. This interrupt is cleared by writing `1'. 0 The number of consecutive NACKS generated as a result of parity errors and buffer overruns is less than or equal to the value in ET7816[RXTHRESHOLD]. 1 The number of consecutive NACKS generated as a result of parity errors and buffer overruns is greater than the value in ET7816[RXTHRESHOLD].

52.3.26 UART 7816 Wait Parameter Register (UARTx_WP7816T0)

The WP7816 register contains constants used in the generation of various wait timer counters. To save register space this register is used differently when C7816[TTYPE] = 0 and C7816[TTYPE] = 1. This register may be read at anytime. This register must only be written when C7816[ISO_7816E] is not set.

Addresses: UART0_WP7816T0 is 4006_A000h base + 1Bh offset = 4006_A01Bh
UART1_WP7816T0 is 4006_B000h base + 1Bh offset = 4006_B01Bh
UART2_WP7816T0 is 4006_C000h base + 1Bh offset = 4006_C01Bh
UART3_WP7816T0 is 4006_D000h base + 1Bh offset = 4006_D01Bh
UART4_WP7816T0 is 400E_A000h base + 1Bh offset = 400E_A01Bh
UART5_WP7816T0 is 400E_B000h base + 1Bh offset = 400E_B01Bh

Bit	7	6	5	4	3	2	1	0
Read Write	WI							
Reset	0	0	0	0	1	0	1	0

UARTx_WP7816T0 field descriptions

Field	Description
7–0 WI	Wait Timer Interrupt (C7816[TTYPE] = 0) This value is used to calculate the value used for the WT counter. It represents a value between 1 and 255. The value of zero is not valid. This value is only used when C7816[TTYPE] = 0. See Wait time and guard time parameters.

52.3.27 UART 7816 Wait Parameter Register (UARTx_WP7816T1)

The WP7816 register contains constants used in the generation of various wait timer counters. To save register space this register is used differently when C7816[TTYPE] = 0 and C7816[TTYPE] = 1. This register maybe read at anytime. This register must only be written when C7816[ISO_7816E] is not set.

Addresses: UART0_WP7816T1 is 4006_A000h base + 1Bh offset = 4006_A01Bh
UART1_WP7816T1 is 4006_B000h base + 1Bh offset = 4006_B01Bh
UART2_WP7816T1 is 4006_C000h base + 1Bh offset = 4006_C01Bh
UART3_WP7816T1 is 4006_D000h base + 1Bh offset = 4006_D01Bh
UART4_WP7816T1 is 400E_A000h base + 1Bh offset = 400E_A01Bh
UART5_WP7816T1 is 400E_B000h base + 1Bh offset = 400E_B01Bh

Bit	7	6	5	4	3	2	1	0
Read Write	CWI				BWI			
Reset	0	0	0	0	1	0	1	0

UARTx_WP7816T1 field descriptions

Field	Description
7–4 CWI	Character Wait Time Integer (C7816[TTYPE] = 1) This value is used to calculate the value used for the CWT counter. It represents a value between 0 and 15. This value is only used when C7816[TTYPE] = 1. See Wait time and guard time parameters.
3–0 BWI	Block Wait Time Integer(C7816[TTYPE] = 1) This value is used to calculate the value used for the BWT counter. It represent a value between 0 and 15. This value is only used when C7816[TTYPE] = 1. See Wait time and guard time parameters.

52.3.28 UART 7816 Wait N Register (UARTx_WN7816)

The WN7816 register contains a parameter that is used in the calculation of the guard time counter. This register may be read at anytime. This register must only be written when C7816[ISO_7816E] is not set.

Addresses: UART0_WN7816 is 4006_A000h base + 1Ch offset = 4006_A01Ch
UART1_WN7816 is 4006_B000h base + 1Ch offset = 4006_B01Ch
UART2_WN7816 is 4006_C000h base + 1Ch offset = 4006_C01Ch
UART3_WN7816 is 4006_D000h base + 1Ch offset = 4006_D01Ch
UART4_WN7816 is 400E_A000h base + 1Ch offset = 400E_A01Ch
UART5_WN7816 is 400E_B000h base + 1Ch offset = 400E_B01Ch

Bit	7	6	5	4	3	2	1	0
Read Write	GTN							
Reset	0	0	0	0	0	0	0	0

UARTx_WN7816 field descriptions

Field	Description
7–0 GTN	Guard Band N This register field defines a parameter used in the calculation of GT, CGT and BGT counters. The value represents an integer number 0-255. See Wait time and guard time parameters.

52.3.29 UART 7816 Wait FD Register (UARTx_WF7816)

The WF7816 contains parameters that are used in the generation of various counters including GT, CGT, BGT, WT and BWT. This register may be read from at anytime. This register must only be written to when C7816[ISO_7816E] is not set.

Addresses: UART0_WF7816 is 4006_A000h base + 1Dh offset = 4006_A01Dh
UART1_WF7816 is 4006_B000h base + 1Dh offset = 4006_B01Dh
UART2_WF7816 is 4006_C000h base + 1Dh offset = 4006_C01Dh
UART3_WF7816 is 4006_D000h base + 1Dh offset = 4006_D01Dh
UART4_WF7816 is 400E_A000h base + 1Dh offset = 400E_A01Dh
UART5_WF7816 is 400E_B000h base + 1Dh offset = 400E_B01Dh

Bit	7	6	5	4	3	2	1	0
Read Write	GTFD							
Reset	0	0	0	0	0	0	0	1

UARTx_WF7816 field descriptions

Field	Description
7–0 GTFD	FD Multiplier This field is used as another multiplier in the calculation of WT and BWT. This values represents a number between 1 and 255. The value of 0 is invalid. This value is NOT used in baud rate generation. See Wait time and guard time parameters and Baud rate generation.

52.3.30 UART 7816 Error Threshold Register (UARTx_ET7816)

The ET7816 register contains fields that determine the number of NACKs that must be received or transmitted before the host processor is notified. This register may be read at anytime. This register must only be written when C7816[ISO_7816E] is not set.

Addresses: UART0_ET7816 is 4006_A000h base + 1Eh offset = 4006_A01Eh
UART1_ET7816 is 4006_B000h base + 1Eh offset = 4006_B01Eh
UART2_ET7816 is 4006_C000h base + 1Eh offset = 4006_C01Eh
UART3_ET7816 is 4006_D000h base + 1Eh offset = 4006_D01Eh
UART4_ET7816 is 400E_A000h base + 1Eh offset = 400E_A01Eh
UART5_ET7816 is 400E_B000h base + 1Eh offset = 400E_B01Eh

Bit	7	6	5	4	3	2	1	0
Read Write	TXTHRESHOLD				RXTHRESHOLD			
Reset	0	0	0	0	0	0	0	0

UARTx_ET7816 field descriptions

Field	Description
7–4 TXTHRESHOLD	Transmit NACK Threshold The value written to this field indicates the maximum number of failed attempts (NACKs) a transmitted character can have before the host processor is notified. Meaning a value of 0 will always result in TXT asserting on the first NACK that is received. A value of 1 will result in TXT being asserted on the second NACK that is received. This field is only meaningful when C7816[TTYPE] = 0 and C7816[ANACK] = 1. The value read from this field represents the number of consecutive NACKs that have been received since the last successful transmission. This counter saturates at 4'hF and does not wrap around. Regardless of how many NACKs that are received, the UART will continue to retransmit indefinitely. This flag only asserts when C7816[TTYPE] = 0. For additional information see the IS7816[TXT] bit description.
3–0 RXTHRESHOLD	Receive NACK Threshold The value written to this field indicates the maximum number of consecutive NACKs generated as a result of a parity error or receiver buffer overruns before the host processor is notified. Once the counter exceeds that value in the field the IS7816[RXT] will be asserted. This field is only meaningful when C7816[TTYPE] = 0. The value read from this field represents the number of consecutive NACKs that have been transmitted since the last successful reception. This counter saturates at 4'hF and does not wrap around. Regardless of the number of NACKs sent, the UART will continue to receive valid packets indefinitely. For additional information see IS7816[RXT] bit description.

52.3.31 UART 7816 Transmit Length Register (UARTx_TL7816)

The TL7816 register is used to indicate how many characters are contained in the block being transmitted. This register is only used when C7816[TTYPE] = 1. This register may be read at anytime. This register should only be written when C2[TE] is not enabled.

Addresses: UART0_TL7816 is 4006_A000h base + 1Fh offset = 4006_A01Fh
UART1_TL7816 is 4006_B000h base + 1Fh offset = 4006_B01Fh
UART2_TL7816 is 4006_C000h base + 1Fh offset = 4006_C01Fh
UART3_TL7816 is 4006_D000h base + 1Fh offset = 4006_D01Fh
UART4_TL7816 is 400E_A000h base + 1Fh offset = 400E_A01Fh
UART5_TL7816 is 400E_B000h base + 1Fh offset = 400E_B01Fh

Bit	7	6	5	4	3	2	1	0
Read Write	TLEN							
Reset	0	0	0	0	0	0	0	0

UARTx_TL7816 field descriptions

Field	Description
7–0 TLEN	Transmit Length This value plus 4 indicates the number of characters contained in the block being transmitted. This register is automatically decremented by 1 for each character in the information field portion of the block. Additionally, this register is automatically decremented by 1 for the first character of a CRC in the epilogue field. Hence, this register should be programmed with the number of bytes in the data packet if a LRC is being transmitted, and the number of bytes + 1 if a CRC is being transmitted. This register is not decremented for characters that are assumed to be part of the Prologue field (first three characters transmitted in a block) or the LRC or last CRC character in the Epilogue field (last character transmitted). This field should only be programed or adjusted when C2[TE] is cleared.

52.3.32 UART CEA709.1-B Control Register 6 (UARTx_C6)

Addresses: UART0_C6 is 4006_A000h base + 21h offset = 4006_A021h
UART1_C6 is 4006_B000h base + 21h offset = 4006_B021h
UART2_C6 is 4006_C000h base + 21h offset = 4006_C021h
UART3_C6 is 4006_D000h base + 21h offset = 4006_D021h
UART4_C6 is 400E_A000h base + 21h offset = 400E_A021h
UART5_C6 is 400E_B000h base + 21h offset = 400E_B021h

Bit	7	6	5	4	3	2	1	0
Read	EN709	TX709	CE	CP	0			
Write					▨			
Reset	0	0	0	0	0	0	0	0

UARTx_C6 field descriptions

Field	Description
7 EN709	EN709 This register is used to enable the CEA709.1-B feature. 0 CEA709.1-B is disabled. 1 CEA709.1-B is enabled
6 TX709	CEA709.1-B Transmit Enable This register is used to start CEA709.1-B transmission. 0 CEA709.1-B transmitter is disabled. 1 CEA709.1-B transmitter is enabled.
5 CE	Collision Enable This bit enables the collision detect functionality. 0 Collision detect feature is disabled. 1 Collision detect feature is enabled.
4 CP	Collision Signal Polarity This bit indicates the polarity of collision signal. 0 Collision signal is active low. 1 Collision signal is active high.
3–0 Reserved	This read-only field is reserved and always has the value zero.

52.3.33 UART CEA709.1-B Packet Cycle Time Counter High (UARTx_PCTH)

Addresses: UART0_PCTH is 4006_A000h base + 22h offset = 4006_A022h
UART1_PCTH is 4006_B000h base + 22h offset = 4006_B022h
UART2_PCTH is 4006_C000h base + 22h offset = 4006_C022h
UART3_PCTH is 4006_D000h base + 22h offset = 4006_D022h
UART4_PCTH is 400E_A000h base + 22h offset = 400E_A022h
UART5_PCTH is 400E_B000h base + 22h offset = 400E_B022h

Bit	7	6	5	4	3	2	1	0
Read Write	PCTH							
Reset	0	0	0	0	0	0	0	0

UARTx_PCTH field descriptions

Field	Description
7–0 PCTH	Packet Cycle Time Counter High This register indicates the most significant byte of maximum period after the line code violation for which the bus could remain idle without decrementing back log count. If the time elapsed after line code violation is greater than packet cycle timer then packet cycle timer expired interrupt is generated. It is measured in terms of bit times, i.e the time that it takes for a single bit or one differential Manchester symbol to be transmitted. This is medium dependent and hence does not usually require adjustment and is programmed only once.

52.3.34 UART CEA709.1-B Packet Cycle Time Counter Low (UARTx_PCTL)

Addresses: UART0_PCTL is 4006_A000h base + 23h offset = 4006_A023h
UART1_PCTL is 4006_B000h base + 23h offset = 4006_B023h
UART2_PCTL is 4006_C000h base + 23h offset = 4006_C023h
UART3_PCTL is 4006_D000h base + 23h offset = 4006_D023h
UART4_PCTL is 400E_A000h base + 23h offset = 400E_A023h
UART5_PCTL is 400E_B000h base + 23h offset = 400E_B023h

Bit	7	6	5	4	3	2	1	0
Read Write	PCTL							
Reset	0	0	0	0	0	0	0	0

UARTx_PCTL field descriptions

Field	Description
7–0 PCTL	Packet Cycle Time Counter Low This register indicates the least significant byte of maximum period after the line code violation for which the bus could remain idle without decrementing back log count. If the time elapsed after line code violation is greater than packet cycle timer, then packet cycle timer expired interrupt is generated. It is measured in terms of bit times, i.e the time that it takes for a single bit or one Differential Manchester symbol to be transmitted. This is medium dependent and hence does not usually require adjustment and is programmed only once.

52.3.35 UART CEA709.1-B Beta1 Timer (UARTx_B1T)

Addresses: UART0_B1T is 4006_A000h base + 24h offset = 4006_A024h
UART1_B1T is 4006_B000h base + 24h offset = 4006_B024h
UART2_B1T is 4006_C000h base + 24h offset = 4006_C024h
UART3_B1T is 4006_D000h base + 24h offset = 4006_D024h
UART4_B1T is 400E_A000h base + 24h offset = 400E_A024h
UART5_B1T is 400E_B000h base + 24h offset = 400E_B024h

Bit	7	6	5	4	3	2	1	0
Read Write	B1T							
Reset	0	0	0	0	0	0	0	0

UARTx_B1T field descriptions

Field	Description
7–0 B1T	Beta1 Timer Beta1 delay is a value that is system dependent and usually does not require adjustment. It is programmed only once and measured in bit times.

52.3.36 UART CEA709.1-B Secondary Delay Timer High (UARTx_SDTH)

Addresses: UART0_SDTH is 4006_A000h base + 25h offset = 4006_A025h
UART1_SDTH is 4006_B000h base + 25h offset = 4006_B025h
UART2_SDTH is 4006_C000h base + 25h offset = 4006_C025h
UART3_SDTH is 4006_D000h base + 25h offset = 4006_D025h
UART4_SDTH is 400E_A000h base + 25h offset = 400E_A025h
UART5_SDTH is 400E_B000h base + 25h offset = 400E_B025h

Bit	7	6	5	4	3	2	1	0
Read Write	SDTH							
Reset	0	0	0	0	0	0	0	0

UARTx_SDTH field descriptions

Field	Description
7–0 SDTH	Secondary Delay Timer High This is the most significant byte of the secondary delay timer and is set by software. This is generally a variable value that must be set for each data message to be transmitted. It is measured in bit times, i.e. the time that it takes for a single bit or one differential Manchester symbol to be transmitted. This value

UARTx_SDTH field descriptions (continued)

Field	Description
	must be between 0 and (BL*Wbase) + (ProritySlots -1), Beta2 timeslots. A value of zero indicates that the queued packet will be sent immediately upon expiration of the beta1 timer.

52.3.37 UART CEA709.1-B Secondary Delay Timer Low (UARTx_SDTL)

Addresses: UART0_SDTL is 4006_A000h base + 26h offset = 4006_A026h
UART1_SDTL is 4006_B000h base + 26h offset = 4006_B026h
UART2_SDTL is 4006_C000h base + 26h offset = 4006_C026h
UART3_SDTL is 4006_D000h base + 26h offset = 4006_D026h
UART4_SDTL is 400E_A000h base + 26h offset = 400E_A026h
UART5_SDTL is 400E_B000h base + 26h offset = 400E_B026h

Bit	7	6	5	4	3	2	1	0
Read Write	SDTL							
Reset	0	0	0	0	0	0	0	0

UARTx_SDTL field descriptions

Field	Description
7–0 SDTL	Secondary Delay Timer Low This is the least significant byte of the secondary delay timer and is set by software. This is generally a variable value that must be set for each data message to be transmitted. It is measured in bit times i.e. the time that it takes for a single bit or one Differential Manchester symbol to be transmitted. This value must be between 0 and (BL*Wbase) + (ProritySlots -1), Beta2 timeslots. A value of zero indicates that the queued packet will be sent immediately upon expiration of the beta1 timer.

52.3.38 UART CEA709.1-B Preamble (UARTx_PRE)

Addresses: UART0_PRE is 4006_A000h base + 27h offset = 4006_A027h
UART1_PRE is 4006_B000h base + 27h offset = 4006_B027h
UART2_PRE is 4006_C000h base + 27h offset = 4006_C027h
UART3_PRE is 4006_D000h base + 27h offset = 4006_D027h
UART4_PRE is 400E_A000h base + 27h offset = 400E_A027h
UART5_PRE is 400E_B000h base + 27h offset = 400E_B027h

Bit	7	6	5	4	3	2	1	0
Read Write	PREAMBLE							
Reset	0	0	0	0	0	0	0	0

UARTx_PRE field descriptions

Field	Description
7–0 PREAMBLE	CEA709.1-B Preamble Register The number of bit-sync characters that occurs prior to the byte-sync character when preamble is transmitted. **NOTE:** The minimum preamble length supported by twisted pair wire is 4-bit sync fields.

52.3.39 UART CEA709.1-B Transmit Packet Length (UARTx_TPL)

Addresses: UART0_TPL is 4006_A000h base + 28h offset = 4006_A028h
UART1_TPL is 4006_B000h base + 28h offset = 4006_B028h
UART2_TPL is 4006_C000h base + 28h offset = 4006_C028h
UART3_TPL is 4006_D000h base + 28h offset = 4006_D028h
UART4_TPL is 400E_A000h base + 28h offset = 400E_A028h
UART5_TPL is 400E_B000h base + 28h offset = 400E_B028h

Bit	7	6	5	4	3	2	1	0
Read Write	TPL							
Reset	0	0	0	0	0	0	0	0

UARTx_TPL field descriptions

Field	Description
7–0 TPL	Transmit Packet Length Register Length of the data packet in bytes that is transmitted by CEA709.1-B transmitter. This includes CRC packet as well.

52.3.40 UART CEA709.1-B Interrupt Enable Register (UARTx_IE)

Addresses: UART0_IE is 4006_A000h base + 29h offset = 4006_A029h
UART1_IE is 4006_B000h base + 29h offset = 4006_B029h
UART2_IE is 4006_C000h base + 29h offset = 4006_C029h
UART3_IE is 4006_D000h base + 29h offset = 4006_D029h
UART4_IE is 400E_A000h base + 29h offset = 400E_A029h
UART5_IE is 400E_B000h base + 29h offset = 400E_B029h

Bit	7	6	5	4	3	2	1	0
Read	0	WBEIE	ISDIE	PRXIE	PTXIE	PCTEIE	PSIE	TXFIE
Write	▨							
Reset	0	0	0	0	0	0	0	0

UARTx_IE field descriptions

Field	Description
7 Reserved	This read-only field is reserved and always has the value zero.
6 WBEIE	Wbase Expired Interrupt Enable Interrupt enable for Wbase expired flag. 0 Interrupt is disabled. 1 Interrupt is enabled.
5 ISDIE	Initial Sync Detection Interrupt Enable Interrupt enable for initial synchronization detection flag. **NOTE:** This bit cannot be cleared except by disabling CEA709. Therefore, the ISDIE bit should be cleared when the first initial sync detection interrupt occurs. If the ISD interrupt is not disabled in the interrupt handler, then user will end up getting interrupts continuously. 0 Interrupt is disabled. 1 Interrupt is enabled.
4 PRXIE	Packet Received Interrupt Enable Interrupt enable for packet received flag. 0 Interrupt is disabled. 1 Interrupt is enabled.
3 PTXIE	Packet Transmitted Interrupt Enable Interrupt enable for packet transmitted flag. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 PCTEIE	Packet Cycle Timer Interrupt Enable Interrupt enable for packet cycle time expired flag. 0 Interrupt is disabled. 1 Interrupt is enabled.
1 PSIE	Preamble Start Interrupt Enable Interrupt enable for preamble start flag. 0 Interrupt is disabled. 1 Interrupt is enabled.
0 TXFIE	Transmission Fail Interrupt Enable Interrupt enable for transmission fail flag. 0 Interrupt is disabled. 1 Interrupt is enabled.

52.3.41 UART CEA709.1-B WBASE (UARTx_WB)

Addresses: UART0_WB is 4006_A000h base + 2Ah offset = 4006_A02Ah
UART1_WB is 4006_B000h base + 2Ah offset = 4006_B02Ah
UART2_WB is 4006_C000h base + 2Ah offset = 4006_C02Ah
UART3_WB is 4006_D000h base + 2Ah offset = 4006_D02Ah
UART4_WB is 400E_A000h base + 2Ah offset = 400E_A02Ah
UART5_WB is 400E_B000h base + 2Ah offset = 400E_B02Ah

Bit	7	6	5	4	3	2	1	0
Read Write	WBASE							
Reset	0	0	0	0	0	0	0	0

UARTx_WB field descriptions

Field	Description
7–0 WBASE	CEA709.1-B WBASE register WBase is the size of the basic randomizing window in bit periods after beta1 time period.

52.3.42 UART CEA709.1-B Status Register (UARTx_S3)

Addresses: UART0_S3 is 4006_A000h base + 2Bh offset = 4006_A02Bh
UART1_S3 is 4006_B000h base + 2Bh offset = 4006_B02Bh
UART2_S3 is 4006_C000h base + 2Bh offset = 4006_C02Bh
UART3_S3 is 4006_D000h base + 2Bh offset = 4006_D02Bh
UART4_S3 is 400E_A000h base + 2Bh offset = 400E_A02Bh
UART5_S3 is 400E_B000h base + 2Bh offset = 400E_B02Bh

Bit	7	6	5	4	3	2	1	0
Read	PEF	WBEF	ISD	PRXF	PTXF	PCTEF	PSF	TXFF
Write			▨					
Reset	0	0	0	0	0	0	0	0

UARTx_S3 field descriptions

Field	Description
7 PEF	Preamble Error Flag This flag indicates that received preamble is in error. If the received preamble length is greater than or less than the transmit preamble length, the preamble error flag is asserted. This flag is cleared by writing `1'.

Table continues on the next page...

UARTx_S3 field descriptions (continued)

Field	Description
	0 Preamble is correct. 1 Preamble is in error.
6 WBEF	Wbase Expired Flag This flag indicates that Wbase time period has been expired after Beta1 time slots. This flag is cleared by writing `1'. 0 Wbase time period is not expired. 1 Wbase time period has been expired after beta1 time slots.
5 ISD	Initial Sync Detect This flag indicates that initially a valid one and a line code violation is detected. This flag is cleared by deasserting EN709 bit. 0 Initial sync is not detected. 1 Initial sync is detected.
4 PRXF	Packet Received Flag This flag indicates that complete packet is received. This flag is cleared by writing `1'. 0 Packet is not received. 1 Packet is received.
3 PTXF	Packet Transmitted Flag This flag indicates that complete packet is transmitted. This flag is cleared by writing `1'. In case TX packet gets aborted due to fifo becoming empty or overflow, packet transmitted flag will still be generated. 0 Packet transmission is not complete. 1 Packet transmission is complete.
2 PCTEF	Packet Cycle Timer Expired Flag This flag indicates that packet cycle time period has been expired with no activity on the line. This flag is cleared by writing `1'. 0 Packet Cycle Time is not expired. 1 Packet cycle time is expired.
1 PSF	Preamble Start Flag This flag indicates start of preamble while the packet is being transmitted. This flag is cleared by writing `1'. 0 Preamble start is not detected. 1 Preamble start is detected.
0 TXFF	Transmission Fail Flag This flag indicates that transmission could not proceed. This flag is asserted when the packet is queued for transmission but before the random delay is expired an incoming receive packet is detected. This flag is also asserted while transmission when the TX fifo becomes empty or overflows. During these cases Line Code Violation is transmitted on TX line immediately after current byte or preamble transmission is finished, without waiting for completion of transmit packet length. If the transmission fail flag is asserted then TX709 bit of UART_C6 register is cleared. This flag is cleared by writing `1'.

Table continues on the next page...

UARTx_S3 field descriptions (continued)

Field	Description
	0 Transmission continues normally. 1 Transmission is failed.

52.3.43 UART CEA709.1-B Status Register (UARTx_S4)

Addresses: UART0_S4 is 4006_A000h base + 2Ch offset = 4006_A02Ch
UART1_S4 is 4006_B000h base + 2Ch offset = 4006_B02Ch
UART2_S4 is 4006_C000h base + 2Ch offset = 4006_C02Ch
UART3_S4 is 4006_D000h base + 2Ch offset = 4006_D02Ch
UART4_S4 is 400E_A000h base + 2Ch offset = 400E_A02Ch
UART5_S4 is 400E_B000h base + 2Ch offset = 400E_B02Ch

Bit	7	6	5	4	3	2	1	0
Read	0			INITF	CDET		ILCV	FE
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_S4 field descriptions

Field	Description
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 INITF	Initial Synchronization Fail Flag This bit indicates that initial synchronization is failed and the packet cycle time is expired after enabling EN709 register. This flag is cleared if EN709 is cleared. 0 Initial synchronization is not failed. 1 Initial synchronization is failed.
3–2 CDET	CDET These bits indicates when the collision occurs during Transmission. This flag is cleared by writing 2'b11. If collision flag is not cleared by software and valid collision pulse is detected during some other phase of transmission, then collision flag will continue to indicate the previous value. 00 No collision. 01 Collision occurred during preamble. 10 Collision occurred during data. 11 Collision occurred during line code violation.
1 ILCV	Improper Line Code Violation This flag indicates that Line Code violation received is not proper. This flag is cleared by writing `1'. 0 Line code violation received is proper. 1 Line code violation received is improper i.e less than 3-bit periods.

Table continues on the next page...

UARTx_S4 field descriptions (continued)

Field	Description
0 FE	Framing Error This flag indicates that received CEA709.1-B packet is finish at byte boundary. This flag is cleared by writing `1'. 0 Received packet is byte bound. 1 Received packet is not byte bound.

52.3.44 UART CEA709.1-B Received Packet Length (UARTx_RPL)

Addresses: UART0_RPL is 4006_A000h base + 2Dh offset = 4006_A02Dh
UART1_RPL is 4006_B000h base + 2Dh offset = 4006_B02Dh
UART2_RPL is 4006_C000h base + 2Dh offset = 4006_C02Dh
UART3_RPL is 4006_D000h base + 2Dh offset = 4006_D02Dh
UART4_RPL is 400E_A000h base + 2Dh offset = 400E_A02Dh
UART5_RPL is 400E_B000h base + 2Dh offset = 400E_B02Dh

Bit	7	6	5	4	3	2	1	0
Read	RPL							
Write	▨							
Reset	0	0	0	0	0	0	0	0

UARTx_RPL field descriptions

Field	Description
7–0 RPL	Received packet length This register field indicates the length of received packet in bytes. If the received packet is not byte aligned then the partial byte received is appended by zeros.

52.3.45 UART CEA709.1-B Received Preamble Length (UARTx_RPREL)

Addresses: UART0_RPREL is 4006_A000h base + 2Eh offset = 4006_A02Eh
UART1_RPREL is 4006_B000h base + 2Eh offset = 4006_B02Eh
UART2_RPREL is 4006_C000h base + 2Eh offset = 4006_C02Eh
UART3_RPREL is 4006_D000h base + 2Eh offset = 4006_D02Eh
UART4_RPREL is 400E_A000h base + 2Eh offset = 400E_A02Eh
UART5_RPREL is 400E_B000h base + 2Eh offset = 400E_B02Eh

Bit	7	6	5	4	3	2	1	0
Read	RPREL							
Write								
Reset	0	0	0	0	0	0	0	0

UARTx_RPREL field descriptions

Field	Description
7–0 RPREL	Received preamble length This register field indicates the number of bit sync fields received in the preamble.

52.3.46 UART CEA709.1-B Collision Pulse Width (UARTx_CPW)

Addresses: UART0_CPW is 4006_A000h base + 2Fh offset = 4006_A02Fh
UART1_CPW is 4006_B000h base + 2Fh offset = 4006_B02Fh
UART2_CPW is 4006_C000h base + 2Fh offset = 4006_C02Fh
UART3_CPW is 4006_D000h base + 2Fh offset = 4006_D02Fh
UART4_CPW is 400E_A000h base + 2Fh offset = 400E_A02Fh
UART5_CPW is 400E_B000h base + 2Fh offset = 400E_B02Fh

Bit	7	6	5	4	3	2	1	0
Read Write	CPW							
Reset	0	0	0	0	0	0	0	0

UARTx_CPW field descriptions

Field	Description
7–0 CPW	CEA709.1-B CPW register This register indicates the width of valid collision pulse in terms of IPG clock cycles.

52.3.47 UART CEA709.1-B Receive Indeterminate Time (UARTx_RIDT)

Addresses: UART0_RIDT is 4006_A000h base + 30h offset = 4006_A030h
UART1_RIDT is 4006_B000h base + 30h offset = 4006_B030h
UART2_RIDT is 4006_C000h base + 30h offset = 4006_C030h
UART3_RIDT is 4006_D000h base + 30h offset = 4006_D030h
UART4_RIDT is 400E_A000h base + 30h offset = 400E_A030h
UART5_RIDT is 400E_B000h base + 30h offset = 400E_B030h

Bit	7	6	5	4	3	2	1	0
Read Write	RIDT							
Reset	0	0	0	0	0	0	0	0

UARTx_RIDT field descriptions

Field	Description
7–0 RIDT	CEA709.1-B Receive IDT Register This register indicates the indeterminate time period after reception during which any activity on RX line will be discarded. Indeterminate time period value should be less than Beta1 timer value.

52.3.48 UART CEA709.1-B Transmit Indeterminate Time (UARTx_TIDT)

Addresses: UART0_TIDT is 4006_A000h base + 31h offset = 4006_A031h
UART1_TIDT is 4006_B000h base + 31h offset = 4006_B031h
UART2_TIDT is 4006_C000h base + 31h offset = 4006_C031h
UART3_TIDT is 4006_D000h base + 31h offset = 4006_D031h
UART4_TIDT is 400E_A000h base + 31h offset = 400E_A031h
UART5_TIDT is 400E_B000h base + 31h offset = 400E_B031h

Bit	7	6	5	4	3	2	1	0
Read Write	TIDT							
Reset	0	0	0	0	0	0	0	0

UARTx_TIDT field descriptions

Field	Description
7–0 TIDT	CEA709.1-B Transmit IDT Register This register indicates the indeterminate time period after transmission during which any activity on RX line will be discarded. Indeterminate time period value should be less than Beta1 timer value.

UARTx_TIDT field descriptions (continued)

Field	Description

52.4 Functional description

This section provides a complete functional description of the UART block.

The UART allows full duplex, asynchronous, NRZ serial communication between the CPU and remote devices, including other CPUs. The UART transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the UART, writes the data to be transmitted, and processes received data.

52.4.1 CEA709.1-B

The UART provides support for CEA709.1-B, which is commonly used in building automation home networking, including all key building automation subsystems such as heating, ventilating, air-conditioning, lighting, security, fire detection, access control, and energy monitoring.

52.4.1.1 CEA709.1-B packet cycle

The following figure illustrates the frame format and differential Manchester encoding. Differential Manchester encoding requires that each transmitted bit includes a clock transition at the start of the bit period. This allows synchronization with the receiver.

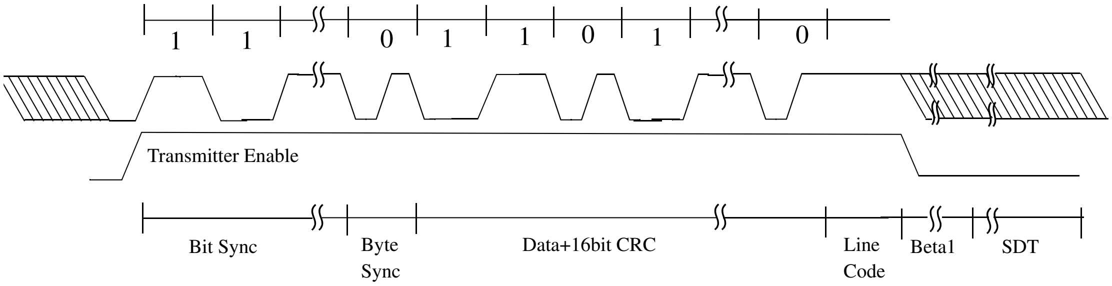


Figure 52-337. Frame format with differential manchester encoding

A logic zero is indicated with the presence of a transition in the middle of the bit period and a logic one is indicated by the absence of any transition. When transitions occur at the start of the bit time, polarity is arbitrary because the last bit of a transmission has no trailing clock edge. A transmitter will transmit a preamble at the beginning of a packet to allow other nodes to synchronize their receiver clocks. The preamble comprises a bit-sync field followed by a byte-sync field. The bit-sync field is a series of differential Manchester logic ones and the byte-sync field is a single differential Manchester logic zero. The byte-sync field marks the end of the preamble and the start of the data field (MPDU/LPDU).

The transmitter terminates the packet by forcing the data output to be transitionless long enough for the receiver to recognize an invalid bit code. This signals the end of the packet. At the end of the packet transmission, the line must remain transitionless for three bit periods after the final clock transition.

The UART is responsible for providing the BitSync and ByteSync fields of the PPDU illustrated below. The layer two software manages all other encapsulating fields and provides these to the UART as part of the packet to be transmitted.

Bit Sync	Byte Sync	Priority	Alt Path	Delta BL	NPDU	CRC

Figure 52-338. Physical protocol data unit structure

52.4.1.2 Packet cycle and delay calculations

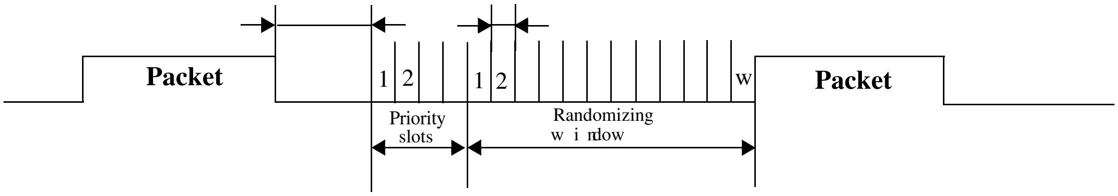


Figure 52-339. CEA709.1-B packet cycle

Predictive p-persistent CSMA is a technique for collision avoidance that randomizes channel access using knowledge of predicted load. It manages software using data and events reported by the hardware.

Beta1 delay is a value set by the software. It is generally a fixed value that is system dependent and hence does not usually require adjustment. It is measured in bit times, that is, the time that it takes for a single bit to be transmitted or one differential Manchester symbol. Beta1 is defined by CEA/EIA-709 specification as:

Beta1 > 1 bit time + $(2 \times T_{aup} + T_{aum})$

Where T_{aup} is the physical propagation delay defined by the media length.

T_{aum} is the detection and turn around-delay within the MAC sublayer; this is the period from the time the idle channel condition is detected, to the point when the first output transition appears on the output. On media where there is a carrier, this time must include the time between turning on the carrier, and it being asserted as a valid carrier on the medium.

The secondary delay timer is a value that is set by software. This is generally a variable value that must be set for each data message to be transmitted. It is measured in bit times, that is, the time that it takes for a single bit to be transmitted or one differential Manchester symbol. This value must be between 0 and (BL × Wbase) + (ProritySlots -1), Beta2 timeslots. A value of zero indicates that the queued packet for transmit is to be sent immediately upon expiration of the beta1 timer. According to the CEA/EIA-709 specification:

- BL is back log
- Wbase is 16 beta2 timeslots
- A priorityslot is the same amount of time as a beta2 timeslot
- Beta2 > 2 × T_{aup} + T_{aum}

Priority slots are handled completely by software. When calculating the secondary delay timer value, the software must take into account any priority slot that is included in the design of the system.

Each node must maintain an estimation of the current channel backlog. Backlog calculation is managed by the layer two software. Initially, the backlog is set to one. The backlog is incremented on transmission by a value indicated in the frames backlog increment field.

The backlog decrements under the following conditions:

- On waiting to transmit: If Wbase randomizing slots go by without channel activity.
- On receive: If a packet is received with a backlog increment of 0.
- On transmit: If a packet is transmitted with a backlog increment of 0.
- On idle: If a packet cycle time expires without channel activity.

The following actions need to be completed when a frame is received to prepare an outgoing message for transmission after the channel becomes idle:

- CRC of incoming message needs to be verified by software.
- If the CRC is good, the BL is recalculated, otherwise BL remains the same.
- Transmit delay (secondary delay timer) is calculated and supplied to UART.

52.4.1.3 Clock resynchronization

The UART is transmitting on time base source. Hence, all receivers keep synchronization with the node that is transmitting and no clock resynchronization occurs in transmitting.

When the UART is receiving or waiting for the opportunity to transmit, clock resynchronization is vital. Since long streams of data are possible (up to 229 bytes + headers), there exists significant potential for nodes to wander regarding time reference over the course of the message. Thus, differential Manchester encoding (DME) is used. While DME requires twice the bandwidth of non-return to zero (NRZ) encoding schemes, it has the benefit of a guaranteed transition at the start of each bit transmitter with a transition occurring at the middle of the bit to encode a logic 0 or the lack of a transition at the middle of the bit time to encode a logic 1. By detecting the transition at the start of a bit period, the receiver is able to be resynthesized to the transmitter every bit period. Resynchronization can only occur after the node is already synchronized with the system. Additionally, for resynchronization to be effective, some basic assumptions regarding the system must be made:

1. Only a single channel sample may be in error (noise) over the entire bit (16 samples) period. 2.
2. While a node is drifted from the system time base, with the resynchronization, the node will never be shifted by more than 2 data samples in a given bit period.
3. If multiple noise events have occurred, no action is taken. 4.
4. If a single noise event occurred and it could be possible to uniquely identify the noise event then resynchronization takes place.

Starting at sample 15 (of the previous time bit period) five data samples are collected. The number, and location of the samples are key to decide if an adjustment in time base is required. Table below lists the possible values and the actions associated with each possibility. In the table, S means the data is the same as the logical value that was received in the second half of the previous bit period while D means that the sample is different from the logical value that was received in the second half of the previous bit period.

Sample Values (15,16,1,2,3)	Action / Event
SSSSS	No start of bit tarnation has been detected. Hence no adjustment to time base is made.
SSSSD	Either two (or more) error events occurred or the time base were off. In this case, the time base is slowed down by two. Sample 3 becomes sample 1. The next sample is treated as sample 2.
SSSDS	Either two (or more) error events occurred, time base were off along with noise occurrence or sample 2 is noise and there is no start of bit transition. Hence no adjustment to time base is made.

Table continues on the next page...

Sample Values (15,16,1,2,3)	Action / Event
SSSDD	It is possible that either noise was received during sample 1 or the time base needs shifting. In this case the time base is slowed down by one. Sample 2 becomes sample 1, sample 3 becomes sample 2. The next sample is treated as sample 3.
SSDSS	It is most likely that sample 1 is noise and there is no start of bit transition. Hence no adjustment to time base is made.
SSDSD	It is possible that sample 1 is noise (and time base needs shifting by 2) or that sample 2 is noise. It is more likely that sample 2 is noise and hence no adjustment to time base is made.
SSDDS	It is most likely that sample 3 is noise, hence no adjustment to time base is made.
SSDDD	This is the expected case. Hence no adjustment to time base is made.
SDSSS	It is most likely that sample 16 is noise and there is no start of bit transition, hence no adjustment to time base is made.
SDSSD	Either multiple errors occurred or sample 16 is noise (and time base is off by 2). In this case, the time base is slowed down by two. Sample 3 becomes sample 1. The next sample is treated as sample 2.
SDSDS	In this case multiple errors have occurred. Hence no adjustment to time base is made.
SDSDD	In this case, there must either be multiple noise or one noise (at sample 16 or sample 1) with a time shift. Assuming that one noise occurred, it unclear what direction the time shift is. Hence no adjustment to time base is made.
SDDSS	In this case either multiple errors occurred, two (or more) noise or two (or more) noise and a time shift. The most likely case is that samples 16 and 1 are noise. Hence no adjustment to time base is made.
SDDSD	The most likely case is noise for sample 2 and a time shift. Hence the time base is sped up by one. Sample 16 becomes sample 1, sample 1 becomes sample 2, sample 2 becomes sample 3, sample 3 becomes sample 4 and the next sample taken is sample 5.
SDDDS	The most likely case is noise for sample 3 and a time shift. Hence the time base is sped up by one. Sample 16 becomes sample 1, sample 1 becomes sample 2, sample 2 becomes sample 3, sample 3 becomes sample 4 and the next sample taken is sample 5.
SDDDD	Either sample 16 is noise or the time base has shifted. In this case it is assumed that a time shift has occurred. Hence the time base is sped up by one. Sample 15 becomes sample 1, sample 1 becomes sample 2, sample 2 becomes sample 3, sample 3 becomes sample 4 and the next sample taken is sample 5.
DSSSS	It is most likely that sample 16 is noise and there is no start of bit transition, hence no adjustment to time base is made.

Table continues on the next page...

Sample Values (15,16,1,2,3)	Action / Event
DSSSD	It is most likely that sample 15 is noise along with time shift. In this case, the time base is slowed down by two. Sample 3 becomes sample 1. The next sample is treated as sample 2.
DSSDS	In this case multiple errors occurred. Hence no adjustment to time base is made
DSSDD	Either multiple errors occurred (possibly with time shift) or more likely sample 15 is noise. In this case the time base is slowed down by one. Sample 2 becomes sample 1, sample 3 becomes sample 2. The next sample is treated as sample 3.
DSDSS	In this case multiple errors occurred. Hence no adjustment to time base is made.
DSDSD	In this case multiple errors occurred. Hence no adjustment to time base is made.
DSDDS	In this case multiple errors occurred. Hence no adjustment to time base is made.
DSDDD	In this case either multiple errors occurred or sample 15 is noise and there is no start of bit transition. Hence no adjustment to time base is made.
DDSSS	In this case multiple errors occurred. It is most likely that samples 15 and 16 are noise. Hence no adjustment to time base is made.
DDSSD	In this case multiple errors occurred. Hence no adjustment to time base is made.
DDSDS	In this case multiple errors occurred. Hence no adjustment to time base is made.
DDSDD	It is most likely that sample 1 is noise. Hence the time base is sped up by two. Sample 15 becomes sample 1, sample 16 becomes sample 2, sample 1 becomes sample 3, sample 2 becomes sample 4, sample 3 becomes sample 5 and the next sample taken is sample 6.
DDDSS	In this case multiple errors occurred along with time shift. Hence no adjustment to time base is made.
DDDSD	It is most likely that sample 2 is noise. Hence the time base is sped up by two. Sample 15 Becomes sample 1, sample 16 becomes sample 2, sample 1 becomes sample 3, sample 2 becomes sample 4, sample 3 becomes sample 5 and the next sample taken is sample 6.
DDDDS	It is most likely that sample 3 is noise. Hence the time base is sped up by two. Sample 15 becomes sample 1, sample 16 becomes sample 2, sample 1 becomes sample 3, sample 2 becomes sample 4, sample 3 becomes sample 5 and the next sample taken is sample 6.
DDDDD	Either samples 15 and 16 are noise or more likely the time base has shifted. Hence the time base is sped up by two. Sample 15 becomes sample 1, sample 16 becomes sample 2, sample 1 becomes sample 3, sample 2 becomes sample 4, sample 3 becomes sample 5 and the next sample taken is sample 6.

52.4.1.4 Data sampling

The receiver samples the unsynchronized receiver input signal at the RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock is re-synchronized after every bit.

To locate the start of preamble, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s or logic 1 preceded by three logic 0s. When the falling edge or rising edge of a possible preamble bit occurs, the RT clock begins to count to 16.

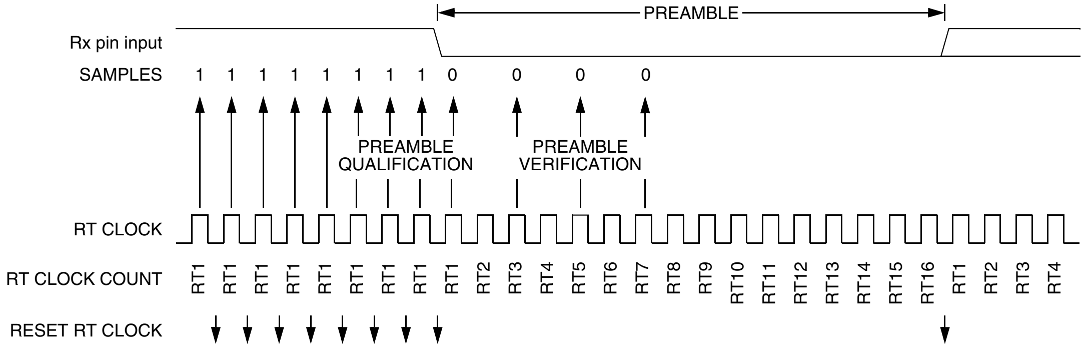


Figure 52-340. Receiver data sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. The following table summarizes the results of the preamble verification samples.

Table 52-347. Preamble/ Data bit verification

RT3, RT5, and RT7 samples	Preamble verification
000	Yes
001	Yes
010	Yes
011	No
100	Yes
101	No
110	No
111	No

If preamble verification is not successful, the RT clock is reset and a new search for a preamble begins.

To determine the value of a data bit, recovery logic takes samples at RT11, RT12, and RT13. The following table summarizes the results of the data bit samples. If the majority of RT11, RT12 and RT13 samples is same as majority of RT3, RT5 and RT7 samples then that bit detected is 1 else the data bit detected is 0.

Table 52-348. Data bit recovery

RT11, RT12, and RT13 samples	Data bit determination
000	0
001	0
010	0
011	1
100	0
101	1
110	1
111	1

To signify the end of a data packet, the transmitter causes a line-code violation to occur i.e the transmitter remains transitionless for at least 3-bit periods after the final clock transition, excluding the final data transition (if it exists). The receiver detects this violation. For the purpose of detecting a line-code violation, the receiver will monitor the channel to locate a series of 5 or 6 back-to-back half bit periods.

52.4.1.5 Initial clock synchronization

When operating with EN709 set, there are various times when initial clock synchronization is required. When the UART has just been enabled, there is clearly no system clock reference. Additionally, if a channel has remained idle for a significant period of time (such as the arbitration time between packets) substantial clock drift may have occurred in the system between nodes since there have been meaningful clock transitions on the channel to keep nodes synchronized. As such, after these events, the clock may require significant synchronization adjustment; this event is referred to as initial clock synchronization.

There are three situations that may occur when a node attempts to obtain initial clock synchronization.

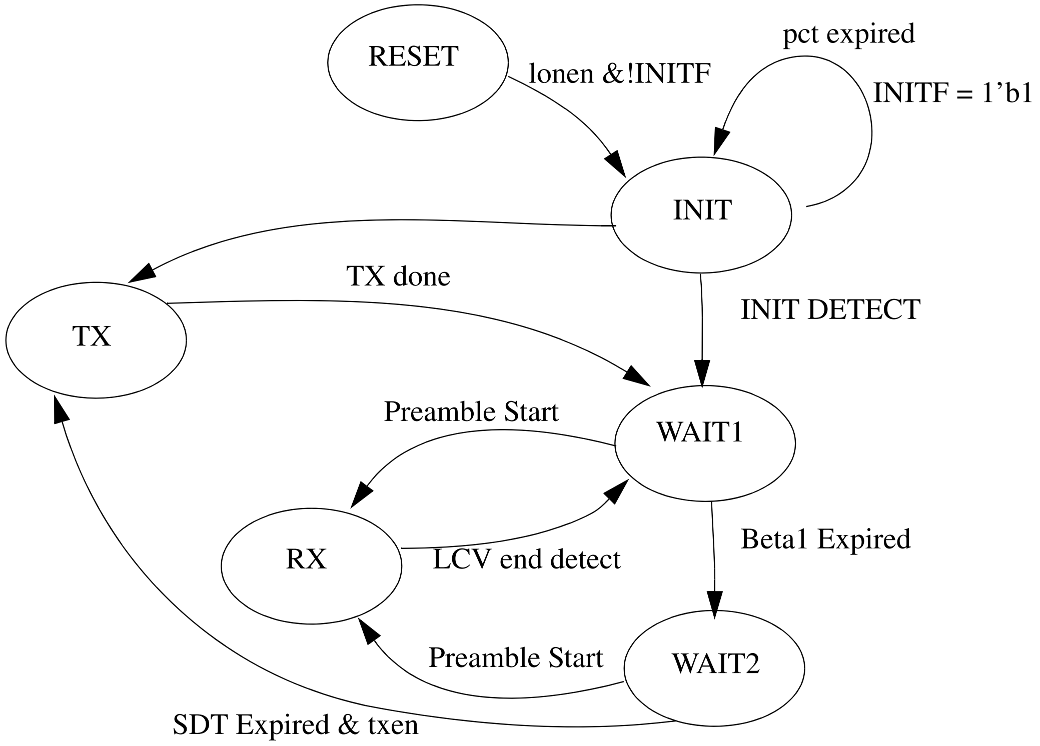
1. The node enters the system while a data packet is being actively transmitted. B.
2. The node enters the system while there is no data packet being actively transmitted on the system. C.
3. The node is already in the system and initial clock synchronization is required due to the end of a packet occurring.

For case 1 and 2, it will implement the following procedure:

1. The UART attempts to identify a valid edge to synchronize with. 2.
2. While the UART attempts to locate a valid edge, it will also seek to identify a line-code violation of 8 back-to-back half bit time samples rather than the 6. It is not required to finish the current bit because the clock has not be synchronized. If the required line-code violation is detected, the beta1 delay timer will start and the UART will transit to case 3.
3. If an edge is determined to be valid, that node will consider itself synchronized but will not start receiving (or attempt to send data) until a line-code violation has been identified. 4.
4. If no valid edge is determined and meanwhile the packet cycle timer is expired, it is indicated to processor that initial synchronization is failed and the processor can choose to transmit the data.

For case 3, it will implement the following procedure:

1. Beta1 delay and secondary delay times increment as appropriate. (i.e. beta1 delay expires prior to the secondary delay timer starting).
2. While the timers are counting, the UART attempts to identify a valid edge.
3. If a valid edge is identified prior to the times expiring and data was queued to be transmitted, the transmission failure will assert and the clock will be considered synchronized. The incoming data packet will be received.
4. If a valid edge is NOT identified prior to the delay times expiring, and data is queued to be transmitted, the UART considers itself synchronized and start the preamble process.
5. If a valid edge is NOT identified prior to the delay time expiring, and data is not queued to be transmitted, the UART continues attempting to locate a valid edge using the same process and receives the incoming data packet like in step 3.



52.4.1.6 Priority packet pre-emption

The first data is fetched from the data buffer immediately after the preamble has completed. Hence, it is possible to decide which data is sent during transmission up until the completion of the preamble. This can be done in two different ways.

- The expected data to be transmitted can be written in to the data buffer prior to or shortly after the TE bit is enabled. In this case, the data will be ready prior to the start of the preamble period. If a high priority packet has been identified for immediate / preemptive transmission, software may flush the data buffer and put the new data into the data buffer. This new data must be put into the data buffer prior to the completion of the preamble. Likewise the transmit packet length register needs to be updated.
- Software can trigger data to be transmitted by asserting TE before the actual data has been placed in the data buffer. At the last possible moment software can write data into the data buffer and update the transmit packet length register. This occurs prior to the preamble completing. To assist in identifying how much time is left before the preamble completes, the preamble started interrupt is asserted when the UART starts transmitting the preamble.

NOTE

If the data buffer does not contain at least one byte of valid data and the transmit packet length register has been updated prior to the preamble completing, an underflow event will occur and TXEN is de-asserted. The packet is terminated by transmitting line code violation.

52.4.1.7 Collision detection

Collision flag is detected only when device is transmitting if UARTx_C6[CE] bit is asserted. The collision pulse is valid if it is asserted for CPW number of ipg clock cycles. If the collision signal is already asserted before the start of packet transmission then width of the collision pulse is calculated from the start of transmit packet as shown in figure below. If the collision signal is not cleared by the software by writing 11b, then flag will continue to retain previous value. Once the flag is cleared collision pulse width is again calculated and the flag is asserted if width is equal to or more than programmed CPW value.

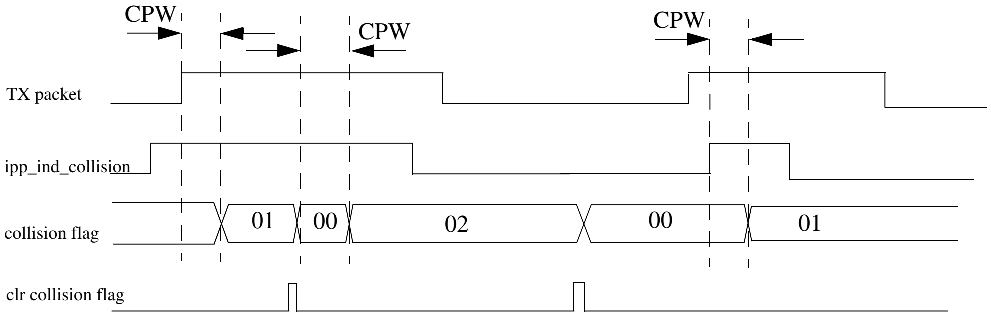


Figure 52-341. Collision pulse detection

The collision signal is asynchronous to the ipg clk, hence the collision pulse of width exactly equal to CPW might not be detected correctly due to synchronization issue. Collision pulse visible to design might get decreased by one ipg clk cycle due to asynchronous nature of collision pulse.

52.4.2 Transmitter

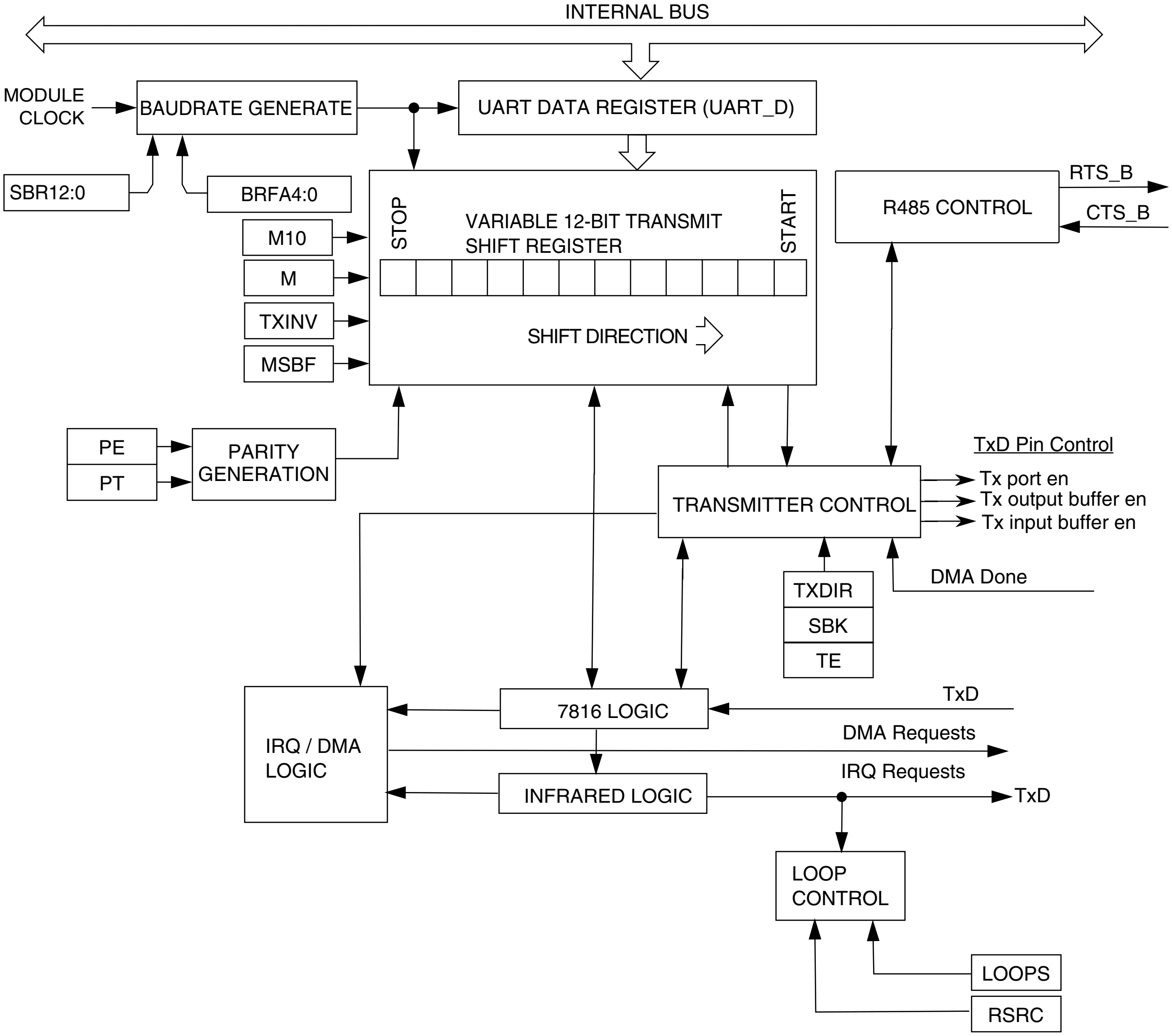


Figure 52-342. Transmitter Block Diagram

52.4.2.1 Transmitter character length

The UART transmitter can accommodate either 8, 9, or 10-bit data characters. The state of the C1[M] and C1[PE] bits and the C4[M10] bit determine the length of data characters. When transmitting 9-bit data, bit C3[T8] is the ninth bit (bit 8).

52.4.2.2 Transmission bit order

When the S2[MSBF] bit is set, the UART automatically transmits the MSB of the data word as the first bit after the start bit. Likewise the LSB of the data word is transmitted immediately preceding the parity bit (or the stop bit if parity is not enabled). All necessary bit ordering is handled automatically by the module hence the format of the data written to the D register for transmission is completely independent of the S2[MSBF] setting.

52.4.2.3 Character transmission

To transmit data, the MCU writes the data bits to the UART transmit buffer using UART data registers (C3[T8]/D). Data in the transmit buffer is then in turn transferred to the transmitter shift register as needed. The transmit shift register then shifts a frame out through the transmit data output signal after it has prefaced it with any required start and stop bits. The UART data registers (C3[T8] and D) provide access to the transmit buffer structure.

The UART also sets a flag, the transmit data register empty flag (S1[TDRE]) and generates interrupt or DMA request (C5[TDMAS]), whenever the number of datawords in the transmit buffer is equal to or less than the value indicated by the TWFIFO[TXWATER]. The transmit driver routine may respond to this flag by writing additional datawords to the transmit buffer using (C3[T8]/D) as space permits.

See Application information for specific programing sequences.

Setting the C2[TE] bit automatically loads the transmit shift register with a preamble of 10 logic 1s (if C1[M] = 0), 11 logic 1s (if C1[M] = 1 and C4[M10] = 0), or 12 logic 1s (if C1[M] = 1, C4[M10] = 1, C1[PE] = 1). After the preamble shifts out, control logic transfers the data from the UART data register into the transmit shift register. The transmitter automatically transmits the correct start bit and stop bit before and after the dataword.

When C7816[ISO_7816E] = 1 setting the C2[TE] bit does not result in a preamble being generated. The transmitter starts transmitting as soon as the corresponding guard time expires. When C7816[TTYPE] = 0 the value in GT is used, when C7816[TTYPE] = 1 the value BGT is used since it is assumed that the C2[TE] will remain asserted until the end of the block transfer. The C2[TE] bit is automatically cleared when in C7816[TTYPE] = 1 and the block being transmitted has been completed. When C7816[TTYPE] = 0, the transmitter listens for a NACK indication. If no NACK is received it is assumed that character was correctly received. If a NACK is received the transmitter will resend the data, assuming that the number of retries for that character (number of NACKs received) is less than or equal to the value in ET7816[TXTHRESHOLD].

Hardware supports odd or even parity. When parity is enabled, the bit immediately preceding the stop bit is the parity bit.

When the transmit shift register is not transmitting a frame, the transmit data output signal goes to the idle condition, logic 1. If at any time software clears the C2[TE] bit, the transmitter enable signal goes low and the transmit signal goes idle.

If software clears C2[TE] while a transmission is in progress, the character in the transmit shift register continues to shift out, provided S1[TC] flag was cleared during the data write sequence. To clear the S1[TC] flag, the S1 register must be read followed by a write to UARTx_D register.

If the S1[TC] flag is cleared during character transmission and the C2[TE] bit is cleared, the transmission enable signal is deasserted at the completion of current frame. Following this, the transmit data out signal enters the idle state even if there is data pending in the UART transmit data buffer. To ensure that all the data written in the FIFO is transmitted on the link before clearing C2[TE], wait for the S1[TC] flag to set. Alternatively, the same can be achieved by setting TWFIFO[TXWATER] to 0x0 and waiting for S1[TDRE] to set.

52.4.2.4 Transmitting break characters

Setting the C2[SBK] loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the C1[M] and C1[PE] bits, the S2[BRK13] bit, and the C4[M10] bit. Refer to the following table.

Table 52-349. Transmit break character length

S2[BRK13]	C1[M]	C4[M10]	C1[PE]	Bits transmitted
0	0	—	—	10
0	1	0	—	11
0	1	1	0	11
0	1	1	1	12
1	0	—	—	13
1	1	—	—	14

As long as C2[SBK] is set, transmitter logic continuously loads break characters into the transmit shift register. After software clears the C2[SBK] bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next character. Break bits are not supported when C7816[ISO_7816E] is set/enabled.

NOTE

When queuing a break character, it will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that if data is queued in the data buffer to be transmitted, the break character will preempt that queued data. The queued data will then be transmitted after the break character is complete.

52.4.2.5 Idle characters

An idle character contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the C1[M] and C1[PE] bits and the C4[M10] bit. The preamble is a synchronizing idle character that begins the first transmission initiated after setting the C2[TE] bit. When C7816[ISO_7816E] is set/enabled, idle characters are not sent or detected. When data is not being transmitted the data I/O line is in an inactive state.

If the C2[TE] bit is cleared during a transmission, the transmit data output signal becomes idle after completion of the transmission in progress. Clearing and then setting the C2[TE] bit during a transmission queues an idle character to be sent after the dataword currently being transmitted.

Note

When queuing an idle character the idle character will be transmitted following the completion of the data value currently being shifted out from the shift register. This means that if data is queued in the data buffer to be transmitted, the idle character will preempt that queued data. The queued data will then be transmitted after the idle character is complete.

If the C2[TE] bit is cleared and the transmission is completed, the UART is not the master of the TXD pin.

52.4.2.6 Hardware flow control

The transmitter supports hardware flow control by gating the transmission with the value of $\overline{\text{CTS}}$. If the clear-to-send operation is enabled, the character is transmitted when $\overline{\text{CTS}}$ is asserted. If $\overline{\text{CTS}}$ is deasserted in the middle of a transmission with characters remaining in the receiver data buffer, the character in the shift register is sent and TXD remains in the mark state until $\overline{\text{CTS}}$ is reasserted.

If the clear-to-send operation is disabled, the transmitter ignores the state of $\overline{\text{CTS}}$. Also, if the transmitter is forced to send a continuous low condition because it is sending a break character, the transmitter ignores the state of $\overline{\text{CTS}}$ regardless of whether the clear-to-send operation is enabled.

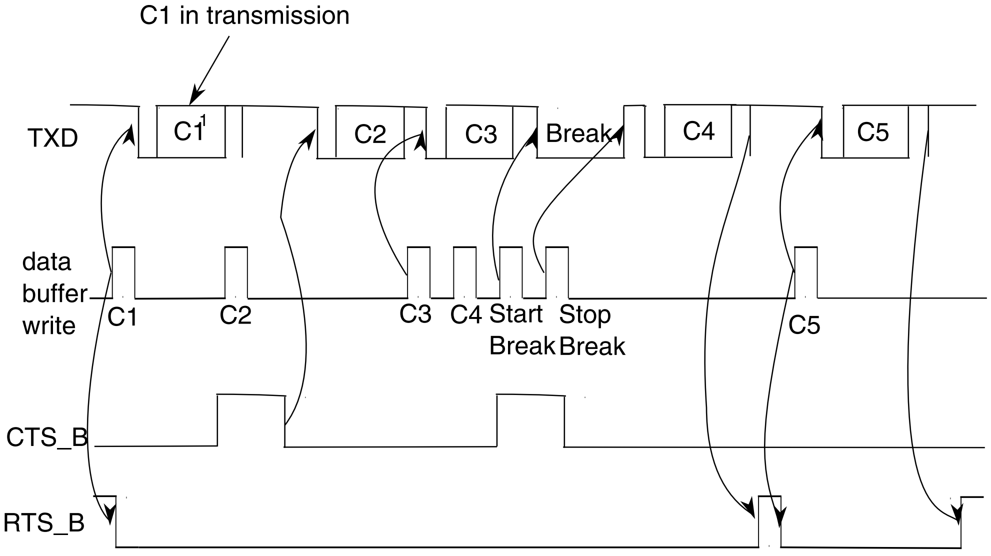
The transmitter's $\overline{\text{CTS}}$ signal can also be enabled even if the same UART receiver's $\overline{\text{RTS}}$ signal is disabled.

52.4.2.7 Transceiver driver enable

The transmitter can use $\overline{\text{RTS}}$ as an enable signal for the driver of an external transceiver, see Transceiver driver enable using $\overline{\text{RTS}}$ for details. If the request-to-send operation is enabled, when a character is placed into an empty transmitter data buffer, $\overline{\text{RTS}}$ asserts one bit time before the start bit is transmitted. $\overline{\text{RTS}}$ remains asserted for the whole time that the transmitter data buffer has any characters. $\overline{\text{RTS}}$ 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 $\overline{\text{RTS}}$, with the same assertion and deassertion timing as having a character in the transmitter data buffer.

The transmitter's $\overline{\text{RTS}}$ signal only asserts when the transmitter is enabled. However, the transmitter's $\overline{\text{RTS}}$ signal is unaffected by its $\overline{\text{CTS}}$ signal. $\overline{\text{RTS}}$ will remain asserted until the transfer is completed, even if the transmitter is disabled mid-way through a data transfer.

The following figure shows the functional timing information for the transmitter. Along with the actual character itself, TXD shows the start bit. The stop bit is also indicated, with a dashed line if necessary.



1. Cn = transmit characters

Figure 52-343. Transmitter RTS and CTS timing diagram

52.4.3 Receiver

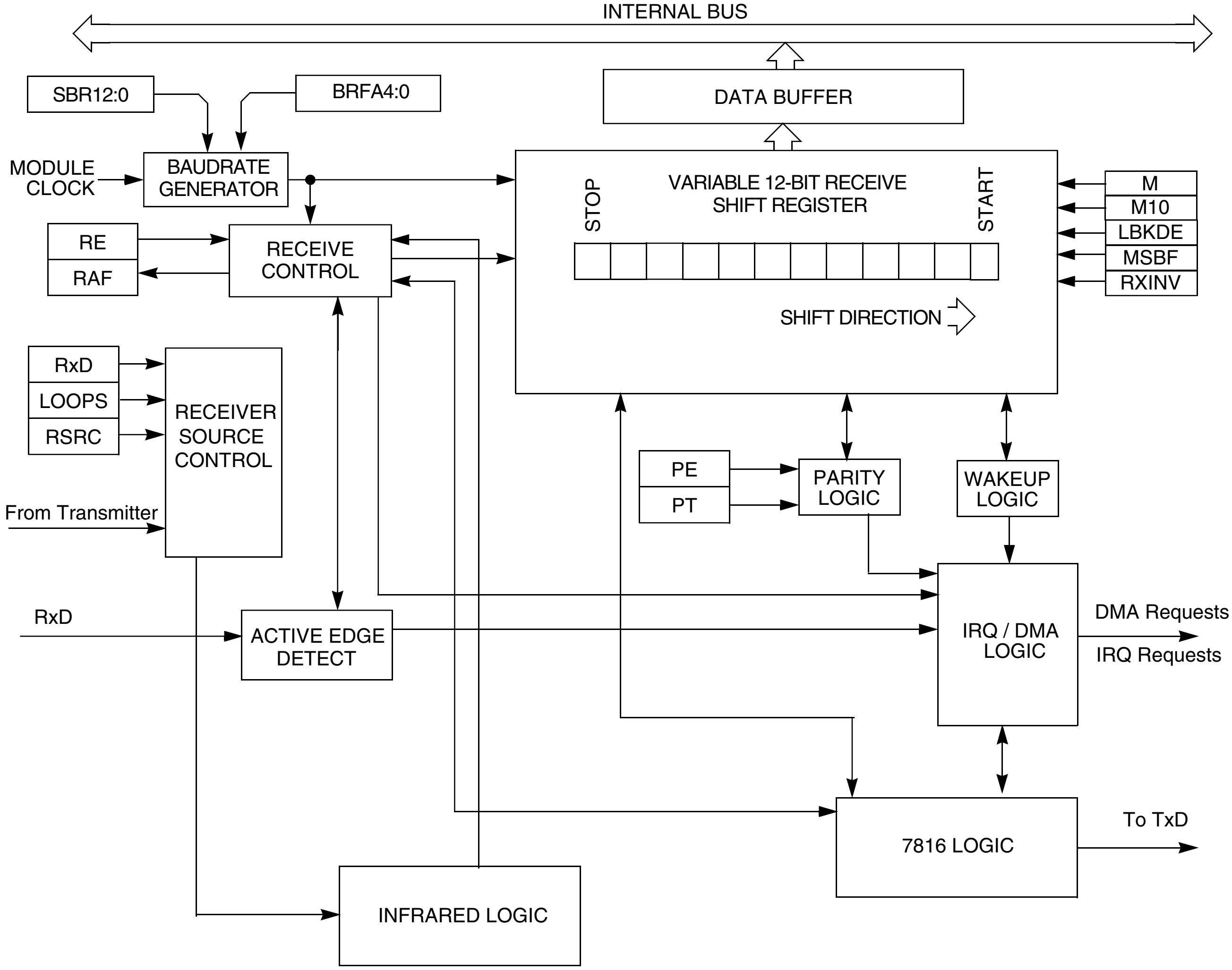


Figure 52-344. UART receiver block diagram

52.4.3.1 Receiver character length

The UART receiver can accommodate 8-, 9-, or 10-bit data characters. The states of the C1[M] and C1[PE] bits and the C4[M10] bit determine the length of data characters. When receiving 9 or 10-bit data, bit C3[R8] is the ninth bit (bit 8).

52.4.3.2 Receiver bit ordering

When the S2[MSBF] bit is set, the receiver operates such that the first bit received after the start bit is the MSB of the data word. Likewise the bit received immediately preceding the parity bit (or the stop bit if parity is not enabled) is treated as the LSB for the data word. All necessary bit ordering is handled automatically by the module hence the format of the data read from receive data buffer is completely independent of the S2[MSBF] setting.

52.4.3.3 Character reception

During UART reception, the receive shift register shifts a frame in from the unsynchronized receiver input signal. After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the UART receive buffer. Additionally, the noise and parity error flags that are calculated during the receive process are also captured in the UART receive buffer. The receive data buffer is accessible via the D and C3[T8] registers. Additional received information flags regarding the receive dataword can be read in ED register. The S1[RDRF] flag is set if the number of resulting datawords in the receive buffer is equal to or greater than the number indicated by RWFIFO[RXWATER]. If the C2[RIE] is also set, the RDRF flag generates an RDRF interrupt request. Alternatively, by programming the C5[RDMAS] bit correctly a DMA request can be generated.

When 7816E is set/enabled and C7816[TTYPE] = 0, character reception operates slightly differently. Upon receipt of the parity bit, the validity of the parity bit is checked. If C7816[ANACK] is set and the parity check fails or if INIT and the received character is not a valid initial character, then a NACK is sent by the receiver. If the number of consecutive receive errors exceeds the threshold set by ET7816[RXTHRESHOLD] then the IS7816[RXT] flag is set and an interrupt generated if IE7816[RXTE] is set. If an error is detected (parity or invalid initial character) the data is NOT transferred from the receive shift register to the receive buffer. Instead, the data is overwritten by the next incoming data.

When the C7816[ISO_7816E] is set/enabled and C7816[ONACK] is set/enabled and the received character would result in the receive buffer overflowing a NACK is issued by the receiver. Additionally, the S1[OR] flag is set and interrupt issued if appropriate and the data in the shift register is discarded.

52.4.3.4 Framing errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, S1[FE] if S2[LBKDE] is disabled. A break character when S2[LBKDE] is disabled also sets the S1[FE] because a break character has no stop bit. The S1[FE] is set at the same time that received data is placed in the receive data buffer. Framing errors are not supported when C7816[ISO7816E] is set/ enabled. However, if the S1[FE] is set data will not be received when C7816[ISO7816E] is set.

52.4.3.5 Receiving break characters

The UART recognizes a break character when a start bit is followed by eight, nine, or ten logic 0 data bits and a logic 0 where the stop bit should be. Receiving a break character has these effects on UART registers:

- Sets the framing error flag, S1[FE].
- Writes an all "0" dataword to the data buffer, which may cause S1[RDRF] to set depending on the watermark and number of values in the data buffer.
- May set the overrun flag, S1[OR], noise flag, S1[NF], parity error flag, S1[PE], or the receiver active flag, S2[RAF].

The detection threshold for a break character can be adjusted when using an internal oscillator in a LIN system by setting the S2[LBKDE] bit. The UART break character detection threshold depends on the C1[M] and C1[PE] bits, the C4[LBKDE] bit, and the C4[M10] bit. Refer to the following table.

Table 52-350. Receive break character detection threshold

LBKDE	M	M10	PE	Threshold (bits)
0	0	—	—	10
0	1	0	—	11
0	1	1	0	11
0	1	1	1	12
1	0	—	—	11
1	1	—	—	12

While C4[LBKDE] is set, it will have these effects on the UART registers:

- Prevents the S1[RDRF], S1[FE], S1[NF], and S1[PF] flags from being set. However, if they are already set they will remain set.
- Sets the LIN break detect interrupt flag, S2[LBKDIF] if a LIN break character is received.

Break characters are not detected or supported when C7816[ISO_7816E] is set/enabled.

52.4.3.6 Hardware flow control

To support hardware flow control, the receiver can be programmed to automatically deassert and assert $\overline{\text{RTS}}$.

- $\overline{\text{RTS}}$ will remain asserted until the transfer is completed, even if the transmitter is disabled mid way through a data transfer, see Transceiver driver enable using $\overline{\text{RTS}}$ for more details.
- If the receiver request-to-send functionality is enabled, the receiver automatically deasserts $\overline{\text{RTS}}$ if the number of characters in the receiver data register is equal to or greater than receiver data buffer's watermark, RWFIFO[RXWATER].
- The receiver asserts $\overline{\text{RTS}}$ when the number of characters in the receiver data register is less than the watermark. It is not affected by whether RDRF is asserted.
- Even if $\overline{\text{RTS}}$ is deasserted, the receiver continues to receive characters until the receiver data buffer is full or is overrun.
- If the receiver request-to-send functionality is disabled, the receiver $\overline{\text{RTS}}$ remains deasserted.

The following figure shows receiver hardware flow control functional timing. Along with the actual character itself, RXD shows the start bit. The stop bit also indicated, with a dashed line if necessary. The watermark is set to 2.

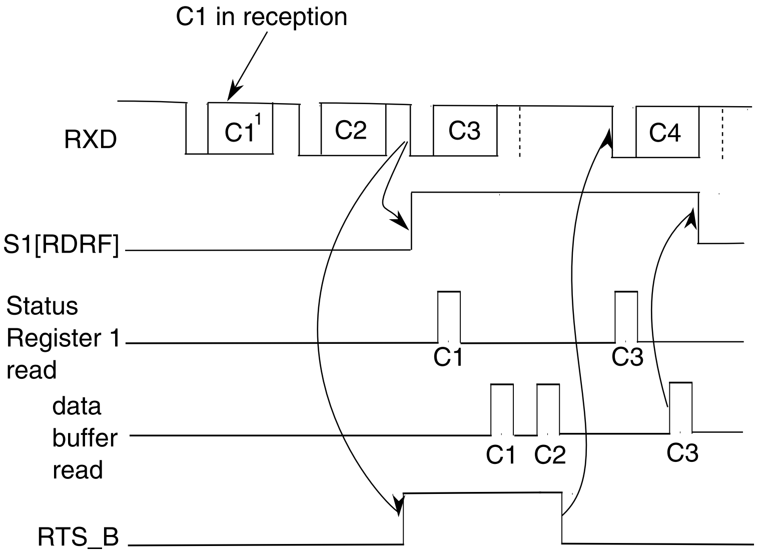


Figure 52-345. Receiver hardware flow control timing diagram

52.4.3.7 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 16-RT clock counter that filters noise and indicates when a '1' is being received.

52.4.3.7.1 Start bit detection

When S2[RXINV] is cleared, the first rising 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. Once 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.

52.4.3.7.2 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 RT clocks can be undetected by it regardless of whether it is seen in the first or second half of the count.

52.4.3.7.3 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 also is reset.

52.4.3.7.4 High-bit detection

At 16-RT 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 of the receiver's bit time sampling period, then the delay of a `0' is not recorded as noise.

52.4.3.8 Baud rate tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the

RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic zero.

As the receiver samples an incoming frame, it re-synchronizes the RT clock on any valid falling edge within the frame. Resynchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

52.4.3.8.1 Slow data tolerance

The following figure shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

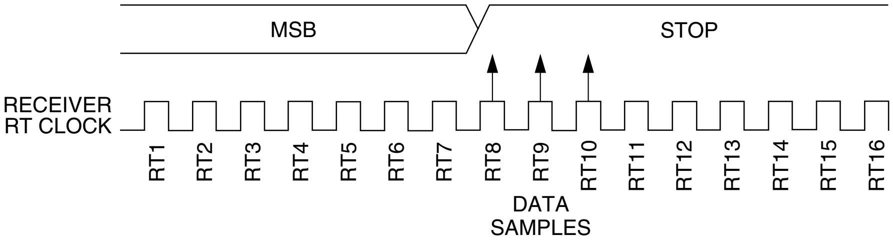


Figure 52-346. Slow data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the above figure, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 147 RT cycles (9 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

$$((154 - 147) \div 154) \times 100 = 4.54\%$$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the above figure, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 163 RT cycles (10 bit times × 16 RT cycles + 3 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

$$((170 - 163) \div 170) \times 100 = 4.12\%$$

52.4.3.8.2 Fast data tolerance

The following figure shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

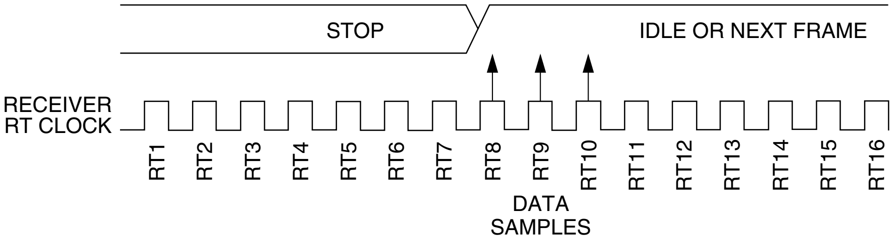


Figure 52-347. Fast data

For an 8-bit data character, data sampling of the stop bit takes the receiver 154 RT cycles (9 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the above figure, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 160 RT cycles (10 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

$((154 - 160) \div 154) \times 100 = 3.90\%$

For a 9-bit data character, data sampling of the stop bit takes the receiver 170 RT cycles (10 bit times × 16 RT cycles + 10 RT cycles).

With the misaligned character shown in the above figure, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 176 RT cycles (11 bit times × 16 RT cycles).

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

$((170 - 176) \div 170) \times 100 = 3.53\%$

52.4.3.9 Receiver wakeup

The C1[WAKE] bit determines how the UART is brought out of the standby state to process an incoming message. The C1[WAKE] bit enables either idle line wakeup or address mark wakeup.

Receiver wakeup is not supported when C7816[ISO_7816E] is set/enabled since multi-receiver systems are not allowed.

52.4.3.9.1 Idle input line wakeup (C1[WAKE] = 0)

In this wakeup method, an idle condition on the unsynchronized receiver input signal clears the C2[RWU] bit and wakes the UART. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] bit and return to the standby state. The C2[RWU] bit remains set and the receiver remains in standby until another idle character appears on the unsynchronized receiver input signal.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

When C2[RWU] is one and S2[RWUID] is zero, the idle character that wakes the receiver does not set the S1[IDLE] flag or the receive data register full flag, S1[RDRF]. The receiver wakes and waits for the first data character of the next message which will be stored in the receive data buffer. When S2[RWUID] and C2[RWU] bits are set and C1[WAKE] is cleared, any idle condition sets the S1[IDLE] flag and generates an interrupt if enabled.

Idle Input Line Wakeup is not supported when C7816[ISO_7816E] is set/enabled.

52.4.3.9.2 Address mark wakeup (C1[WAKE] = 1)

In this wakeup method, a logic 1 in the bit position immediately preceding the stop bit of a frame clears the C2[RWU] bit and wakes the UART. A logic 1 in the bit position immediately preceeding the stop bit marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its C2[RWU] bit and return to the standby state. The C2[RWU] bit remains set and the receiver remains in standby until another address frame appears on the unsynchronized receiver input signal.

A logic 1 in the bit position immediately preceding the stop bit clears the receiver's C2[RWU] bit before the stop bit is received and places the received data into the receiver data buffer.

Address mark wakeup allows messages to contain idle characters but requires that the bit position immediately preceding the stop bit be reserved for use in address frames.

If module is in standby mode and nothing triggers to wake the UART, no error flag is set even if an invalid error condition is detected on the receiving data line.

Address mark wakeup is not supported when C7816[ISO_7816E] is set/enabled.

52.4.3.9.3 Match address operation

Match address operation is enabled when the C4[MAEN1] or C4[MAEN2] bit is set. In this function, a frame received by the RX pin with a logic 1 in the bit position immediately preceding the stop bit is considered an address and is compared with the associated MA1 or MA2 register. The frame is only transferred to the receive buffer, and S1[RDRF] is set, if the comparison matches. All subsequent frames received with a logic 0 in the bit position immediately preceding the stop bit 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 frames with logic zero in the bit position immediately preceding the stop bit are also discarded. If both the C4[MAEN1] and C4[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Match Address operation functions in the same way for both MA1 and MA2 registers.

- If only one of C4[MAEN1] and C4[MAEN2] is asserted, a marked address is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If C4[MAEN1] and C4[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

Address match operation is not supported when C7816[ISO_7816E] is set/enabled.

52.4.4 Baud rate generation

A 13-bit modulus counter and a 5-bit fractional fine-adjust counter in the baud rate generator derive the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to the SBR[12:0] bits determines the module clock divisor. The SBR bits are in the UART baud rate registers (BDH and BDL). The baud rate clock is synchronized with the module clock and drives the receiver. The fractional fine-adjust counter adds fractional delays to the baud rate clock to allow fine trimming of the baud rate to match the system baud rate. The baud rate clock divided by 16 drives the transmitter. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to two sources of error:

- Integer division of the module clock may not give the exact target frequency. This error can be reduced by means of the fine-adjust counter.
- Synchronization with the module clock can cause phase shift.

The Table 52-351 lists the available baud divisor fine adjust values.

UART baud rate = UART module clock / (16 × (SBR[12:0] + BRFD))

The following table lists some examples of achieving target baud rates with a module clock frequency of 10.2 MHz, with and without fractional fine adjustment.

Table 52-351. Baud rates (example: module clock = 10.2 MHz)

Bits SBR (decimal)	Bits BRFA	BRFD value	Receiver clock (Hz)	Transmitter clock (Hz)	Target Baud rate	Error (%)
17	00000	0	600,000.0	37,500.0	38,400	2.3
16	10011	19/32=0.59375	614,689.3	38,418.08	38,400	0.047
33	00000	0	309,090.9	19,318.2	19,200	0.62
33	00110	6/32=0.1875	307,344.6	19,209.04	19,200	0.047
66	00000	0	154,545.5	9659.1	9600	0.62
133	00000	0	76,691.7	4793.2	4800	0.14
266	00000	0	38,345.9	2396.6	2400	0.14
531	00000	0	19,209.0	1200.6	1200	0.11
1062	00000	0	9604.5	600.3	600	0.05
2125	00000	0	4800.0	300.0	300	0.00
4250	00000	0	2400.0	150.0	150	0.00
5795	00000	0	1760.1	110.0	110	0.00

Table 52-352. Baud rate fine adjust

BRFA	Baud Rate Fractional Divisor (BRFD)
0 0 0 0 0	0/32 = 0
0 0 0 0 1	1/32 = 0.03125
0 0 0 1 0	2/32 = 0.0625
0 0 0 1 1	3/32 = 0.09375
0 0 1 0 0	4/32 = 0.125
0 0 1 0 1	5/32 = 0.15625
0 0 1 1 0	6/32 = 0.1875
0 0 1 1 1	7/32 = 0.21875
0 1 0 0 0	8/32 = 0.25
0 1 0 0 1	9/32 = 0.28125
0 1 0 1 0	10/32 = 0.3125
0 1 0 1 1	11/32 = 0.34375
0 1 1 0 0	12/32 = 0.375

Table continues on the next page...

Table 52-352. Baud rate fine adjust (continued)

BRFA	Baud Rate Fractional Divisor (BRFD)
0 1 1 0 1	13/32 = 0.40625
0 1 1 1 0	14/32 = 0.4375
0 1 1 1 1	15/32 = 0.46875
1 0 0 0 0	16/32 = 0.5
1 0 0 0 1	17/32 = 0.53125
1 0 0 1 0	18/32 = 0.5625
1 0 0 1 1	19/32 = 0.59375
1 0 1 0 0	20/32 = 0.625
1 0 1 0 1	21/32 = 0.65625
1 0 1 1 0	22/32 = 0.6875
1 0 1 1 1	23/32 = 0.71875
1 1 0 0 0	24/32 = 0.75
1 1 0 0 1	25/32 = 0.78125
1 1 0 1 0	26/32 = 0.8125
1 1 0 1 1	27/32 = 0.84375
1 1 1 0 0	28/32 = 0.875
1 1 1 0 1	29/32 = 0.90625
1 1 1 1 0	30/32 = 0.9375
1 1 1 1 1	31/32 = 0.96875

52.4.5 Data format (non ISO-7816)

Each data character is contained in a frame that includes a start bit and a stop bit. The rest of the data format depends upon UARTx_C1[M], UARTx_C1[PE], UARTx_S2[MSBF], and UARTx_C4[M10].

52.4.5.1 Eight-bit configuration

Clearing the UART_C1[M] configures the UART for 8-bit data characters, that is, eight bits are memory mapped in UART_D. A frame with eight data bits has a total of 10 bits. The most significant bit of the eight data bits can be used as an address mark to wake the receiver. If that bit is used in this way, then it serves as an address or data indication, leaving the remaining seven bits as actual data. When UART_C1[PE] is set, the 8th databit is automatically calculated as the parity bit. Refer to the following table.

Table 52-353. Configuration of 8-bit data format

UART_C1[PE]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	1	8	0	0	1
0	1	7	1[1]	0	1
1	1	7	0	1	1

[1] The address bit identifies the frame as an address character. See Receiver wakeup.

52.4.5.2 Nine-bit configuration

When UARTx_C1[M] is set and UARTx_C4[M10] is cleared the UART is configured for 9-bit data characters. The 9th bit is either UARTx_C3[T8/R8] or the internally generated parity bit if UARTx_C1[PE] is enabled. This results in a frame consisting of a total of 11 bits. In the event that the 9th data bit is selected to be UARTx_C3[T8] it will remain unchanged after transmission and can be used repeatedly without rewriting it unless the value needs to be changed. This feature may be useful when the 9th data bit is being used as an address mark.

When UARTx_C1[M] is set and UARTx_C4[M10] is set the UART is configured for 9-bit data characters, but the frame consists of a total of 12 bits. The 12 bits include the start and stop bits, the 9 data character bits and a 10th internal data bit. Note that if UARTx_C4[M10] is set UARTx_C1[PE] must also be set. In this case, the 10th bit is the internally generated parity bit. The 9th bit is can either be used as a address mark or a 9th data bit.

Refer to the following table.

Table 52-354. Configuration of 9-bit data formats

C1[PE]	UC1[M]	C1[M10]	Start bit	Data bits	Address bits	Parity bits	Stop bit
0	0	0	See Eight-bit configuration				
0	0	1	Invalid configuration				
0	1	0	1	9	0	0	1
0	1	0	1	8	1[1]	0	1
0	1	1	Invalid Configuration				
1	0	0	See Eight-bit configuration				
1	0	1	Invalid Configuration				
1	1	0	1	8	0	1	1
1	1	1	1	9	0	1	1
1	1	1	1	8	1[2]	1	1

1. The address bit identifies the frame as an address character.
2. The address bit identifies the frame as an address character.

Note

> Unless in 9-bit mode with M10 set, do not use address mark wakeup with parity enabled.

52.4.5.3 Timing examples

Timing examples of these configurations in the NRZ mark/space data format are illustrated in the following figures. The timing examples show all of the configurations in the following sub-sections along with the LSB and MSB first variations.

52.4.5.3.1 Eight-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

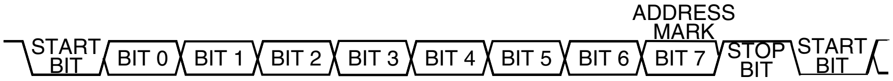


Figure 52-348. Eight bits of data with LSB first

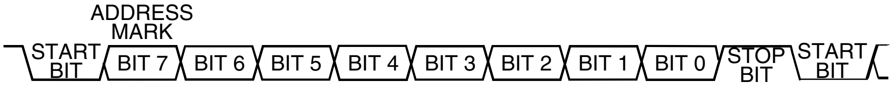


Figure 52-349. Eight bits of data with MSB first

52.4.5.3.2 Eight-bit format with parity enabled

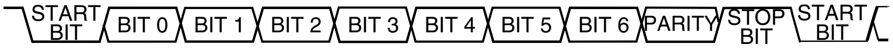


Figure 52-350. Seven bits of data with LSB first and parity

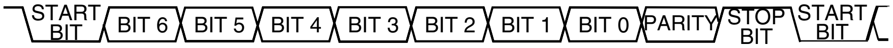


Figure 52-351. Seven bits of data with MSB first and parity

52.4.5.3.3 Nine-bit format with parity disabled

The most significant bit can be used for address mark wakeup.

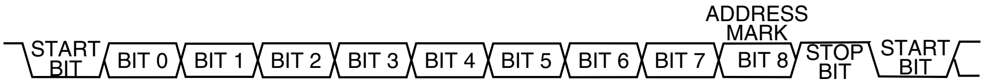


Figure 52-352. Nine bits of data with LSB first

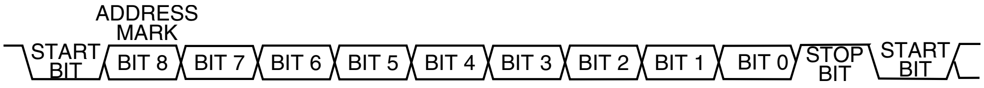


Figure 52-353. Nine bits of data with MSB first

52.4.5.3.4 Nine-bit format with parity enabled

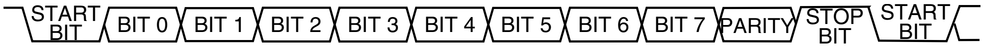


Figure 52-354. Eight bits of data with LSB first and parity

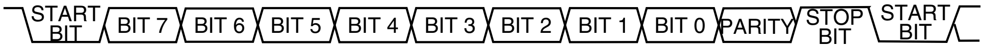


Figure 52-355. Eight bits of data with MSB first and parity

52.4.5.3.5 Non-memory mapped tenth bit for parity

The most significant memory-mapped bit can be used for address mark wakeup.

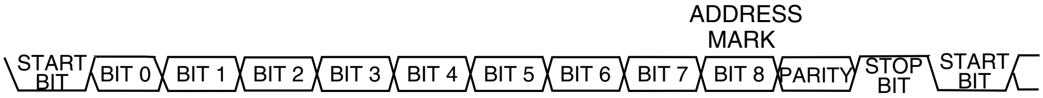


Figure 52-356. Nine bits of data with LSB first and parity

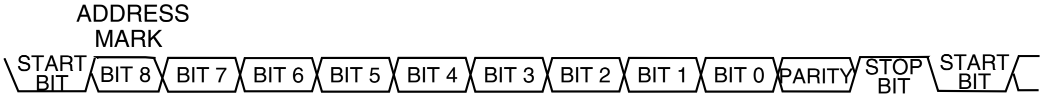


Figure 52-357. Nine bits of data with MSB first and parity

52.4.6 Single-wire operation

Normally, the UART uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the UART and the UART implements a half-duplex serial connection. The UART uses the TXD pin for both receiving and transmitting.

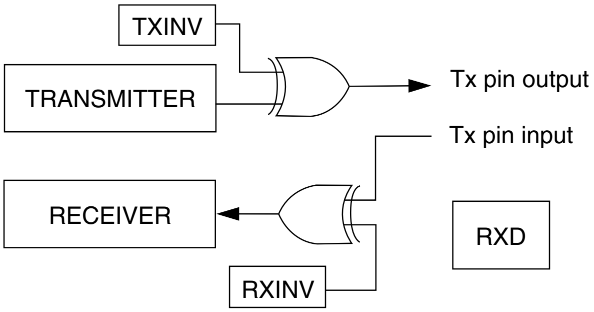


Figure 52-358. Single-wire operation (C1[LOOPS] = 1, C1[RSRC] = 1)

Enable single-wire operation by setting the C1[LOOPS] bit and the receiver source bit, C1[RSRC]. Setting the C1[LOOPS] bit disables the path from the unsynchronized receiver input signal to the receiver. Setting the C1[RSRC] bit connects the receiver input to the output of the TXD pin driver. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1). When C7816[ISO_7816EN] is set, it is not a requirement that both C2[TE] and C2[RE] are set.

52.4.7 Loop operation

In loop operation the transmitter output goes to the receiver input. The unsynchronized receiver input signal is disconnected from the UART.

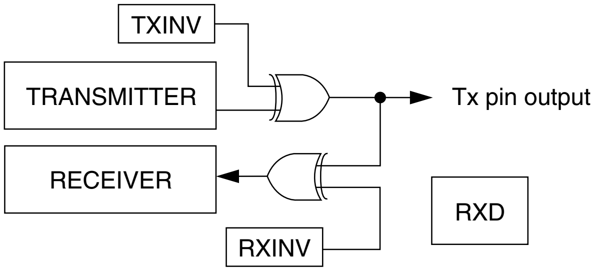


Figure 52-359. Loop operation (C1[LOOPS] = 1, C1[RSRC] = 0)

Enable loop operation by setting the C1[LOOPS] bit and clearing the C1[RSRC] bit. Setting the C1[LOOPS] bit disables the path from the unsynchronized receiver input signal to the receiver. Clearing the C1[RSRC] bit connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (C2[TE] = 1 and C2[RE] = 1). When C7816[ISO_7816EN] is set, it is not a requirement that both C2[TE] and C2[RE] are set.

52.4.8 ISO-7816 / smartcard support

The UART provides mechanisms to support the ISO-7816 protocol that is commonly used to interface with smartcards. The ISO-7816 protocol is an NRZ, single wire, half-duplex interface. The TxD pin is used in open-drain mode since the data signal is used for both transmitting and receiving. There are multiple subprotocols within the ISO-7816 standard. The UART supports both T = 0 and T = 1 protocols. The module also provides for automated initial character detection and configuration which allows for support of both direct convention and inverse convention data formats. A variety of interrupts specific to 7816 are provided in addition to the general interrupts to assist software.

Additionally the module is able to provide automated NACK responses and has programing automated retransmission of failed packets. An assortment of programmable timeouts and guard band times are also supported.

The term elemental time unit (ETU) is frequently used in the context of ISO-7816. This concept is used to relate the frequency that the system (UART) is running at and the frequency that data is being transmitted and received. One ETU represents the time it takes to transmit or receive a single bit. For example, a standard 7816 packet, excluding any guard time or NACK elements is 10 ETUs (start bit, 8 data bits and a parity bit). Guard times and wait times are also measured in ETUs.

NOTE

The ISO-7816 specification may have certain configuration options that are reserved. In order to maintain maximum flexibility to support future 7816 enhancements or devices which may not strictly conform to the specification, the UART does not prevent those options being used. Further, the UART may provide configuration options that exceed the flexibility of options explicitly allowed by the 7816 specification. Failure to correctly configure the UART may result in unexpected behavior or incompatibility with the ISO-7816 specification.

52.4.8.1 Initial characters

In ISO-7816 with T = 0 mode, the UART can be configured to use the C7816[INIT] bit to detect the next valid initial character, referred to by the ISO-7816 specifically as a TS character. When the initial character is detected, the UART provides the host processor with an interrupt if IE7816[INITDE] is set. Additionally, the UART will alter the S2[MSBF], C3[TXINV] and S2[RXINV] register fields automatically based on the initial character. The corresponding initial character and resulting register settings are listed in the following table.

Table 52-355. Initial character automated settings

Initial character (bit 1-10)	Initial character (hex)	MSBF	TXINV	RXINV
LHHL LLL LLH inverse convention	3F	1	1	1
LHHL HHH LLH direct convention	3B	0	0	0

The register fields S2[MSBF], C3[TXINV] and S2[RXINV] must be reset to their default values before C7816[INIT] bit is set. Once the C7816[INIT] bit is set, the receiver will search all received data for the first valid initial character. Detecting a Direct Convention Initial Character will cause no change to S2[MSBF], C3[TXINV] and S2[RXINV] fields while detecting an Inverse Convention Initial Character will cause these fields to set automatically. All data received which is not a valid initial character will be ignored and all flags resulting from the invalid data will be blocked from asserting. If the C7816[ANACK] bit is set, a NACK will be returned for invalid received initial characters and a RXT interrupt will be generated as programmed.

52.4.8.2 Protocol T = 0

When T = 0 protocol is selected, a relatively complex error detection scheme is used. Data characters are formatted as illustrated in the following figure. This scheme is also used for answer to reset and PPS formats.

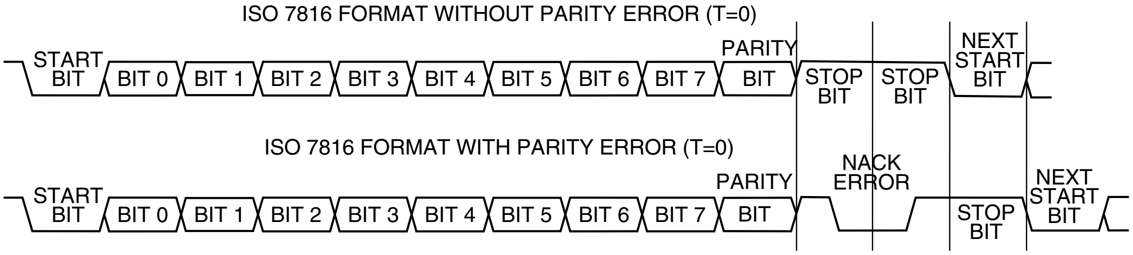


Figure 52-360. ISO-7816 T = 0 data format

As with other protocols supported by the UART the data character includes a start bit. However, in this case there are two stop bits rather than the typical single stop bit. In addition to a standard even parity check, the receiver has the ability to generate and return a NACK during the second half of the first stop bit period. The NACK must be at least one time period (ETU) in length and no more than 2 time periods (ETU) in length. The transmitter must wait for at least two time units (ETU) after detection of the error signal before attempting to retransmit the character.

It is assumed that the UART and the device (smartcard) know in advance which device is receiving and which is transmitting. No special mechanism is supplied by the UART to control receive and transmit in the mode other than the C2[TE] and C2[RE] bits. Initial Character Detect feature is also supported in this mode.

52.4.8.3 Protocol T = 1

When T = 1 protocol is selected the NACK error detection scheme is not used. Rather, the parity bit is used on a character basis and a CRC or LRC is used on the block basis (i.e. each group of characters). As such, in this mode the data format allows for a single stop bit although additional inactive bit periods may be present between the stop bit and the next start bit. Data characters are formatted as illustrated in the following figure.

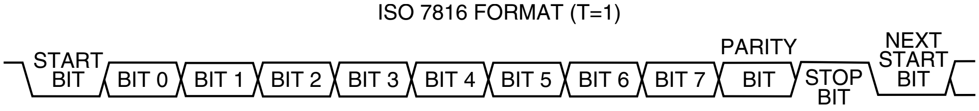


Figure 52-361. ISO 7816 T=1 data format

The smallest data unit that is transferred is a block. A block is made up of several data characters and may vary in size depending on the block type. The UART does not provide a mechanism to decode the block type. As part of the block, an LRC or CRC is included. The UART does not calculate the CRC or LRC for transmitted blocks nor does it verify the validity of the CRC or LRC for received blocks. The 7816 protocol requires that the initiator and the smartcard (device) takes alternate turns in transmitting and receiving blocks. When the UART detects that the last character in a block has been transmitted it will automatically clear the C2[TE] bit and enter receive mode. Hence, software must program the transmit buffer with the next data to be transmitted and then enable the C2[TE] bit once software has determined that the last character of the received block has been received. The UART detects that the last character of the transmit block has been sent when TL7816[TLEN] = 0 and four additional characters have been sent. The four additional characters are made up of three prior to TL7816[TLEN] decrementing (prologue) and one after TL7816[TLEN] = 0, the final character of the epilogue.

52.4.8.4 Wait time and guard time parameters

The ISO-7816 specification defines several wait time and guard time parameters. The UART allows for flexible configuration and violation detection of these settings. On reset the wait time (IS7816[WT]) defaults to 9600 ETUs and guard time (GT) to 12 ETUs. These values are controlled by parameters in the WP7816, WN7816 and WF7816 registers. Additionally the value of C7816[TTYPE] also factors into the calculation. The formulas used to calculate the number ETU for each wait time and guard time value are shown in the following table.

Wait time (WT) is defined as the maximum allowable time between the leading edge of a character transmitted by the device (smartcard) and the leading edge of the previous character that was transmitted by the UART or the device. Likewise character wait time

(CWT) is defined as the maximum allowable time between the leading edge of two characters within the same block, and block wait time (BWT) is defined as the maximum time between the leading edge character of the last block received by the device/ smartcard and the leading edge of the first character transmitted by the device/smartcard.

Guard time (GT) is defined as the minimum allowable time between the leading edge of two consecutive characters. Character guard time (CGT) is the minimum allowable time between the leading edges of two consecutive characters in the same direction (transmission or reception). Block guard time (BGT) is the minimum allowable time between the leading edges of two consecutive characters in opposite directions (transmission then reception or reception then transmission).

The GT and WT counters reset whenever C7816[TTYPE] = 1 or C7816[ISO_7816E] = 0 or a new dataword start bit has been received or transmitted as specified by the counter descriptions. The CWT, CGT, BWT, BGT counters reset whenever C7816[TTYPE] = 0 or C7816[ISO_7816E] = 0 or a new dataword start bit has been received or transmitted as specified by the counter descriptions. When C7816[TTYPE] = 1 some of the counter values require an assumption regarding the first data transferred when the UART first starts. This assumption is required when the 7816E has been disabled, when transition from C7816[TTYPE] = 0 to C7816[TTYPE] = 1 or when coming out of reset. In this case, it is assumed that the previous (non-existent) transfer was a received transfer.

The UART will automatically handle GT, CGT and BGT such that the UART will not send a packet prior to the corresponding guard time expiring.

Table 52-356. Wait and guard time calculations

Parameter	Reset value [ETU]	C7816[TTYPE] = 0 [ETU]	C7816[TTYPE] = 1 [ETU]
Wait time (WT)	9600	WI × 960 × GTFD	Not used
Character wait time (CWT)	Not used	Not used	$11 + 2^{CWI}$
Block wait time (BWT)	Not used	Not used	$11 + 2^{BWI} \times 960 \times GTFD$
Guard time (GT)	12	**GTN not wqual to 255** 12 + GTN **GTN wqual to 255** 12	Not used
Character guard time (CGT)	Not used	Not used	**GTN not equal to 255** 12 + GTN **GTN equal to 255** 11
Block guard time (BGT)	Not used	Not used	22

52.4.8.5 Baud rate generation

The value in WF7816[GTFD] does not impact the clock frequency. The SBR and BRFD are used to generate the clock frequency. This clock frequency is used by the UART only and is not seen by the device (smartcard). The transmitter clocks operates at 1/16 the frequency of the receive clock so that the receiver is able to sample the received value 16 times during the ETU.

52.4.8.6 UART restrictions in ISO-7816 operation

Due to the flexibility of the UART module, there are several features and interrupts that are not supported while running in ISO-7816 mode. These restrictions are documented within the register bit definitions.

52.4.9 Infrared interface

The UART 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 UART. 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 design covers data rates only between 2.4 kbits/s and 115.2 kbits/s.

The UART has an infrared transmit encoder and receive decoder. The UART transmits serial bits of data which 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 MCU). The narrow pulses are then stretched by the infrared receive decoder to get back to a serial bit stream to be received by the UART. 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 low pulses.

The infrared submodule receives its clock sources from the UART. One of these two clocks are selected in the infrared submodule in order to generate either 3/16, 1/16, 1/32 or 1/4 narrow pulses during transmission.

52.4.9.1 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 zero bit and no pulse for a one bit. The narrow pulse is sent in the middle of the bit with a duration of 1/32, 1/16, 3/16 or 1/4 of a bit time. A narrow high pulse is transmitted for a zero bit when C3[TXINV] is cleared, while a narrow low pulse is transmitted for a zero bit when C3[TXINV] is set.

52.4.9.2 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 zero received and no pulse is expected for each one received. A narrow high pulse is expected for a zero bit when S2[RXINV] is cleared, while a narrow low pulse is expected for a zero bit when S2[RXINV] is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

52.5 Reset

All registers reset to a particular value are indicated in Memory map and registers.

52.6 System level interrupt sources

There are several interrupt signals that are sent from the UART. The following table lists the interrupt sources generated by the UART. The local enables for the UART interrupt sources are described in this table. Details regarding the individual operation of each interrupt are contained under various sub-sections of Memory map and registers. However, RXEDGIF description also outlines additional details regarding the RXEDGIF interrupt because of its complexity of operation. Any of the UART interrupt requests listed in the table can be used to bring the CPU out of wait mode.

Table 52-357. UART interrupt sources

Interrupt Source	Flag	Local enable	DMA select
Transmitter	TDRE	TIE	TDMAS = 0
Transmitter	TC	TCIE	-
Receiver	IDLE	ILIE	-
Receiver	RDRF	RIE	RDMAS = 0

Table continues on the next page...

Table 52-357. UART interrupt sources (continued)

Interrupt Source	Flag	Local enable	DMA select
Receiver	LBKDIF	LBKDIE	-
Receiver	RXEDGIF	RXEDGIE	-
Receiver	OR	ORIE	-
Receiver	NF	NEIE	-
Receiver	FE	FEIE	-
Receiver	PF	PEIE	-
Receiver	RXUF	RXUFE	-
Transmitter	TXOF	TXOFE	-
Receiver	WT	WTWE	-
Receiver	CWT	CWTE	-
Receiver	BWT	BWTE	-
Receiver	INITD	INITDE	-
Receiver	TXT	TXTE	-
Receiver	RXT	RXTE	-
Receiver	GTV	GTVE	-

52.6.1 RXEDGIF description

The S2[RXEDGIF] is set when an active edge is detected on the RxD pin. Hence, the active edge can only be detected when in two wire mode. A RXEDGIF interrupt is only generated when S2[RXEDGIF] is set. If RXEDGIE is not enabled prior to S2[RXEDGIF] getting set, an interrupt is not generated until S2[RXEDGIF] bit gets set.

52.6.1.1 RxD edge detect sensitivity

Edge sensitivity can be software programmed to be either falling or rising. The polarity of the edge sensitivity is selected using the S2[RXINV] bit. To detect falling edge S2[RXINV] is programmed to zero and to detect rising edge S2[RXINV] is programmed to one.

Synchronizing logic is used prior to detect edges. Prior to detecting an edge, the receive data on RxD input must be at the de-asserted logic level. A falling edge is detected when the RxD input signal is seen as a logic 1 (the deasserted level) during one module clock cycle and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input is seen as a logic 0 during one module clock cycle and then a logic 1 during the next cycle.

52.6.1.2 Clearing RXEDGIF interrupt request

Writing a logic 1 to the S2[RXEDGIF] bit immediately clears the RXEDGIF interrupt request even if the RxD input remains asserted. S2[RXEDGIF] will remain set if another active edge is detected on RxD while attempting to clear the S2[RXEDGIF] flag by writing a 1 to it.

52.6.1.3 Exit from low-power modes

The receive input active edge detect circuit is still active on low power modes (wait and stop). An active edge on the receive input brings the CPU out of low power mode if the interrupt is not masked (S2[RXEDGIF]=1).

52.7 DMA operation

In the transmitter, flags S1[TDRE] can be configured to assert a DMA transfer request. In the receiver, flag S1[RDRF], can be configured to assert a DMA transfer request. The following table shows the configuration bit settings required to configure each flag for DMA operation.

Table 52-358. DMA configuration

Flag	Request enable bit	DMA select bit
TDRE	TIE = 1	TDMAS = 1
RDRF	RIE = 1	RDMAS = 1

When a flag is configured for a DMA request, its associated DMA request is asserted when the flag is set. When the S1[RDRF] flag is configured as a DMA request, the clearing mechanism of reading S1 register followed by reading D register does not clear the associated flag. The DMA request remains asserted until an indication is received that the DMA transactions are done. When this indication is received, the flag bit and the associated DMA request are cleared. If the DMA operation failed to remove the situation that caused the DMA request another request will be issued.

52.8 Application information

This section describes the UART application information.

52.8.1 Transmit/receive data buffer operation

The UART has independent receive and transmit buffers. The size of these buffers may vary depending on the implementation of the module. The implemented size of the buffers is a fixed constant via the PFIFO[TXFIFOSIZE] and PFIFO[RXFIFOSIZE] fields. Additionally, legacy support is provided that allows for the FIFO structure to operate as a depth of one. This is the default/reset behavior of the module and can be adjusted using the PFIFO[RXFE] and PFIFO[TXFE] bits. Individual watermark levels are also provided for transmit and receive.

There are multiple ways to ensure that a data block (set of characters) has completed transmission. These methods include:

1. Set TXFIFO[TXWATER] to 0. The TDRE flag will assert when there is no further data in the transmit buffer. Alternatively the S1[TC] flag can be used to indicate when the transmit shift register is also empty.
2. Poll the TCFIFO[TXCOUNT] field. Assuming that only data for a data block has been put into the data buffer, when TCFIFO[TXCOUNT] = 0 all data has been transmitted or is in the process of transmission.
3. The S1[TC] flag can be monitored. When S1[TC] asserts it indicates that all data has been transmitted and there is no data currently being transmitted in the shift register.

52.8.2 ISO-7816 initialization sequence

This section outlines how to program the UART for ISO-7816 operation. Elements such as procedures to power up or power down the smartcard, and when to take those actions, are beyond the scope of this description. To setup the UART for ISO-7816 operation:

1. Select a baud rate. Write this value to the UART baud registers (BDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the BDH has no effect without also writing to BDL. According to the 7816 specification the initial (default) baud rating setting should be Fi = 372 and Di = 1 and a maximum frequency of 5 MHz. In other words the BDH, BDL and C4 registers should be programmed such that the transmission frequency should be 1/372th of the clock provided to the smartcard device and should not exceed 5 MHz.
2. Write to set BDH[LBKDIE] = 0.

3. Write to C1 to configure word length, parity, and other configuration bits (LOOPS, RSRC) and set C1[M] = 1, C1[PE] = 1, C1[PT] = 0.
4. Write to set S2[RWUID] = 0, S2[LBKDE] = 0.
5. Write to set MODEM[RXRTSE] = 0, MODEM[TXRTSPOL] = 0, MODEM[TXRTSE] = 0, and MODEM[TXCTSE] = 0.
6. Write to set up interrupt enable bits desired (C3[ORIE], C3[NEIE], C3[PEIE], and C3[FEIE])
7. Write to set C4[MAEN1] = 0 and C4[MAEN2] = 0.
8. Write to C5 register and configure DMA control register bits as desired for application.
9. Write to set C7816[INIT] = 1,C7816[TTYPE] = 0, 7C7816[ISO_7816E] = 1. Program C7816[ONACK] and C7816[ANACK] as desired.
10. Write to IE7816 register to set interrupt enable parameters as desired.
11. Write to ET7816 register and set as desired.
12. Write to set C2[ILIE] = 0, C2[RE] = 1, C2[TE] = 1, C2[RWU] = 0 and C2[SBK] = 0. Setup interrupt enables C2[TIE], C2[TCIE] and C2[RIE] as desired.

At this time the UART will start listening for an initial character. Once identified it will automatically adjust the S2[MSBF], C3[TXINV] and S2[RXINV] bits. The software should then receive and process an answer to reset. Upon processing the answer to reset software should write to set C2[RE] = 0 and C2[TE] = 0. Software should then adjust 7816 specific and UART generic parameters to match and configuration data that was received during the answer on reset period. Once the new settings have been programmed (including the new baud rate and C7816[TTYPE]) the C2[RE] and C2[TE] can be re-enabled as required.

52.8.2.1 Transmission procedure for (C7816[TTYPE] = 0)

When the protocol selected is C7816[TTYPE] = 0, it is assumed that the software has a prior knowledge of who should be transmitting and receiving. Hence, no mechanism is provided for automated transmission/receipt control. Software should monitor the S1[TDRE] flag (or configure for an interrupt) and provide additional data for transmission as appropriate. Additionally, software should set C2[TE] = 1 and control TXDIR whenever it is the UART's turn to transmit information. For ease of monitoring it is suggested that only data to be transmitted until the next receiver/transmit switch over be loaded into the transmit buffer/FIFO.

52.8.2.2 Transmission procedure for (C7816[TTYPE] = 1)

When the protocol selected is C7816[TTYPE] = 1, data is transferred in blocks. Prior to starting a transmission software should write the size (number of bytes) for the Information Field portion of the block in to the TLEN register. If a CRC is being transmitted for the block the value in TLEN should be one more than the size of the information field. Software should then set C2[TE] = 1, and C2[RE] = 1. Software should then monitor the S1[TDRE] flag / interrupt and write the prologue, Information and epilogue field to the transmit buffer. The TLEN register will automatically decrement (except for prologue bytes and the final epilogue byte). When the final epilogue byte has been transmitted the UART automatically clears the C2[TE] bit to 0, and the UART automatically starts capturing the response to the block that was transmitted. Once software has detected the receipt of the response, the transmission process should be repeated as needed with sufficient urgency to ensure that the block wait time and character wait times are not violated.

52.8.3 Initialization sequence (non ISO-7816)

To initiate an UART transmission:

1. Configure the UART:
   a. Select a baud rate. Write this value to the UART baud registers (BDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the BDH register has no effect without also writing to BDL register.
   b. Write to C1 register to configure word length, parity, and other configuration bits (LOOPS, RSRC, M, WAKE, ILT, PE, PT). Write to C4, MA1 and MA2 register to configure.
   c. Enable the transmitter, interrupts, receiver, and wakeup as required by writing to the C2 register bits (TIE, TCIE, RIE, ILIE, TE, RE, RWU, SBK), S2 register bits (MSBF, BRK13) and C3 register bits (ORIE, NEIE, PEIE, FEIE). A preamble or idle character is then shifted out of the transmitter shift register.
2. Transmit procedure for each byte:
   a. Monitor the S1[TDRE] flag by reading the S1 or responding to the TDRE interrupt. Or monitor the amount of free space in the transmit buffer directly using TCFIFO[TXCOUNT].

b. If the TDRE flag is set, or there is space in the transmit buffer, write the data to be transmitted to (C3[T8]/D). A new transmission will not result until data exists in the transmit buffer.

3. Repeat step 2 for each subsequent transmission.

Note

> During normal operation, the S1[TDRE] flag is set when the shift register is loaded with the next data to be transmitted from the transmit buffer and the number of datawords contained in the transmit buffer is less than or equal to the value in TWFIFO[TXWATER], which occurs 9/16ths of a bit time after the start of the stop bit of the previous frame.

To separate messages with preambles with minimum idle line time, use this sequence between messages:

1. Write the last dataword of the first message to C3[T8]/D.
2. Wait for the S1[TDRE] flag to go high (with TWFIFO[TXWATER] = 0), indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting the C2[TE] bit.
4. Write the first (and subsequent) datawords of the second message to C3[T8]/D.

52.8.4 Overrun (OR) flag implications

To be flexible the overrun flag (OR) operates slight differently depending on the mode of operation. As such there may be implications that need to be carefully considered. This section clarifies that behavior and the resulting implications. Regardless of mode, if a dataword is received while the S1[OR] flag is set, the S1[RDRF] and S1[IDLE] flags are blocked from asserting. If the S1[RDRF] or S1[IDLE] flag were previously asserted they will remain asserted until cleared.

52.8.4.1 Overrun operation

The assertion of the S1[OR] flag indicates that a significant event has occurred. The assertion indicates that received data has been lost since there was a lack of room to store it in the data buffer. Hence, while the S1[OR] flag is set no further data will be stored in the data buffer until the S1[OR] flag is cleared. This ensures that the application will be able to handle the overrun condition.

In most applications since the total amount of lost data is known, the application will desire to return the system to a known state. Prior to the S1[OR] flag being cleared all received data will be dropped. To do this the software would:

1. Remove data from the receive data buffer. This could be done by reading data from the data buffer and processing it if the data in the FIFO was still valuable when though the overrun event occurred or using the CFIFO[RXFLUSH] bit to clear the buffer.
2. Clear the S1[OR] flag. Note that if data was cleared using the CFIFO[RXFLUSH] bit, then clearing the S1[OR] flag will result in the SFIFO[RXUF] flag asserting because the only way to clear the S1[OR] requires reading additional information from the FIFO. Care should be taken to disable the SFIFO[RXUF] interrupt prior to clearing the OR flag and then clearing the SFIFO[RXUF] flag after the OR flag has been cleared.

Note that in some applications if an overrun event is responded to fast enough, the lost data can be recovered. For example when C7816[ISO_7816E] is asserted, C7816[TTYPE]=1 and C7816[ONACK] = 1 the application may reasonably be able to determine if the lost data will be resent by the device. In this scenario flushing the receiver data buffer might not be required. Rather, if the S1[OR] flag is cleared the lost data may be resent and hence recoverable.

When LIN break detect (LBKDE) is asserted the S1[OR] flag has significantly different behavior than in other modes. The S1[OR] bit will be set, regardless of how much space is actually available in the data buffer, if a LIN break character has been detected and the corresponding flag (S2[LBKDIF]) is not cleared before the first data character is received after the S2[LBKDIF] asserted. This behavior is intended to allow software sufficient time to read the LIN break character from the data buffer to ensure that a break character was actually detected. The checking of the break character was used on some older implementations and is hence supported for legacy reasons. Applications that do not require this checking can simply clear the S2[LBKDIF] without checking the stored value to ensure it is a break character.

52.8.5 Overrun NACK considerations

When C7816[ISO_7816E] is enabled and C7816[TTYPE] = 0 the retransmission feature of the 7816 protocol can be used to help avoid lost data when the data buffer overflows. Using C7816[ONACK] the module can be programmed to issue a NACK on an overflow event. Assuming that the device (smartcard) has implemented retransmission, the lost data will be retransmitted. While useful, there is a programming implication which may require special consideration. The need to transmit a NACK must be determined and committed to prior to the dataword being fully received. While the NACK is being received it is possible that the application code will read the data buffer such that sufficient room will be made to store the dataword that is being NACKed. Even if room has been made in the data buffer once the transmission of a NACK is completed, the received data will always be discarded as a result of an overflow and the ET7816[RXTHRESHOLD] value will be incremented by one. However, if sufficient space now exists to write the received data which was NACK'ed the S1[OR] flag will be blocked and kept from asserting.

52.8.6 Match address registers

The two match address registers allow a second match address function for a broadcast or general call address to the serial bus, as an example.

52.8.7 Modem feature

This section describes the modem features.

52.8.7.1 Ready-to-receive using $\overline{\text{RTS}}$

To help to stop overrun of the receiver data buffer, the $\overline{\text{RTS}}$ signal can be used by the receiver to indicate to another UART that it is ready to receive data. The other UART can send the data when its $\overline{\text{CTS}}$ signal is asserted. This handshaking conforms to the TIA-232-E standard. A transceiver is necessary if the required voltage levels of the communication link do not match the voltage levels of the UART's $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$ signals.

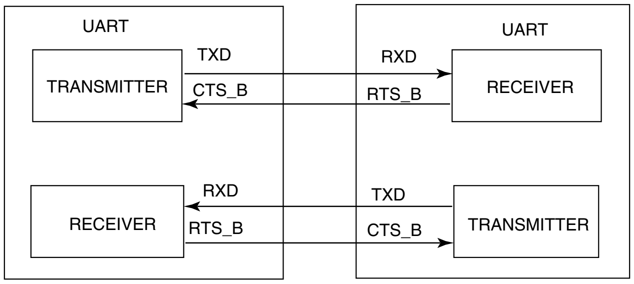


Figure 52-362. Ready-to-receive

The transmitter's $\overline{\text{CTS}}$ signal can be used for hardware flow control whether its $\overline{\text{RTS}}$ signal is used for hardware flow control, transceiver driver enable, or not at all.

52.8.7.2 Transceiver driver enable using $\overline{\mathrm{RTS}}$

RS-485 is a multiple drop communication protocol in which the UART transceiver's driver is 3-stated unless that UART is driving. The $\overline{\mathrm{RTS}}$ signal can be used by the transmitter to enable the driver of a transceiver. The polarity of $\overline{\mathrm{RTS}}$ can be matched to the polarity of the transceiver's driver enable signal. Refer to the following figure.

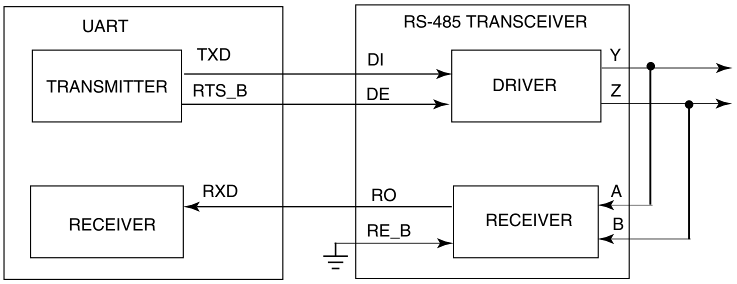


Figure 52-363. Transceiver driver enable using $\overline{\mathrm{RTS}}$

In the figure, the receiver enable signal is asserted. Another option for that 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 UART in single-wire mode, freeing the RXD pin for other uses.

52.8.8 IrDA minimum pulse width

The IrDA specifies a minimum pulse width of 1.6 μs. The UART hardware does not include a mechanism to restrict/force the pulse width to be greater than or equal to 1.6 μs. However, configuring the baud rate to 115.2 kbit/s and the narrow pulse width to 3/16 of a bit time results in a pulse width of 1.6 μs.

52.8.9 Clearing 7816 wait timer (WT, BWT, CWT) interrupts

The 7816 wait timer interrupts associated with IS7816[WT], IS7816[BWT] and IS7816[CWT] will automatically reassert if they are cleared and the wait time is still violated. This behavior is similar to most of the other interrupts on the UART as in most cases if the condition that caused the interrupt to trigger still exists when the interrupt is cleared, than the interrupt will reassert. For example, consider the following scenario:

1. IS7816[WT] is programmed to assert after 9600 cycles of unresponsiveness.
2. The 9600 cycles pass without a response resulting in the WT interrupt asserting.
3. The IS7816[WT] is cleared at cycle 9700 by the interrupt service routine.

4. After the WT interrupt has been cleared, the smartcard remains unresponsive. At cycle 9701 the WT interrupt will reasserted.

If the intent of clearing the interrupt is such that it does not reassert, the interrupt service routine must remove or clear the condition that originally caused the interrupt to assert prior to clearing the interrupt. There are multiple ways that this can be accomplished including ensuring that an event that results in the wait timer resetting occurs such as the transmission of another packet.

52.8.10 Legacy and reverse compatibility considerations

Recent versions of the UART have added several new features. Whenever reasonably possible, reverse compatibility was maintained. However, in some cases this was either not feasible or the behavior was deemed as not intended. This section describes several differences to legacy operation that resulted from these recent enhancements. If application codes from previous versions is used, they should be reviewed and modified to take the following items into account. Depending on the application code, additional items that are not listed here may also need to be considered.

1. Various reserved registers and register bits are used, such as MSFB and M10.
2. This module now generates an error when invalid address spaces are used.
3. While documentation indicated otherwise, in some cases it was possible for S1[IDLE] to assert even if S1[OR] was set.
4. The S1[OR] flag will be set only if the data buffer (FIFO) does not have sufficient room. Previously, the data buffer was always a fixed size of one and the S1[OR] flag would set so long as the S1[RDRF] flag was set even if there was room in the data buffer. While the clearing mechanism has remained the same for the S1[RDRF] flag, keeping the OR flag assertion tied to the RDRF event rather than the data buffer being full would have greatly reduced the usefulness of the buffer when its size is larger than one.
5. Previously, when the C2[RWU] was set (and WAKE = 0), the IDLE flag could reassert up to every bit period causing an interrupt and requiring the host processor to reassert the C2[RWU] bit. This behavior has been modified. Now, when the C2[RWU] is set (and WAKE = 0), at least one non-idle bit must be detected before an idle can be detected.

Chapter 53
Secured digital host controller (SDHC)

53.1 Introduction

NOTE

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

The chapter is intended for a module driver software developer. It describes module-level operation and programming.

53.2 Overview

53.2.1 Supported types of cards

Different types of cards supported by the SDHC are described briefly as follows:

The multi-media card (MMC) is a universal low cost data storage and communication media that is designed to cover a wide area of applications including mobile video and gaming. Old MMC cards are 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 environment 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 has 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 figure does not show cards with reduced size or mini cards.

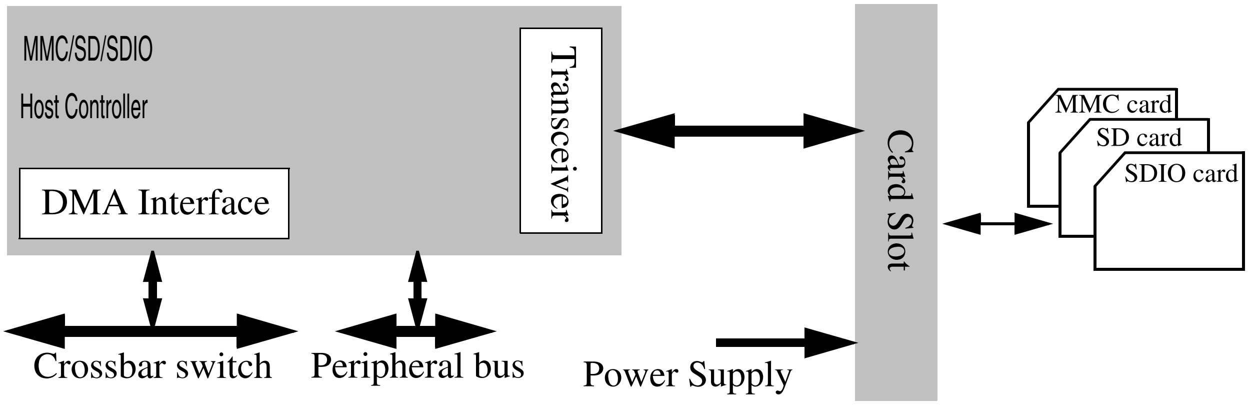


Figure 53-1. System connection of the SDHC

CE-ATA is a hard drive interface that is optimized for embedded applications storage. The device is layered on the top of the MMC protocol stack using the same physical interface. The interface electrical and signaling definition is defined like that in the MMC specification. Refer to the CE-ATA specification for more details.

53.2.2 SDHC block diagram

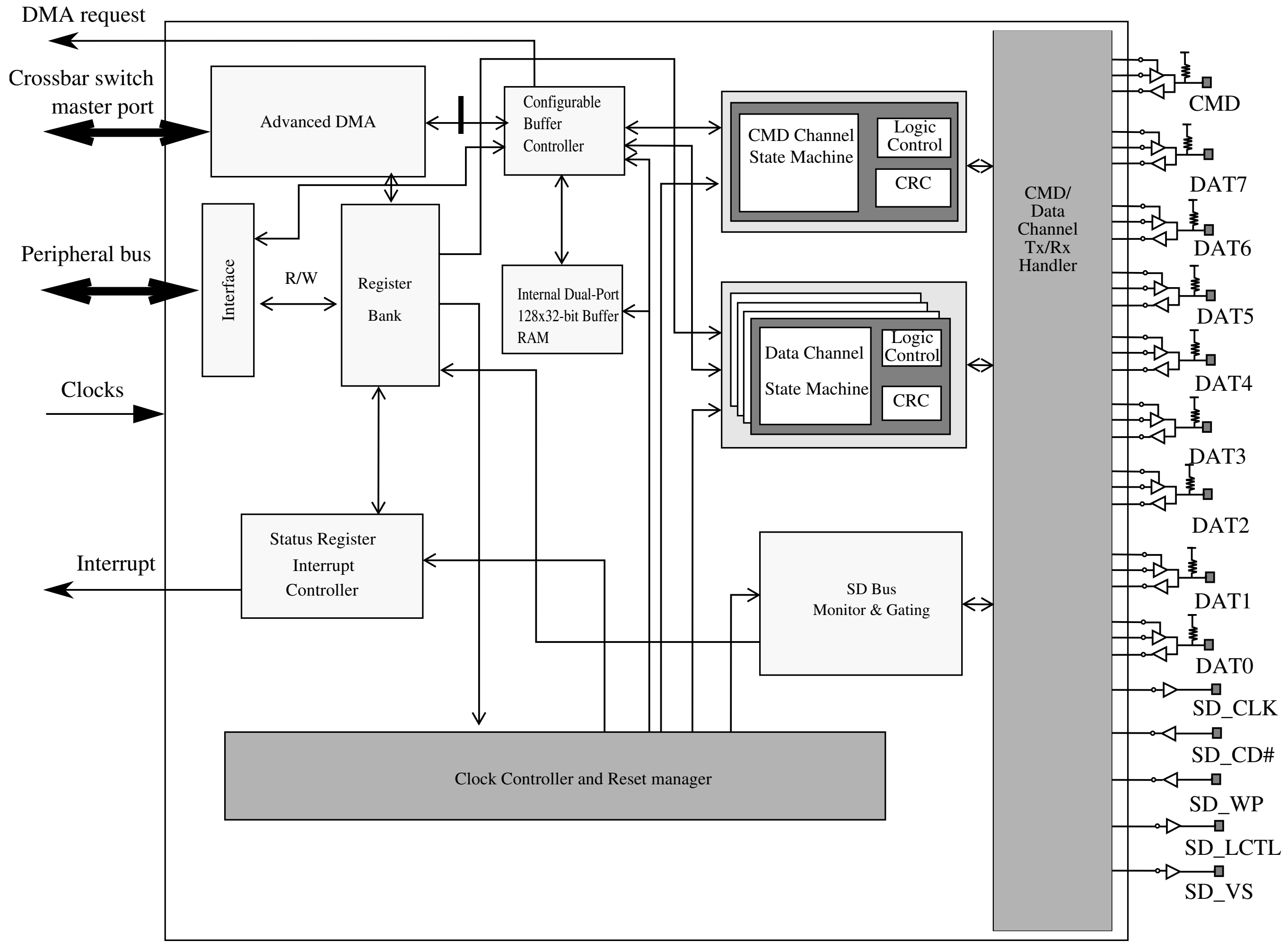

DMA request
Crossbar switch master port
Peripheral bus
Clocks
Interrupt
Advanced DMA
Configurable Buffer Controller
Interface
R/W
Register Bank
Internal Dual-Port 128x32-bit Buffer RAM
CMD Channel State Machine
Logic Control
CRC
Data Channel State Machine
Logic Control
CRC
CMD/ Data Channel Tx/Rx Handler
Status Register Interrupt Controller
SD Bus Monitor & Gating
Clock Controller and Reset manager
CMD
DAT7
DAT6
DAT5
DAT4
DAT3
DAT2
DAT1
DAT0
SD_CLK
SD_CD#
SD_WP
SD_LCTL
SD_VS

Figure 53-2. Enhanced secure digital host controller block diagram

53.2.3 Features

The features of the SDHC module include the following:

- Conforms to the SD Host Controller Standard Specification version 2.0 including test event register support
- Compatible with the MMC System Specification version 4.2/4.3
- Compatible with the SD Memory Card Specification version 2.0 and supports the high capacity SD memory card
- Compatible with the SDIO Card Specification version 2.0
- Compatible with the CE-ATA Card Specification version 1.0

- Designed to work with CE-ATA, SD memory, miniSD memory, SDIO, miniSDIO, SD Combo, MMC, MMC plus, and MMC RS cards
- Card bus clock frequency up to 52 MHz
- Supports 1-bit / 4-bit SD and SDIO modes, 1-bit / 4-bit / 8-bit MMC modes, 1-bit / 4-bit / 8-bit CE-ATA devices
    - Up to 200 Mbps of data transfer for SD/SDIO cards using 4 parallel data lines
    - Up to 416 Mbps of data transfer for MMC cards using 8 parallel data lines in SDR (single data rate) mode
- Supports 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 (both hardware and software CMD12)
- 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 128x32-bit FIFO for read/write data
- Supports internal and external DMA capabilities
- Supports advanced DMA to perform linked memory access

53.2.4 Modes and operations

The SDHC can select the following modes for data transfer:

- SD 1-bit
- SD 4-bit
- MMC 1-bit

- MMC 4-bit
- MMC 8-bit
- CE-ATA 1-bit
- CE-ATA 4-bit
- CE-ATA 8-bit
- Identification mode (up to 400 kHz)
- MMC full speed mode (up to 20 MHz)
- MMC high speed mode (up to 52 MHz)
- SD/SDIO full speed mode (up to 25 MHz)
- SD/SDIO high speed mode (up to 50 MHz)

53.3 SDHC signal descriptions

Table 53-1. SDHC signal descriptions

Signal	Description	I/O
SDHC_DCLK	Generated clock used to drive the MMC, SD, SDIO or CE-ATA cards.	O
SDHC_CMD	Send commands to and receive responses from the card.	I/O
SDHC_D0	DAT0 line or busy-state detect	I/O
SDHC_D1	8-bit mode: DAT1 line 4-bit mode: DAT1 line or interrupt detect 1-bit mode: Interrupt detect	I/O
SDHC_D2	4-/8-bit mode: DAT2 line or read wait 1-bit mode: Read wait	I/O
SDHC_D3	4-/8-bit mode: DAT3 line or configured as card detection pin 1-bit mode: May be configured as card detection pin	I/O
SDHC_D4	DAT4 line in 8-bit mode Not used in other modes	I/O
SDHC_D5	DAT5 line in 8-bit mode Not used in other modes	I/O

Table continues on the next page...

Table 53-1. SDHC signal descriptions (continued)

Signal	Description	I/O
SDHC_D6	DAT6 line in 8-bit mode Not used in other modes	I/O
SDHC_D7	DAT7 line in 8-bit mode Not used in other modes	I/O

53.4 Memory map and register definition

This section includes the module memory map and detailed descriptions of all registers.

SDHC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400B_1000	DMA System Address Register (SDHC_DSADDR)	32	R/W	0000_0000h	53.4.1/ 1727
400B_1004	Block Attributes Register (SDHC_BLKATTR)	32	R/W	0000_0000h	53.4.2/ 1728
400B_1008	Command Argument Register (SDHC_CMDARG)	32	R/W	0000_0000h	53.4.3/ 1729
400B_100C	Transfer Type Register (SDHC_XFERTYP)	32	R/W	0000_0000h	53.4.4/ 1730
400B_1010	Command Response 0 (SDHC_CMDRSP0)	32	R	0000_0000h	53.4.5/ 1734
400B_1014	Command Response 1 (SDHC_CMDRSP1)	32	R	0000_0000h	53.4.6/ 1735
400B_1018	Command Response 2 (SDHC_CMDRSP2)	32	R	0000_0000h	53.4.7/ 1735
400B_101C	Command Response 3 (SDHC_CMDRSP3)	32	R	0000_0000h	53.4.8/ 1735
400B_1020	Buffer Data Port Register (SDHC_DATPORT)	32	R/W	0000_0000h	53.4.9/ 1737
400B_1024	Present State Register (SDHC_PRSSTAT)	32	R	0000_0000h	53.4.10/ 1737
400B_1028	Protocol Control Register (SDHC_PROCTL)	32	R/W	0000_0020h	53.4.11/ 1742
400B_102C	System Control Register (SDHC_SYSCTL)	32	R/W	0000_8008h	53.4.12/ 1746

Table continues on the next page...

SDHC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400B_1030	Interrupt Status Register (SDHC_IRQSTAT)	32	R/W	0000_0000h	53.4.13/ 1749
400B_1034	Interrupt Status Enable Register (SDHC_IRQSTATEN)	32	R/W	117F_013Fh	53.4.14/ 1755
400B_1038	Interrupt Signal Enable Register (SDHC_IRQSIGEN)	32	R/W	0000_0000h	53.4.15/ 1758
400B_103C	Auto CMD12 Error Status Register (SDHC_AC12ERR)	32	R	0000_0000h	53.4.16/ 1760
400B_1040	Host Controller Capabilities (SDHC_HTCAPBLT)	32	R	07F3_0000h	53.4.17/ 1763
400B_1044	Watermark Level Register (SDHC_WML)	32	R/W	0010_0010h	53.4.18/ 1765
400B_1050	Force Event Register (SDHC_FEVT)	32	W (always reads zero)	0000_0000h	53.4.19/ 1766
400B_1054	ADMA Error Status Register (SDHC_ADMAES)	32	R	0000_0000h	53.4.20/ 1768
400B_1058	ADMA System Address Register (SDHC_ADSADDR)	32	R/W	0000_0000h	53.4.21/ 1770
400B_10C0	Vendor Specific Register (SDHC_VENDOR)	32	R/W	0000_0001h	53.4.22/ 1771
400B_10C4	MMC Boot Register (SDHC_MMCBOOT)	32	R/W	0000_0000h	53.4.23/ 1772
400B_10FC	Host Controller Version (SDHC_HOSTVER)	32	R	0000_1201h	53.4.24/ 1773

53.4.1 DMA System Address Register (SDHC_DSADDR)

This register contains the physical system memory address used for DMA transfers.

Address: SDHC_DSADDR is 400B_1000h base + 0h offset = 400B_1000h

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	DSADDR																														0	
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

SDHC_DSADDR field descriptions

Field	Description
31–2 DSADDR	DMA System Address

Table continues on the next page...

SDHC_DSADDR field descriptions (continued)

Field	Description
	This register contains the 32-bit system memory address for a DMA transfer. Since the address must be word (4 bytes) align, the least 2 bits are reserved, always 0. When the SDHC stops a DMA transfer, this register points to the system address of the next contiguous data position. It can be accessed only when no transaction is executing (i.e. after a transaction has stopped). Read operation during transfers may return an invalid value. The host driver shall initialize 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 shall wait, until the PRSSTAT[DLA] is cleared, before writing to this register. The SDHC 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, SDHC will automatically change SEQ burst type to NSEQ. Since this register supports dynamic address reflecting, when IRQSTAT[TC] bit is set, it automatically alters the value of internal address counter, so SW cannot change this register when IRQSTAT[TC] bit is set.
1–0 Reserved	This read-only field is reserved and always has the value zero.

53.4.2 Block Attributes Register (SDHC_BLKATTR)

This register is used to configure the number of data blocks and the number of bytes in each block.

Address: SDHC_BLKATTR is 400B_1000h base + 4h offset = 400B_1004h

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	BLKCNT																0			BLKSIZE												
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

SDHC_BLKATTR field descriptions

Field	Description
31–16 BLKCNT	Blocks Count For Current Transfer This register is enabled when the XFERTYP[BCEN] is set to 1 and is valid only for multiple block transfers. For single block transfer, this register will always read as 1. The host driver shall set this register to a value between 1 and the maximum block count. The SDHC decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to 0 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 as a result 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 suspend command is sent out, SDHC will regard the current transfer is aborted and change BLKCNT back to its original value instead of keeping the dynamical indicator of remained block count.

Table continues on the next page...

SDHC_BLKATTR field descriptions (continued)

Field	Description
	When restoring transfer content prior to issuing a resume command, the host driver shall restore 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 XFERTYP[MSBSEL] bit is 0, indicating a single block transfer, the read value of BLKCNT is always 1. 0000h Stop count 0001h 1 block 0002h 2 blocks ... FFFFh 65535 blocks
15–13 Reserved	This read-only field is reserved and always has the value zero.
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 will be ignored. 000h No data transfer 001h 1 Byte 002h 2 Bytes 003h 3 Bytes 004h 4 Bytes ... 1FFh 511 Bytes 200h 512 Bytes ... 800h 2048 Bytes ... 1000h 4096 Bytes

53.4.3 Command Argument Register (SDHC_CMDARG)

This register contains the SD/MMC command argument.

Address: SDHC_CMDARG is 400B_1000h base + 8h offset = 400B_1008h

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	CMDARG																															
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

SDHC_CMDARG field descriptions

Field	Description
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 PRSSTAT[CDIHB0] bit is set.

53.4.4 Transfer Type Register (SDHC_XFERTYP)

This register is used to control the operation of data transfers. The host driver shall set this register before issuing a command followed by a data transfer, or before issuing a resume command. To prevent data loss, the SDHC prevents writing to the bits, that 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 shall check the PRSSTAT[CDIHB] and the PRSSTAT[CIHB] before writing to this register. When the PRSSTAT[CDIHB] is set, any attempt to send a command with data by writing to this register is ignored; when the PRSSTAT[CIHB] bit is set, any write to this register is ignored.

On sending commands with data transfer invovled, it is mandatory that the block size is non-zero. Besides, block count must also be non-zero, or indicated as 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 SDHC will ignore 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 (WPSPL bit of Present State Register is '1), otherwise SDHC will also ignore the command.

If the commands with data transfer does not receive the response in 64 clock cycles, i.e., response time-out, SDHC will regard the external device does not accept the command and abort the data transfer. In this scenario, the driver should issue the command again to re-try the transfer. It is also possible that for some reason the card responds the command but SDHC 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 following table shows the summary of how register settings determine the type of data transfer.

Table 53-7. Transfer Type Register Setting for Various Transfer Types

Multi/Single block select	Block count enable	Block count	Function
0	Don't care	Don't care	Single transfer

Table continues on the next page...

Table 53-7. Transfer Type Register Setting for Various Transfer Types (continued)

Multi/Single block select	Block count enable	Block count	Function
1	0	Don't care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

The following table shows the relationship between the XFERTYP[CICEN] and XFERTYP[CCCEN], in regards to the XFERTYP[RSPTYP] as well as the name of the response type.

Table 53-8. Relationship Between Parameters and the Name of the Response Type

Response type (RSPTYP)	Index check enable (CICEN)	CRC check enable (CCCEN)	Name of response type
00	0	0	No Response
01	0	1	IR2
10	0	0	R3,R4
10	1	1	R1,R5,R6
11	1	1	R1b,R5b

NOTE

- 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 the SDHC will check the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command shall be used with R5b.
- The CRC field for R3 and R4 is expected to be all 1 bits. The CRC check shall be disabled for these response types.

Address: SDHC_XFERTYP is 400B_1000h base + Ch offset = 400B_100Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		CMDINX						CMDTYP		DPSEL	CICEN	CCCEN	0	RSPTYP	
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										MSBSEL	DTDSEL	0	AC12EN	BCEN	DMAEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDHC_XFERTYP field descriptions

Field	Description
31–30 Reserved	This read-only field is reserved and always has the value zero.
29–24 CMDINX	Command Index These bits shall be 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 shall be set to 00b for all other commands. • Suspend command: If the suspend command succeeds, the SDHC shall assume that the card bus has been released and that it is possible to issue the next command which uses the DAT line. Since the SDHC 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 the SDHC that a suspend command was successfully issued. After the end bit of command is sent, the SDHC de-asserts read wait for read transactions and stops checking busy for write transactions. In 4-bit mode, the interrupt cycle starts. If the suspend command fails, the SDHC will maintain its current state, and the host driver shall restart the transfer by setting the PROCTL[CREQ]. • 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 SDHC will check for a pending busy state before starting write transfers. • Abort command: If this command is set when executing a read transfer, the SDHC will stop reads to the buffer. If this command is set when executing a write transfer, the SDHC will stop driving the DAT 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

Table continues on the next page...

SDHC_XFERTYP field descriptions (continued)

Field	Description
	This bit is set to 1 to indicate that data is present and shall be transferred using the DAT 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 DAT[0] line (R1b or R5b, for example: CMD38). **NOTE:** In resume command, this bit shall be set, and other bits in this register shall be set the same as when the transfer was initially launched. When the Write Protect switch is on, (i.e. the WPSPL bit is active as ‘0’), any command with a write operation will be ignored. That is to say, when this bit is set, while the DTDSEL bit 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 bit is set to 1, the SDHC will check 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 bit is set to 0, the index field is not checked. 0b Disable 1b Enable
19 CCCEN	Command CRC Check Enable If this bit is set to 1, the SDHC shall check the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this bit 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. 0b Disable 1b Enable
18 Reserved	This read-only field is reserved and always has the value zero.
17–16 RSPTYP	Response Type Select 00b No response 01b Response length 136 10b Response length 48 11b Response length 48, check busy after response
15–6 Reserved	This read-only field is reserved and always has the value zero.
5 MSBSEL	Multi/Single Block Select This bit enables multiple block DAT line data transfers. For any other commands, this bit shall be set to 0. If this bit is 0, it is not necessary to set the block count register. 0b Single block 1b Multiple blocks
4 DTDSEL	Data Transfer Direction Select This bit defines the direction of DAT line data transfers. The bit is set to 1 by the host driver to transfer data from the SD card to the SDHC and is set to 0 for all other commands.

Table continues on the next page...

SDHC_XFERTYP field descriptions (continued)

Field	Description
	0b Write (host to card) 1b Read (card to host)
3 Reserved	This read-only field is reserved and always has the value zero.
2 AC12EN	Auto CMD12 Enable Multiple block transfers for memory require a CMD12 to stop the transaction. When this bit is set to 1, the SDHC will issue a CMD12 automatically when the last block transfer has completed. The host driver shall not set this bit 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, the SDHC will ignore this bit no matter if it is set or not. 0b Disable 1b Enable
1 BCEN	Block Count Enable This bit is used to enable the Block Count register, which is only relevant for multiple block transfers. When this bit 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 bit enables DMA functionality. If this bit is set to 1, a DMA operation shall begin when the host driver sets the DPSEL bit of this register. Whether the simple DMA, or the advanced DMA, is active depends on the PROCTL[DMAS]. 0b Disable 1b Enable

53.4.5 Command Response 0 (SDHC_CMDRSP0)

This register is used to store part 0 of the response bits from the card.

Address: SDHC_CMDRSP0 is 400B_1000h base + 10h offset = 400B_1010h

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	CMDRSP0																															
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

SDHC_CMDRSP0 field descriptions

Field	Description
31–0 CMDRSP0	Command Response 0

53.4.6 Command Response 1 (SDHC_CMDRSP1)

This register is used to store part 1 of the response bits from the card.

Address: SDHC_CMDRSP1 is 400B_1000h base + 14h offset = 400B_1014h

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	CMDRSP1																															
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

SDHC_CMDRSP1 field descriptions

Field	Description
31–0 CMDRSP1	Command Response 1

53.4.7 Command Response 2 (SDHC_CMDRSP2)

This register is used to store part 2 of the response bits from the card.

Address: SDHC_CMDRSP2 is 400B_1000h base + 18h offset = 400B_1018h

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	CMDRSP2																															
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

SDHC_CMDRSP2 field descriptions

Field	Description
31–0 CMDRSP2	Command Response 2

53.4.8 Command Response 3 (SDHC_CMDRSP3)

This register is used to store part 3 of the response bits from the card.

The following table describes the mapping of command responses from the SD bus to command response registers for each response type. In the table, R[ ] refers to a bit range within the response data as transmitted on the SD bus.

Table 53-13. 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-bit 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-bit 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, the SDHC only stores part of the response data in the command response registers. This enables the host driver to efficiently read 32-bit 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 the SDHC (as specified by the XFERTYP[CICEN] and the XFERTYP[CCCEN] bits) 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, the SDHC will check R[47:1], and if the response length is 136 the SDHC will check R[119:1].

Since the SDHC may have a multiple block data transfer executing concurrently with a CMD_wo_DAT command, the SDHC stores the auto CMD12 response in the CMDRSP3 register. The CMD_wo_DAT response is stored in CMDRSP0. This allows the SDHC to avoid overwriting the Auto CMD12 response with the CMD_wo_DAT and vice versa. When the SDHC modifies part of the command response registers, as shown in the table above, it preserves the unmodified bits.

Address: SDHC_CMDRSP3 is 400B_1000h base + 1Ch offset = 400B_101Ch

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	CMDRSP3																															
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

SDHC_CMDRSP3 field descriptions

Field	Description
31–0 CMDRSP3	Command Response 3

53.4.9 Buffer Data Port Register (SDHC_DATPORT)

This is a 32-bit data port register used to access the internal buffer and it can not be updated in idle mode.

Address: SDHC_DATPORT is 400B_1000h base + 20h offset = 400B_1020h

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	DATCONT																															
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

SDHC_DATPORT field descriptions

Field	Description
31–0 DATCONT	Data Content The Buffer Data Port register is for 32-bit data access by the CPU or the external DMA. When the internal DMA is enabled, any write to this register is ignored, and any read from this register will always yield 0s.

53.4.10 Present State Register (SDHC_PRSSTAT)

The host driver can get status of the SDHC from this 32-bit read only register.

NOTE

> The host driver can issue CMD0, CMD12, CMD13 (for memory) and CMD52 (for SDIO) when the DAT lines are busy during a data transfer. These commands can be issued when Command Inhibit (CIHB) is set to zero. Other commands shall be issued when Command Inhibit (CDIHB) is set to zero. Possible changes to the SD Physical Specification may add other commands to this list in the future.

Address: SDHC_PRSSTAT is 400B_1000h base + 24h offset = 400B_1024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLSL								CLSL	0						CINS
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				BREN	BWEN	RTA	WTA	SDOFF	PEROFF	HCKOFF	IPGOFF	SDSTB	DLA	CDIHB	CIHB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDHC_PRSSTAT field descriptions

Field	Description
31–24 DLSL	DAT Line Signal Level This status is used to check the DAT line level to recover from errors, and for debugging.This is especially useful in detecting the busy signal level from DAT[0]. The reset value is effected by the external pullup/pulldown resistors. By default, the read value of this bit field after reset is 8’b11110111, when DAT[3] is pulled down and the other lines are pulled up. DAT[0] Data 0 line signal level DAT[1] Data 1 line signal level DAT[2] Data 2 line signal level DAT[3] Data 3 line signal level DAT[4] Data 4 line signal level DAT[5] Data 5 line signal level DAT[6] Data 6 line signal level DAT[7] 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 effected by the external pullup/pulldown resistor, by default, the read value of this bit after reset is 1b, when the command line is pulled up.
22–17 Reserved	This read-only field is reserved and always has the value zero.
16 CINS	Card Inserted This bit indicates whether a card has been inserted. The SDHC debounces this signal so that the host driver will 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 effect this bit.

Table continues on the next page...

SDHC_PRSSTAT field descriptions (continued)

Field	Description
	The SYSCTL[RSTA] does not effect this bit.A software reset does not effect this bit. 0b Power on reset or no card 1b Card inserted
15–12 Reserved	This read-only field is reserved and always has the value zero.
11 BREN	Buffer Read Enable This status bit is used for non-DMA read transfers. The SDHC may implement multiple buffers to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this bit is high, valid data greater than the watermark level exist in the buffer. This read only flag indicates that valid data exists in the host side buffer. 0b Read disable, valid data less than the watermark level exist in the buffer. 1b Read enable, valid data greater than the watermark level exist in the buffer.
10 BWEN	Buffer Write Enable This status bit is used for non-DMA write transfers. The SDHC can implement multiple buffers to transfer data efficiently. This read only flag indicates if space is available for write data. If this bit is 1, valid data greater than the watermark level can be written to the buffer.This read only flag indicates if space is available for write data. 0b Write disable, the buffer can hold valid data less than the write watermark level. 1b Write enable, the buffer can hold valid data greater than the write watermark level.
9 RTA	Read Transfer Active This status bit is used for detecting completion of a read transfer. This bit is set for either of the following conditions: • After the end bit of the read command. • When writing a 1 to the PROCTL[CREQ] to restart a read transfer. A transfer complete interrupt is generated when this bit changes to 0. This bit 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 SDHC internal buffer. • When all valid data blocks have been transferred from SDHC internal buffer to the system and no current block transfers are being sent as a result 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 bit indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the SDHC. This bit is set in either of the following cases: • After the end bit of the write command. • When writing 1 to the PROCTL[CREQ] to restart a write transfer. This bit 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.

Table continues on the next page...

SDHC_PRSSTAT field descriptions (continued)

Field	Description
	During a write transaction, a block gap event interrupt is generated when this bit 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 bit 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 has cleared SYSCTL[SDCLKEN] bit to stop the SD clock. This bit is for the host driver to debug data transaction on the SD bus. 0b SD clock is active 1b SD clock is gated off
6 PEROFF	SDHC clock Gated Off Internally This status bit indicates that the SDHC clock is internally gated off. This bit is for the host driver to debug transaction on the SD bus. When INITA bit is set, SDHC sending 80 clock cycles to the card, the SDCLKEN bit must be ‘1’ to enable the output card clock, otherwise the SDHC clock will never be gate off, so SDHC clock and bus clock will be always active. 0b SDHC clock is active 1b SDHC clock is gated off
5 HCKOFF	System Clock Gated Off Internally This status bit indicates that the system clock is internally gated off. This bit is for the host driver to debug during a data transfer. 0b System clock is active 1b System clock is gated off
4 IPGOFF	Bus Clock Gated Off Internally This status bit indicates that the bus clock is internally gated off. This bit is for the host driver to debug.

Table continues on the next page...

SDHC_PRSSTAT field descriptions (continued)

Field	Description
	0b Bus clock is active 1b Bus clock is gated off
3 SDSTB	SD Clock Stable This status bit indicates that the internal card clock is stable. This bit is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear SYSCTL[SDCLKEN] bit to remove glitch on the card clock when the frequency is changing. 0b Clock is changing frequency and not stable 1b Clock is stable
2 DLA	Data Line Active This status bit indicates whether one of the DAT 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 bit will be set in either of the following cases: • After the end bit of the read command. • When writing a 1 to the PROCTL[CREQ] to restart a read transfer. This bit will be cleared in either of the following cases: 1. When the end bit of the last data block is sent from the SD bus to the SDHC. 2. When the read wait state is stopped by a suspend command and the DAT2 line is released. The SDHC will wait 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), the SDHC can wait for a current block gap by continuing to drive the read wait signal. It is necessary to support read wait in order to use the suspend / resume function. This bit will remain 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. This bit will be set in either of the following cases: • After the end bit of the write command. • When writing to 1 to the PROCTL[CREQ] to continue a write transfer. This bit will be cleared in either of the following cases: • When the SD card releases write busy of the last data block, the SDHC will also detect if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, the SDHC shall assume the card drive “Not busy”. • When the SD card releases write busy, prior to waiting for write transfer, and as a result 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 bit will be cleared when the DAT0 line is released. 0b DAT line inactive 1b DAT line active
1 CDIHB	Command Inhibit (DAT)

Table continues on the next page...

SDHC_PRSSTAT field descriptions (continued)

Field	Description
	This status bit is generated if either the DLA or the RTA is set to 1. If this bit is 0, it indicates that the SDHC can issue the next SD/MMC Command. Commands with a busy signal belong to CDIHB (e.g. R1b, R5b type). Except in the case when the command busy is finished, 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 bit has changed from 1 to 0. 0b Can issue command which uses the DAT line 1b Cannot issue command which uses the DAT line
0 CIHB	Command Inhibit (CMD) If this status bit is 0, it indicates that the CMD line is not in use and the SDHC can issue a SD/MMC Command using the CMD line. This bit is set also immediately after the transfer type register is written. This bit is cleared when the command response is received. Even if the CDIHB bit is set to 1, Commands using only the CMD line can be issued if this bit is 0. Changing from 1 to 0 generates a command complete interrupt in the interrupt status register. If the SDHC cannot issue the command because of a command conflict error (Refer to command CRC error) or because of a command not issued by auto CMD12 error, this bit will remain 1 and the command complete is not set. The status of issuing an auto CMD12 does not show on this bit. 0b Can issue command using only CMD line 1b Cannot issue command

53.4.11 Protocol Control Register (SDHC_PROCTL)

There are three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether the SDHC issues a suspend command or the SD card accepts the suspend command.

1. If the host driver does not issue a suspend command, the continue request shall be used to restart the transfer.
2. If the host driver issues a suspend command and the SD card accepts it, a resume command shall be used to restart the transfer.
3. If the host driver issues a suspend command and the SD card does not accept it, the continue request shall be used to restart the transfer.

Any time stop at block gap request stops the data transfer, the host driver shall wait 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 shall clear the stop at block gap request before or simultaneously.

Address: SDHC_PROCTL is 400B_1000h base + 28h offset = 400B_1028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					WECRM	WECINS	WECINT	0				IABG	RWCTL	CREQ	SABGREQ
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						DMAS		CDSS	CDTL	EMODE		D3CD	DTW		LCTL
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

SDHC_PROCTL field descriptions

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value zero.
26 WECRM	Wakeup Event Enable On SD Card Removal This bit enables a wakeup event, via IRQSTAT[CRM]. FN_WUS (Wake Up Support) in CIS does not effect this bit. When this bit is set, the IRQSTAT[CRM] and the SDHC interrupt can be asserted without SD_CLK toggling. When the wakeup feature is not enabled, the SD_CLK must be active in order to assert the IRQSTAT[CRM] and the SDHC interrupt. 0b Disabled 1b Enabled
25 WECINS	Wakeup Event Enable On SD Card Insertion This bit enables a wakeup event, via IRQSTAT[CINS]. FN_WUS (Wake Up Support) in CIS does not effect this bit. When this bit is set, the IRQSTATEN[CINSEN] and the SDHC interrupt can be asserted without SD_CLK toggling. When the wakeup feature is not enabled, the SD_CLK must be active in order to assert the IRQSTATEN[CINSEN] and the SDHC interrupt. 0b Disabled 1b Enabled
24 WECINT	Wakeup Event Enable On Card Interrupt This bit enables a wakeup event, via IRQSTAT[CINT]. This bit can be set to 1 if FN_WUS (Wake Up Support) in CIS is set to 1. When this bit is set, the card interrupt status and the SDHC interrupt can be asserted without SD_CLK toggling. When the wakeup feature is not enabled, the SD_CLK must be active in order to assert the card interrupt status and the SDHC interrupt. 0b Disabled 1b Enabled
23–20 Reserved	This read-only field is reserved and always has the value zero.
19 IABG	Interrupt At Block Gap

Table continues on the next page...

SDHC_PROCTL field descriptions (continued)

Field	Description
	This bit 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 can't signal an interrupt during a multiple block transfer, this bit should be set to 0 to avoid an inadvertent interrupt. When the host driver detects an SDIO card insertion, it shall set this bit according to the CCCR of the card. 0b Disabled 1b Enabled
18 RWCTL	Read Wait Control The read wait function is optional for SDIO cards. If the card supports read wait, set this bit to enable use of the read wait protocol to stop read data using the DAT[2] line. Otherwise the SDHC has to stop the SD Clock to hold read data, which restricts commands generation. When the host driver detects an SDIO card insertion, it shall set this bit according to the CCCR of the card. If the card does not support read wait, this bit shall never be set to 1, otherwise DAT line conflicts may occur. If this bit is set to 0, stop at block gap during read operation is also supported, but the SDHC will stop the SD Clock to pause reading operation. 0b Disable read wait control, and stop SD clock at block gap when SABGREQ bit is set. 1b Enable read wait control, and assert read wait without stopping SD clock at block gap when SABGREQ bit is set.
17 CREQ	Continue Request This bit is used to restart a transaction which was stopped using the PROCTL[SABGREQ]. When a suspend operation is not accepted by the card, it is also by setting this bit to restart the paused transfer. To cancel stop at the block gap, set PROCTL[SABGREQ] to 0 and set this bit to 1 to restart the transfer. The SDHC automatically clears this bit, therefore it is not necessary for the host driver to set this bit to 0. If both PROCTL[SABGREQ] and this bit are 1, the continue request is ignored. 0b No effect 1b Restart
16 SABGREQ	Stop At Block Gap Request This bit is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the IRQSTATEN[TCSEN] is set to 1, indicating a transfer completion, the host driver shall leave this bit set to 1. Clearing both the PROCTL[SABGREQ] and PROCTL[CREQ] does not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The SDHC will honor the PROCTL[SABGREQ] for write transfers, but for read transfers it requires that the SDIO card support read wait. Therefore, the host driver shall not set this bit during read transfers unless the SDIO card supports read wait and has set the PROCTL[RWCTL] to 1, otherwise the SDHC will stop 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 shall set this bit after all block data is written. If this bit is set to 1, the host driver shall not write data to the data port register after a block is sent. Once this bit is set, the host driver shall not clear this bit before the IRQSTATEN[TCSEN] is set, otherwise the SDHC's behavior is undefined. This bit effects PRSSTAT[RTA], PRSSTAT[WTA], PRSSTAT[CDIHB]. 0b Transfer 1b Stop
15–10 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

SDHC_PROCTL field descriptions (continued)

Field	Description
9–8 DMAS	DMA Select This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation. 00 No DMA or simple DMA is selected 01 ADMA1 is selected 10 ADMA2 is selected 11 Reserved
7 CDSS	Card Detect Signal Selection This bit 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 is bit is enabled while the CDSS 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 SDHC supports all four endian modes in data transfer. 00b Big endian mode 01b Half word big endian mode 10b Little endian mode 11b Reserved
3 D3CD	DAT3 as Card Detection Pin If this bit is set, DAT3 should be pulled down to act as a card detection pin. Be cautious when using this feature, because DAT3 is also a chip-select for the SPI mode. A pulldown on this pin and CMD0 may set the card into the SPI mode, which the SDHC does not support. Note: Keep this bit set if SDIO interrupt is used. 0b DAT3 does not monitor card Insertion 1b DAT3 as card detection pin
2–1 DTW	Data Transfer Width This bit selects the data width of the SD bus for a data transfer. The host driver shall set 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 bit, 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 bit can be set during

Table continues on the next page...

SDHC_PROCTL field descriptions (continued)

Field	Description
	all these transactions. It is not necessary to change for each transaction. When the software issues multiple SD commands, setting the bit once before the first command is sufficient: it is not necessary to reset the bit between commands. 0b LED off 1b LED on

53.4.12 System Control Register (SDHC_SYSCTL)

Address: SDHC_SYSCTL is 400B_1000h base + 2Ch offset = 400B_102Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				INITA	0	0	0	0				DTOCV			
W						RSTD	RSTC	RSTA								
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	SDCLKFS								DVS				SDCLKEN	PEREN	HCKEN	IPGEN
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

SDHC_SYSCTL field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value zero.
27 INITA	Initialization Active When this bit is set, 80 SD-clocks are sent to the card. After the 80 clocks are sent, this bit is self cleared. This bit 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 bit is already 1 has no effect. Writing 0 to this bit at any time has no effect. When either of the PRSSTAT[CIHB] and PRSSTAT[CDIHB] bits are set, writing 1 to this bit is ignored (i.e. when command line or data lines are active, write to this bit is not allowed). On the otherhand, when this bit is set, i.e., during intialization active period, it is allowed to issue command, and the command bit stream will appear 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 send 80 cycles to the card and does not want to wait till this bit is self cleared.
26 RSTD	Software Reset For DAT Line Only part of the data circuit is reset. DMA circuit is also reset. The following registers and bits are cleared by this bit: • Data port register • Buffer is cleared and initialized.Present State register • Buffer Read Enable

Table continues on the next page...

SDHC_SYSCTL field descriptions (continued)

Field	Description
	• Buffer Write Enable • Read Transfer Active • Write Transfer Active • DAT Line Active • Command Inhibit (DAT) 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 0b No reset 1b Reset
25 RSTC	Software Reset For CMD Line Only part of the command circuit is reset. The following registers and bits are cleared by this bit: • PRSSTAT[CIHB] • IRQSTAT[CC] 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 shall set this bit to 1 to reset the SDHC. The SDHC shall reset this bit 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 bit is set, it is recommended that the host driver reset the external card and re-initialize it. 0b No reset 1b Reset
23–20 Reserved	This read-only field is reserved and always has the value zero.
19–16 DTOCV	Data Timeout Counter Value This value determines the interval by which DAT line timeouts are detected. Refer to the IRQSTAT[DTOE] for information on factors that dictate time-out generation. Time-out clock frequency will be generated by dividing the base clock SDCLK value by this value. The host driver can clear the IRQSTATEN[DTOESEN] to prevent inadvertent time-out events. 0000b SDCLK x 2^{13} 0001b SDCLK x 2^{14} ... 1110b SDCLK x 2^{27} 1111b Reserved
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.

Table continues on the next page...

SDHC_SYSCTL field descriptions (continued)

Field	Description
	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 SDHC clock and the following divisor bits. The frequency of SDCLK is set by the following formula: Clock frequency = (Base clock) / (prescaler x divisor) For example, 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 will yield 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 bit field is 80h, so if the input base clock (SDHC clock) 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 shall never exceed this limit. Only the following settings are allowed: 01h Base clock divided by 2 02h Base clock divided by 4 04h Base clock divided by 8 08h Base clock divided by 16 10h Base clock divided by 32 20h Base clock divided by 64 40h Base clock divided by 128 80h Base clock divided by 256
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 divisor without deterioration of duty cycle. The setting are as following: 0h Divisor by 1 1h Divisor by 2 ... Eh Divisor by 15 Fh Divisor by 16
3 SDCLKEN	SD Clock Enable The host controller shall stop SDCLK when writing this bit to 0. SDCLK frequency can be changed when this bit is 0. Then, the host controller shall maintain the same clock frequency until SDCLK is stopped (stop at SDCLK = 0). If the IRQSTAT[CINS] is cleared, this bit should be cleared by the host driver to save power.
2 PEREN	Peripheral Clock Enable If this bit is set, SDHC clock will always be active and no automatic gating is applied. Thus the SDCLK is active except for when auto gating-off during buffer danger (buffer about to over-run or under-run). When this bit is cleared, the SDHC clock will be automatically off whenever there is no transaction on the SD bus. Since this bit is only a feature enabling bit, clearing this bit does not stop SDCLK immediately. The SDHC clock will be internally gated off, if none of the following factors are met: • The cmd part is reset, or • Data part is reset, or • A soft reset, or • The cmd is about to send, or • Clock divisor is just updated, or

Table continues on the next page...

SDHC_SYSCTL field descriptions (continued)

Field	Description
	• Continue request is just set, or • This bit is set, or • Card insertion is detected, or • Card removal is detected, or • Card external interrupt is detected, or • 80 clocks for initialization phase is ongoing 0b SDHC clock will be internally gated off 1b SDHC clock will not be automatically gated off
1 HCKEN	System Clock Enable If this bit is set, system clock will always be active and no automatic gating is applied. When this bit is cleared, system clock will be automatically off when no data transfer is on the SD bus. 0b System clock will be internally gated off 1b System clock will not be automatically gated off
0 IPGEN	IPG Clock Enable If this bit is set, bus clock will always be active and no automatic gating is applied. The bus clock will be internally gated off, if none of the following factors are met: • The cmd part is reset, or • Data part is reset, or • Soft reset, or • The cmd is about to send, or • Clock divisor is just updated, or • Continue request is just set, or • This bit is set, or • Card insertion is detected, or • Card removal is detected, or • Card external interrupt is detected, or • The SDHC clock is not gated off **NOTE:** The bus clock will not be auto gated off if the SDHC clock is not gated off. So clearing only this bit has no effect unless the PEREN bit is also cleared. 0b Bus clock will be internally gated off 1b Bus clock will not be automatically gated off

53.4.13 Interrupt Status Register (SDHC_IRQSTAT)

An interrupt is generated when the Normal Interrupt Signal Enable is enabled and at least one of the status bits is set to 1. For all bits, 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 bit will be asserted again.

The table below shows the relationship between the CTOE and the CC bits.

Table 53-19. SDHC status for CTOE/CC 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 53-20. SDHC 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 (CCE) and command timeout error (CTOE).

Table 53-21. SDHC status for CCE/CTOE Bit Combinations

Command complete	Command timeout error	Meaning of the status
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Address: SDHC_IRQSTAT is 400B_1000h base + 30h offset = 400B_1030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAE	0			AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W				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

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W								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

SDHC_IRQSTAT field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
28 DMAE	DMA Error Occurs when an Internal DMA transfer has failed. This bit 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. Since any error corrupts the whole data block, the host driver shall re-start the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DSADDR value or from the remaining number of blocks and the block size. 0b No Error 1b Error
27–25 Reserved	This read-only field is reserved and always has the value zero.
24 AC12E	Auto CMD12 Error Occurs when detecting that one of the bits in the Auto CMD12 Error Status register has changed from 0 to 1. This bit 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	This read-only field is reserved and always has the value zero.
22 DEBE	Data End Bit Error Occurs either when detecting 0 at the end bit position of read data, which uses the DAT line, or at the end bit position of the CRC.

Table continues on the next page...

SDHC_IRQSTAT field descriptions (continued)

Field	Description
	0b No Error 1b Error
21 DCE	Data CRC Error Occurs when detecting a CRC error when transferring read data, which uses the DAT line, or when detecting the Write CRC status having a value other than 010. 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 0b No Error 1b Time out
19 CIE	Command Index Error Occurs if a Command Index error occurs in the command response. 0b No Error 1b Error
18 CEBE	Command End Bit Error Occurs when detecting that the end bit of a command response is 0. 0b No Error 1b End Bit Error Generated
17 CCE	Command CRC Error Command CRC Error is generated in two cases. • If a response is returned and the Command Timeout Error is set to 0 (indicating no time-out), this bit is set when detecting a CRC error in the command response. • The SDHC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the SDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then the SDHC shall abort the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error shall also be set to 1 to distinguish CMD line conflict. 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 bit of the command. If the SDHC detects a CMD line conflict, in which case a Command CRC Error shall also be set, this bit shall be set without waiting for 64 SDCLK cycles. This is because the command will be aborted by the SDHC. 0b No Error 1b Time out

Table continues on the next page...

SDHC_IRQSTAT field descriptions (continued)

Field	Description
15–9 Reserved	This read-only field is reserved and always has the value zero.
8 CINT	Card Interrupt This status bit is set when an interrupt signal is detected from the external card. In 1-bit mode, the SDHC will detect 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 bit to 1 can clear this bit, but as the interrupt factor from the SDIO card does not clear, this bit is set again. In order to clear this bit, it is required to reset the interrupt factor from the external card followed by a writing 1 to this bit. 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 factors in the SDIO card and the interrupt signal may not be asserted), write 1 to clear this bit, set the Card Interrupt Signal Enable to 1, and start sampling the interrupt signal again. 0b No Card Interrupt 1b Generate Card Interrupt
7 CRM	Card Removal This status bit is set if the Card Inserted bit in the Present State register changes from 1 to 0. When the host driver writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state may possibly be changed when the host driver clears this bit and the interrupt event may not be generated. When this bit is cleared, it will be set again if no card is inserted. In order to leave it cleared, clear the Card Removal Status Enable bit in Interrupt Status Enable register. 0b Card state unstable or inserted 1b Card removed
6 CINS	Card Insertion This status bit is set if the Card Inserted bit in the Present State register changes from 0 to 1. When the host driver writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state may possibly be changed when the host driver clears this bit and the interrupt event may not be generated. When this bit is cleared, it will be set again if a card is inserted. In order to leave it cleared, clear the Card Inserted Status Enable bit in Interrupt Status Enable register. 0b Card state unstable or removed 1b Card inserted
5 BRR	Buffer Read Ready This status bit is set if the Buffer Read Enable bit, in the Present State register, changes from 0 to 1. Refer to the Buffer Read Enable bit in the Present State register for additional information. 0b Not ready to read buffer 1b Ready to read buffer
4 BWR	Buffer Write Ready This status bit is set if the Buffer Write Enable bit, in the Present State register, changes from 0 to 1. Refer to the Buffer Write Enable bit in the Present State register for additional information.

Table continues on the next page...

SDHC_IRQSTAT field descriptions (continued)

Field	Description
	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 bit will not be set. Instead, the DMAE bit will be set. Either Simple DMA or ADMA finishes data transferring, this bit will be set. 0b No DMA Interrupt 1b DMA Interrupt is generated
2 BGE	Block Gap Event If the PROCTL[SABGREQ] is set, this bit is set when a read or write transaction is stopped at a block gap. If PROCTL[SABGREQ] is not set to 1, this bit is not set to 1. In the case of a read transaction: This bit is set at the falling edge of the DAT line active status (When the transaction is stopped at SD Bus timing). The read wait must be supported in order to use this function. In the case of write transaction: This bit 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 bit is set when a read or write transfer is completed. In the case of a read transaction: This bit 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 PROCTL[SABGREQ] (after valid data has been read to the host system). In the case of a write transaction: This bit is set at the falling edge of the DAT 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 PROCTL[SABGREQ], and the data transfers are completed. (after valid data is written to the SD card and the busy signal released). 0b Transfer not complete 1b Transfer complete
0 CC	Command Complete This bit is set when you receive the end bit of the command response (except Auto CMD12). Refer to the PRSSTAT[CIHB]. 0b Command not complete 1b Command complete

53.4.14 Interrupt Status Enable Register (SDHC_IRQSTATEN)

Setting the bits in this register to 1 enables the corresponding interrupt status to be set by the specified event. If any bit is cleared, the corresponding interrupt status bit is also cleared (i.e. when the bit in this register is cleared, the corresponding bit in interrupt status register is always 0).

NOTE

- Depending on PROCTL[IABG] bit setting, SDHC may be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There will be 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 IRQSTATEN[CTOESEN] and IRQSTATEN[CCESEN] to 1.

Address: SDHC_IRQSTATEN is 400B_1000h base + 34h offset = 400B_1034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAESEN	0			AC12ESEN	0	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN
W																
Reset	0	0	0	1	0	0	0	1	0	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							CINTSEN	CRMSEN	CINSEN	BRRSEN	BWRSEN	DINTSEN	BGESEN	TCSEN	CCSEN
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1

SDHC_IRQSTATEN field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
28 DMAESEN	DMA Error Status Enable 0b Masked 1b Enabled
27–25 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

SDHC_IRQSTATEN field descriptions (continued)

Field	Description
24 AC12ESEN	Auto CMD12 Error Status Enable 0b Masked 1b Enabled
23 Reserved	This read-only field is reserved and always has the value zero.
22 DEBESEN	Data End Bit Error Status Enable 0b Masked 1b Enabled
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
15–9 Reserved	This read-only field is reserved and always has the value zero.
8 CINTSEN	Card Interrupt Status Enable If this bit is set to 0, the SDHC will clear the interrupt request to the system. The card interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The host driver should clear the this bit before servicing the card interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0b Masked 1b Enabled
7 CRMSEN	Card Removal Status Enable

Table continues on the next page...

SDHC_IRQSTATEN field descriptions (continued)

Field	Description
	0b Masked 1b Enabled
6 CINSEN	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 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

53.4.15 Interrupt Signal Enable Register (SDHC_IRQSIGEN)

This register is used to select which interrupt status is indicated to the host system as the interrupt. These status bits all share the same interrupt line. Setting any of these bits to 1 enables interrupt generation. The corresponding status register bit will generate an interrupt when the corresponding interrupt signal enable bit is set.

Address: SDHC_IRQSIGEN is 400B_1000h base + 38h offset = 400B_1038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAEIEN	0			AC12EIEN	0	DEBEIEN	DCEIEN	DTOEIEN	CIEIEN	CEBEIEN	CCEIEN	CTOEIEN
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							CINTIEN	CRMIEN	CINSIEN	BRRIEN	BWRIEN	DINTIEN	BGEIEN	TCIEN	CCIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDHC_IRQSIGEN field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
28 DMAEIEN	DMA Error Interrupt Enable 0b Masked 1b Enabled
27–25 Reserved	This read-only field is reserved and always has the value zero.
24 AC12EIEN	Auto CMD12 Error Interrupt Enable 0b Masked 1b Enabled
23 Reserved	This read-only field is reserved and always has the value zero.
22 DEBEIEN	Data End Bit Error Interrupt Enable 0b Masked 1b Enabled

Table continues on the next page...

SDHC_IRQSIGEN field descriptions (continued)

Field	Description
21 DCEIEN	Data CRC Error Interrupt Enable 0b Masked 1b Enabled
20 DTOEIEN	Data Timeout Error Interrupt Enable 0b Masked 1b Enabled
19 CIEIEN	Command Index Error Interrupt Enable 0b Masked 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
15–9 Reserved	This read-only field is reserved and always has the value zero.
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

Table continues on the next page...

SDHC_IRQSIGEN field descriptions (continued)

Field	Description
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 1b Enabled

53.4.16 Auto CMD12 Error Status Register (SDHC_AC12ERR)

When the AC12ESEN bit in the Status register is set, the host driver shall check this register to identify what kind of error the Auto CMD12 indicated. This register is valid only when the Auto CMD12 Error status bit is set.

The following table shows the relationship between the Auto CMGD12 CRC error and the Auto CMD12 command timeout error.

Table 53-25. 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:

1. When the SDHC is going to issue an auto CMD12.
    - Set bit 0 to 1 if the auto CMD12 can't be issued due to an error in the previous command.
    - Set bit 0 to 0 if the auto CMD12 is issued.

2. At the end bit of an auto CMD12 response.
    - Check errors correspond to bits 1-4.
    - Set bits 1-4 corresponding to detected errors.
    - Clear bits 1-4 corresponding to detected errors.
3. Before reading the auto CMD12 error status bit 7.
    - Set bit 7 to 1 if there is a command that can't be issued.
    - Clear bit 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, bit 7 shall be sampled when the driver is not writing to the command register. So it is suggested to read this register only when the IRQSTAT[AC12E] is set. An Auto CMD12 error interrupt is generated when one of the error bits (0-4) is set to 1. The command not issued by auto CMD12 error does not generate an interrupt.

Address: SDHC_AC12ERR is 400B_1000h base + 3Ch offset = 400B_103Ch

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								CNIBAC12E	0		AC12IE	AC12CE	AC12EBE	AC12TOE	AC12NE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDHC_AC12ERR field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 CNIBAC12E	Command Not Issued By Auto CMD12 Error Setting this bit 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	This read-only field is reserved and always has the value zero.
4 AC12IE	Auto CMD12 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
3 AC12CE	Auto CMD12 CRC Error Occurs when detecting a CRC error in the command response. 0b No CRC error 1b CRC Error met in auto CMD12 Response
2 AC12EBE	Auto CMD12 End Bit Error Occurs when detecting that the end bit of command response is 0 which should be 1. 0b No error 1b End bit error generated
1 AC12TOE	Auto CMD12 Timeout Error Occurs if no response is returned within 64 SDCLK cycles from the end bit of the command. If this bit is set to 1, the other error status bits (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 bit is not set because it is not necessary to issue an auto CMD12. Setting this bit to 1 means the SDHC cannot issue the auto CMD12 to stop a memory multiple block data transfer due to some error. If this bit is set to 1, other error status bits (1-4) have no meaning. 0b Executed 1b Not executed

53.4.17 Host Controller Capabilities (SDHC_HTCAPBLT)

This register provides the host driver with information specific to the SDHC implementation. The value in this register is the power-on-reset value, and does not change with a software reset. Any write to this register is ignored.

Address: SDHC_HTCAPBLT is 400B_1000h base + 40h offset = 400B_1040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					VS18	VS30	VS33	SRS	DMAS	HSS	ADMAS	0	MBL		
W																
Reset	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1

Bit	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

SDHC_HTCAPBLT field descriptions

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value zero.
26 VS18	Voltage Support 1.8 V This bit shall depend on the host system ability. 0b 1.8 V not supported 1b 1.8 V supported
25 VS30	Voltage Support 3.0 V This bit shall depend on the host system ability. 0b 3.0 V not supported 1b 3.0 V supported
24 VS33	Voltage Support 3.3 V This bit shall depend on the host system ability.

Table continues on the next page...

SDHC_HTCAPBLT field descriptions (continued)

Field	Description
	0b 3.3 V not supported 1b 3.3 V supported
23 SRS	Suspend/Resume Support This bit indicates whether the SDHC supports suspend / resume functionality. If this bit is 0, the suspend and resume mechanism, as well as the read Wwait, are not supported, and the host driver shall not issue either suspend or resume commands. 0b Not supported 1b Supported
22 DMAS	DMA Support This bit indicates whether the SDHC 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 bit indicates whether the SDHC 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 bit indicates whether the SDHC supports the ADMA feature. 0b Advanced DMA not supported 1b Advanced DMA supported
19 Reserved	This read-only field is reserved and always has the value zero.
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 the SDHC. The buffer shall transfer block size without wait cycles. 000b 512 bytes 001b 1024 bytes 010b 2048 bytes 011b 4096 bytes
15–0 Reserved	This read-only field is reserved and always has the value zero.

53.4.18 Watermark Level Register (SDHC_WML)

Both write and read watermark levels (FIFO threshold) are configurable. There 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.

Address: SDHC_WML is 400B_1000h base + 44h offset = 400B_1044h

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			0					WRWML								0			0					RDWML							
W																																
Reset	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	1	0	0	0	0

SDHC_WML field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
28–24 Reserved	This read-only field is reserved and always has the value zero.
23–16 WRWML	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	This read-only field is reserved and always has the value zero.
12–8 Reserved	This read-only field is reserved and always has the value zero.
7–0 RDWML	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.

53.4.19 Force Event Register (SDHC_FEVT)

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 bit 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 bit of interrupt status register. A read from this register always results in 0's. In order to change the corresponding status bits in the interrupt status register, make sure to set SYSCTL[IPGEN] so that bus clock is always active.

Forcing a card interrupt will generate a short pulse on the DAT[1] line, and the driver may treat this interrupt as a normal interrupt. The interrupt service routine may skip polling the card interrupt factor as the interrupt is self cleared.

Address: SDHC_FEVT is 400B_1000h base + 50h offset = 400B_1050h

Bit	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
W	CINT	0		DMAE	0			AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
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			0	0	0	0	0
W	0								CNIBAC12E	0		AC12IE	AC12EBE	AC12CE	AC12TOE	AC12NE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDHC_FEVT field descriptions

Field	Description
31 CINT	Force Event Card Interrupt Writing 1 to this bit generates a short low-level pulse on the internal DAT[1] line, as if a self clearing interrupt was received from the external card. If enabled, the CINT bit will be set and the interrupt service routine may treat this interrupt as a normal interrupt from the external card.

Table continues on the next page...

SDHC_FEVT field descriptions (continued)

Field	Description
30–29 Reserved	This field is reserved.
28 DMAE	Force Event DMA Error Forces the DMAE bit of Interrupt Status Register to be set.
27–25 Reserved	This field is reserved.
24 AC12E	Force Event Auto Command 12 Error Forces the IRQSTAT[AC12E] to be set.
23 Reserved	This field is reserved.
22 DEBE	Force Event Data End Bit Error Forces the IRQSTAT[DEBE] bit to be set.
21 DCE	Force Event Data CRC Error Forces the IRQSTAT[DCE] bit to be set.
20 DTOE	Force Event Data Time Out Error Force the IRQSTAT[DTOE] bit to be set.
19 CIE	Force Event Command Index Error Forces the IRQSTAT[CCE] bit to be set.
18 CEBE	Force Event Command End Bit Error Forces the IRQSTAT[CEBE] bit to be set.
17 CCE	Force Event Command CRC Error Forces the IRQSTAT[CCE] bit to be set.
16 CTOE	Force Event Command Time Out Error Forces the IRQSTAT[CTOE] bit to be set.
15–8 Reserved	This field is reserved.
7 CNIBAC12E	Force Event Command Not Executed By Auto Command 12 Error Forces the AC12ERR[CNIBAC12E] bit to be set.
6–5 Reserved	This field is reserved.
4 AC12IE	Force Event Auto Command 12 Index Error Forces the AC12ERR[AC12IE] bit to be set.
3 AC12EBE	Force Event Auto Command 12 End Bit Error Forces the AC12ERR[AC12EBE] bit to be set.

Table continues on the next page...

SDHC_FEVT field descriptions (continued)

Field	Description
2 AC12CE	Force Event Auto Command 12 CRC Error Forces the AC12ERR[AC12CE] bit to be set.
1 AC12TOE	Force Event Auto Command 12 Time Out Error Forces the AC12ERR[AC12TOE] bit to be set.
0 AC12NE	Force Event Auto Command 12 Not Executed Forces the AC12ERR[AC12NE] bit to be set.

53.4.20 ADMA Error Status Register (SDHC_ADMAES)

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 doesn’t 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, since unwritten data may 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 bit of the error descriptor.

Table 53-30. 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
11	ST_TFR (Transfer Data)	Holds the address of the next executable descriptor command

Address: SDHC_ADMAES is 400B_1000h base + 54h offset = 400B_1054h

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												ADMADCE	ADMALME	ADMAES	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDHC_ADMAES field descriptions

Field	Description
31–4 Reserved	This read-only field is reserved and always has the value zero.
3 ADMADCE	ADMA Descritor Error This error occurs when invalid descriptor is fetched by ADMA. 0b No error 1b Error
2 ADMALME	ADMA Length Mismatch Error This error occurs in the following 2 cases:

Table continues on the next page...

SDHC_ADMAES field descriptions (continued)

Field	Description
	- 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 can not 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.

53.4.21 ADMA System Address Register (SDHC_ADSADDR)

This register contains the physical system memory address used for ADMA transfers.

Address: SDHC_ADSADDR is 400B_1000h base + 58h offset = 400B_1058h

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	ADSADDR																														0	
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

SDHC_ADSADDR field descriptions

Field	Description
31–2 ADSADDR	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 shall set 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 shall hold 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. Since this register supports dynamic address reflecting, when TC bit is set, it automatically alters the value of internal address counter, so SW cannot change this register when TC bit is set.
1–0 Reserved	This read-only field is reserved and always has the value zero.

53.4.22 Vendor Specific Register (SDHC_VENDOR)

This register contains the vendor specific control/status register.

Address: SDHC_VENDOR is 400B_1000h base + C0h offset = 400B_10C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				0				INTSTVAL							
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														EXBLKNU	EXTDMAEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

SDHC_VENDOR field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value zero.
27–24 Reserved	This read-only field is reserved and always has the value zero.
23–16 INTSTVAL	Internal State Value Internal state value, reflecting the corresponding state value selected by **Debug Select** field. This field is read-only and write to this field does not have effect.
15–2 Reserved	This read-only field is reserved and always has the value zero.
1 EXBLKNU	Exact block number block read enable for SDIO CMD53 This bit must be set before S/W issues CMD53 multi-block read with exact block number. This bit must not be set if the CMD53 multi-block read is not exact block number. 0 none exact block read. 1 Exact block read for SDIO CMD53.
0 EXTDMAEN	External DMA Request Enable Enable the request to external DMA. When the internal DMA (either simple DMA or advanced DMA) is not in use, and this bit is set, SDHC will send out DMA request when the internal buffer is ready. This bit is particularly useful when transferring data by CPU polling mode, and it is not allowed to send out the external DMA request. By default, this bit is set. 0 In any scenario, SDHC does not send out external DMA request. 1 When internal DMA is not active, the external DMA request will be sent out.

53.4.23 MMC Boot Register (SDHC_MMCBOOT)

This register contains the MMC fast boot control register.

Address: SDHC_MMCBOOT is 400B_1000h base + C4h offset = 400B_10C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BOOTBLKCNT															
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								AUTOSABGEN	BOOTEN	BOOTMODE	BOOTACK	DTOCVACK			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDHC_MMCBOOT field descriptions

Field	Description
31–16 BOOTBLKCNT	The value defines the stop at block gap value of automatic mode. When received card block cnt is equal to BOOTBLKCNT and AUTOSABGEN is 1, then stop at block gap.
15–8 Reserved	This read-only field is reserved and always has the value zero.
7 AUTOSABGEN	When boot, enable auto stop at block gap function. This function will be triggered, and host will stop at block gap when received card block cnt is equal to BOOTBLKCNT.
6 BOOTEN	Boot mode enable 0 Fast boot disable 1 Fast boot enable
5 BOOTMODE	Boot mode select 0 Normal boot 1 Alternative boot
4 BOOTACK	Boot ack mode select 0 No ack 1 Ack
3–0 DTOCVACK	Boot ACK time out counter value. 0000b SDCLK x 2^8 0001b SDCLK x 2^9

Table continues on the next page...

SDHC_MMCBOOT field descriptions (continued)

Field	Description
	0010b SDCLK x 2^10 0011b SDCLK x 2^11 0100b SDCLK x 2^12 0101b SDCLK x 2^13 0110b SDCLK x 2^14 0111b SDCLK x 2^15 ... 1110b SDCLK x 2^22 1111b Reserved

53.4.24 Host Controller Version (SDHC_HOSTVER)

This register contains the vendor host controller version information. All bits are read only and will read the same as the power-reset value.

Address: SDHC_HOSTVER is 400B_1000h base + FCh offset = 400B_10FCh

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="16">0</td><td colspan="8">VVN</td><td colspan="8">SVN</td></tr>
<tr><td>W</td><td colspan="16"></td><td colspan="8"></td><td colspan="8"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
</table>

SDHC_HOSTVER field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–8 VVN	Vendor Version Number These status bits are reserved for the vendor version number. The host driver shall not use this status. 00h Freescale SDHC version 1.0 10h Freescale SDHC version 2.0 11h Freescale SDHC version 2.1 12h Freescale SDHC version 2.2 All others Reserved
7–0 SVN	Specification Version Number These status bits indicate the host controller specification version. 01h SD host specification version 2.0, supports test event register and ADMA. All others Reserved

53.5 Functional description

The following sections provide a brief functional description of the major system blocks, including the data buffer, DMA crossbar switch interface, dual-port memory wrapper, data/command controller, clock & reset manager and clock generator.

53.5.1 Data buffer

The SDHC uses one configurable data buffer, so that data can be transferred between the system bus and the SD card, with an optimized manner to maximize throughput between the two clock domains (that is, the IP peripheral clock, and the master clock). The following diagram illustrates the buffer scheme. The buffer is used as temporary storage for data being transferred between the host system and the card. The watermark levels for read and write are both configurable, and can be any number from 1 to 128 words. The burst lengths for read and write are also configurable, and can be any number from 1 to 31 words.

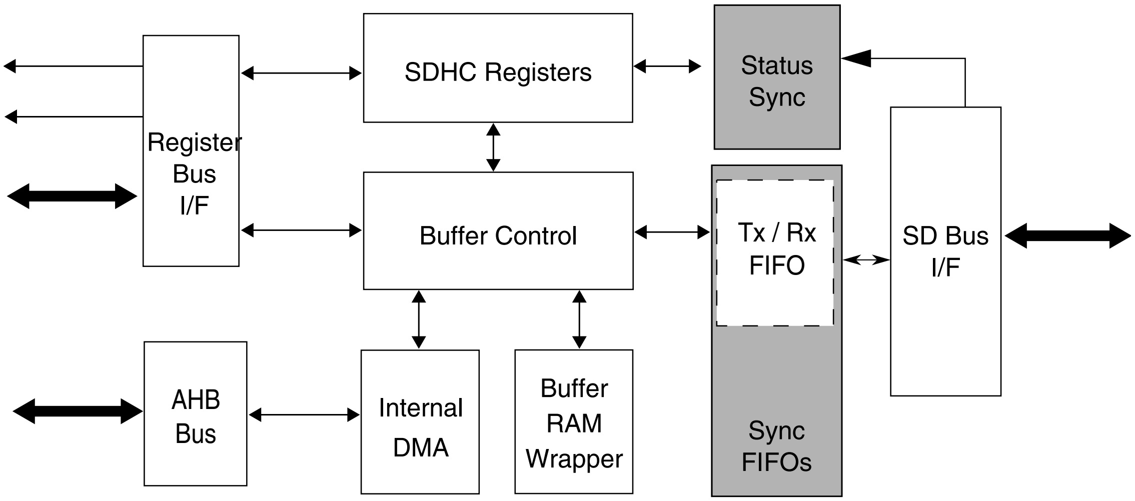


Figure 53-27. SDHC buffer scheme

There are 3 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 RDWML watermark value, then by polling the IRQSTAT[BRR] bit the host driver can read the DATPORT register to fetch the amount of words set in the WML register from the buffer. The write operation is similar.

- External DMA mode:
    - For a read operation, when there are more words received in the buffer than the amount set in the RDWML register, a DMA request is sent out to inform the external DMA to fetch the data. The request will be immediately de-asserted when there is an access on the DATPORT register. If the number of words in the buffer after the current burst meets or exceeds RDWML value, then the DMA request is asserted again. So for instance if there are twice as many words in the buffer than the RDWML value, there are two successive DMA requests with only one cycle of de-assertion between. The write operation is similar.

      Note the accesses CPU polling mode and external DMA mode both use the IP bus, and if the external DMA is enable, in both modes an external DMA request is sent out whenever the buffer is ready.
- Internal DMA mode (includes simple and advanced DMA access's):
    - The internal DMA access, either by simple or advanced DMA, is over the crossbar switch bus. For internal DMA access mode, the external DMA request will never be sent out.

For a read operation, when there are more words in the buffer than the amount set in the WML register, the internal DMA starts fetching data over the crossbar switch bus. Except INCR4 and INCR8, the burst type is always INCR mode and the burst length depends on the shortest of following factors:

1. Burst length configured in the burst length field of the WML register
2. Watermark level boundary
3. Block size boundary
4. Data boundary configured in the current descriptor (if the ADMA is active)
5. 1 KB address boundary

Write operation is similar.

Sequential and contiguous access is necessary to ensure the pointer address value is correct. Random or skipped access is not possible. The byte order, by reset, is little endian mode. The actually byte order is swapped inside the buffer, according to the endian mode configured by software, as illustrated in the following diagrams. For a host write operation, byte order is swapped after data is fetched from the buffer and ready to send to the SD bus. For a host read operation, byte order is swapped before the data is stored into the buffer.

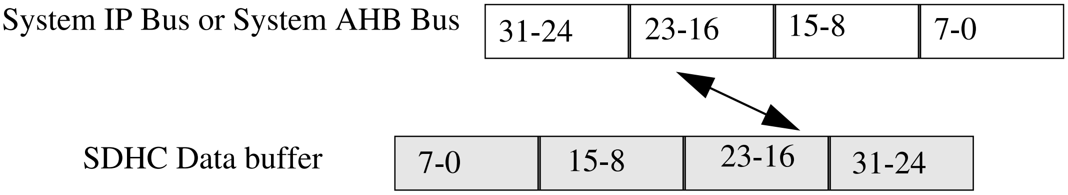


Figure 53-28. Data swap between system bus and SDHC data buffer in byte little endian mode

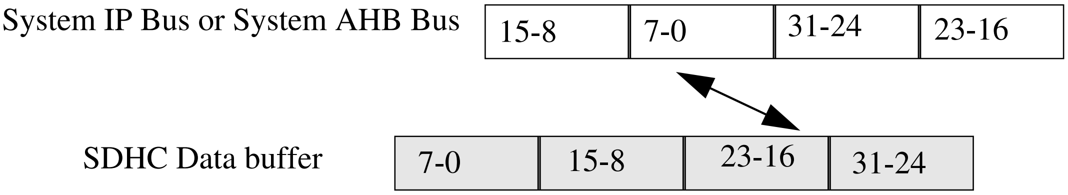


Figure 53-29. Data swap between system bus and SDHC data buffer in half word big endian mode

53.5.1.1 Write operation sequence

There are three ways to write data into the buffer when the user transfers data to the card:

1. By using external DMA through the SDHC DMA request signal.
2. By processor core polling through the IRQSTAT[BWR] bit (interrupt or polling).
3. By using the internal DMA.

When the internal DMA is not used, (i.e. the XFERTYP[DMAEN] bit is not set when the command is sent), the SDHC asserts a DMA request when the amount of buffer space exceeds the value set in the WML register, and is ready for receiving new data. At the same time, the SDHC would set the IRQSTAT[BWR] bit. The buffer write ready interrupt will be generated if it is enabled by software.

When internal DMA is used, the SDHC will not inform the system before all the required number of bytes are transferred (if no error was encountered). When an error occurs during the data transfer, the SDHC will abort the data transfer and abandon the current block. The host driver should read the contents of the DSADDR to get the starting address of the abandoned data block. If the current data transfer is in multi block mode, the SDHC will not automatically send CMD12, even though the XFERTYP[AC12EN] bit is set. The host driver shall send CMD12 in this scenario and re-start the write operation from that address. It is recommended that a software reset for data be applied before the transfer is re-started after error recovery.

The SDHC will not start data transmission until the number of words set in the WML register can be held in the buffer. If the buffer is empty and the host system does not write data in time, the SDHC will stop the SD_CLK to avoid the data buffer under-run situation.

53.5.1.2 Read operation sequence

There are three ways to read data from the buffer when the user transfers data to the card:

1. By using the external DMA through the SDHC DMA request signal
2. By processor core polling through the IRQSTAT[BRR] bit (interrupt or polling)
3. By using the internal DMA

When internal DMA is not used (i.e. XFERTYP[DMAEN] bit is not set when the command is sent), the SDHC asserts a DMA request when the amount of data exceeds the value set in the WML register, that is available and ready for system fetching data. At the same time, the SDHC would set the IRQSTAT[BRR] bit. The buffer read ready interrupt will be generated if it is enabled by software.

When internal DMA is used, the SDHC will not inform the system before all the required number of bytes are transferred (if no error was encountered). When an error occurs during the data transfer, the SDHC will abort the data transfer and abandon 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, the SDHC will not automatically send CMD12, even though the XFERTYP[AC12EN] bit is set. The host driver shall send CMD12 in this scenario and re-start the read operation from that address. It is recommended that a software reset for data be applied before the transfer is re-started after error recovery.

For any write transfer mode, the SDHC will not start data transmission until the number of words set in the WML register are in the buffer. If the buffer is full and the Host System does not read data in time, the SDHC will stop the SDHC_DCLK to avoid the data buffer over-run situation.

53.5.1.3 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 SDHC, the only data buffer can hold up to 128 words (32-bit), and the watermark levels for write and read can be configured respectively. For both read and write, the watermark level can be from 1 word to the maximum of 128 words. For both read and write, the burst length, can be from 1 word to the maximum of 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 may be set to any value between 1 and 4096 bytes inclusive which satisfies the requirements of the external card. The only restriction is from the external card. It might not support that large of a block or it doesn't support a partial block access (which is not the integer times of 512 bytes).

For block size not times of 4, i.e., not word aligned, SDHC requires stuff bytes at the end of each block, because SDHC treats each block individually. For example, the block size is 7 bytes, there are 12 blocks to write, the system side must write 2 times for each block, and for each block, the ending byte would be abandoned by SDHC since it only sends 7 bytes to the card and picks data from the following system write, so there would be 24 beats of write access in total.

53.5.1.4 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 can't be divided evenly into a multiple of the block size, then there are two ways to transfer the data which 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.

The following diagram illustrates the dividing of large data transfers. Assuming a kind of WLAN SDIO card only supports block size up to 64 bytes. Although the SDHC supports a block size of up to 4096 bytes, the SDIO can only accept a block size less than64 bytes, so the data must be divided (see example below).

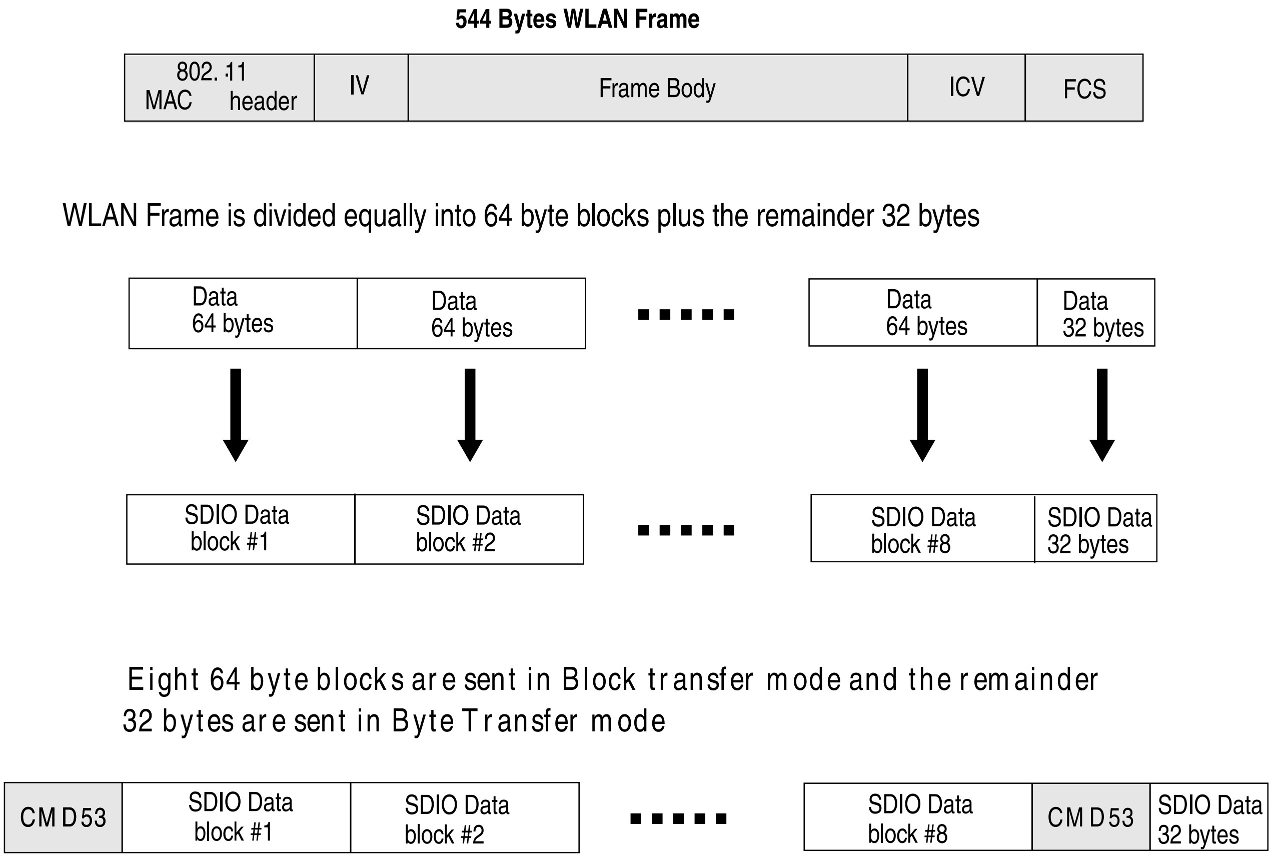


Figure 53-30. Example for dividing large data transfers

53.5.1.5 External DMA request

When the internal DMA is not in use, and external DMA request is enabled, the data buffer will generate a DMA request to the system. During a write operation, when the number of WRWML words can be held in the buffer free space, a DMA request is sent , informing the host system of a DMA write. The IRQSTAT[BWR] bit is also set, as long as the IRQSTATEN[BWRSEN] bit is set. The DMA request is immediately de-asserted when an access to the DATPORT register is made. If the buffer's free space still meets the watermark condition, the DMA request is asserted again after a cycle.

On read operation, when the number of RDWML words are already in the buffer, a DMA request is sent , informing the host system for a DMA read. The IRQSTAT[BRR] bit is also set, as long as the IRQSTATEN[BRRSEN] bit is set. The DMA request is immediately de-asserted when an access to the DATPORT register is made. If the buffer's data still meets the watermark condition, the DMA request is asserted again after a cycle.

Because the DMA burst length can't change during a data transfer for an external DMA transfer, the watermark level (read or write) must be a divisor of the block size. If it is not, transferring of the block may cause buffer under-run (read operation) or over-run (write operation). For example, if the block size is 512 bytes, the watermark level of read (or write) must be a power of two between 1 and 128. For processor core polling access, as the last access in the block transfer can be controlled by software, there is no such issue. The watermark level can be any value, even larger than the block size (but no greater than 128 words). This is because the actual number of bytes transferred by the software can be controlled and does not exceed the block size in each transfer.

The SDHC also supports non-word aligned block size, as long as the card supports that block size. In this case, the watermark level should be set as the number of words. For example, if the block size is 31 bytes, the watermark level can be set to any number of word. For this case, the BLKATTR[BLKSIZE] bits shall be set as 1fh. For the CPU polling access, the burst length can be 1 to 128 words, without restriction. This is because the software will transfer 8 words, and the SDHC will also set the IRQSTAT[BWR] or IRQSTAT[BRR] bits when the remaining data does not violate data buffer. Refer to DMA burst length for more details about the dynamic watermark level of the data buffer. For the above example, even though 8 words are transferred via the DATPORT register, the SDHC will transfer only 31 bytes over the SD bus, as required by the BLKATTR[BLKSIZE] bits. In this data transfer, with non-word aligned block size, the endian mode should be set cautiously, or invalid data will be transferred to/from the card.

53.5.2 DMA crossbar switch interface

The internal DMA implements a DMA engine and the crossbar switch master. When the internal DMA is enabled (XFERTYP[DMAEN] is set), the interrupt status bits are set if they are enabled. To avoid setting them, clear IRQSTATEN[BWRSEN, BRRSEN]. The following diagram illustrates the DMA crossbar switch interface block.

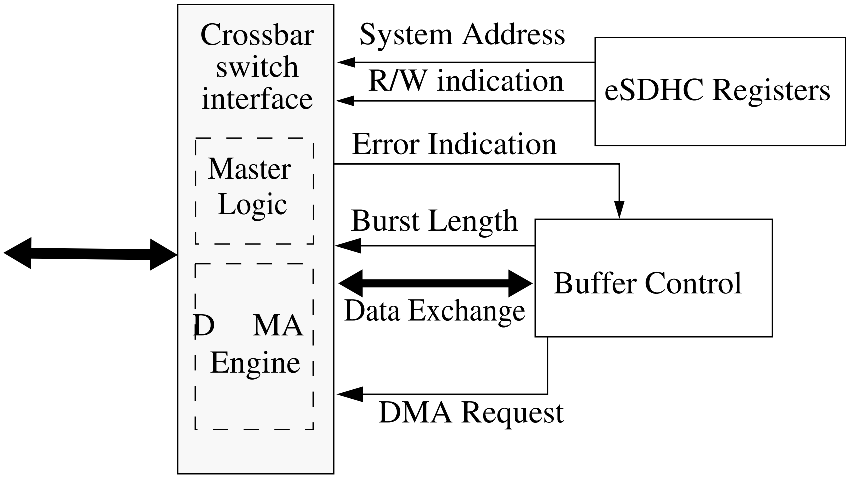


Figure 53-31. DMA crossbar switch interface block

53.5.2.1 Internal DMA request

If the watermark level requirement is met in data transfer, and the internal DMA is enabled, the data buffer block sends a DMA request to the crossbar switch interface. Meanwhile, the external DMA request signal is disabled. The delay in response from the internal DMA engine depends on the system bus loading and the priority assigned to the SDHC. 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 once an access to the buffer is made. Upon access to the buffer by internal DMA, the data buffer updates its internal buffer pointer, and when the watermark level is satisfied, 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 6 words. After the first burst transfer, if there are more than 2 words in the buffer (which might contain some data of the next block), another DMA request read is sent. This is because the remaining number of words to send for the current block is (31 - 6 * 4) / 4 = 2. The SDHC will read 2 words out of the buffer, with 7 valid bytes and 1 stuff byte.

53.5.2.2 DMA burst length

Just like a CPU polling access, the DMA burst length for the internal DMA engine can be from 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. Take the example in Internal DMA request again. The following burst length after 6 words are read will be 2 words, and the next burst length will be 6 words again. This is because the next block starts, which is 31 bytes, more than 6 words. The host driver writer 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 will be the same.

53.5.2.3 Crossbar switch master interface

It is possible that the internal DMA engine could fail during the data transfer. When this error occurs, the DMA engine stops the transfer and goes to the idle state as well as the internal data buffer stops accepting incoming data. The IRQSTAT[DMAE] is set to inform the driver.

Once the DMAE interrupt is received, the software shall send a CMD12 to abort the current transfer and read the DSADDR[DSADDR] to get the starting address of the corrupted block. After the DMA error is fixed, the software should apply a data reset and re-start the transfer from this address to recover the corrupted block.

53.5.2.4 ADMA engine

In the SD host controller standard, the new DMA transfer algorithm called the ADMA (advanced DMA) is defined. For simple DMA, once the page boundary is reached, a DMA interrupt will be generated and the new system address shall be 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 since the host MCU intervention would not be needed during long DMA based data transfers.

There are two types of ADMA: ADMA1 and ADMA2 in host controller. ADMA1 can support data transfer of 4 KB aligned data in system memory. ADMA2 improves the restriction so that 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 define in SD host controller standard, and if 'end' flag is detected in the descriptor, ADMA will stop after this descriptor is processed.

53.5.2.4.1 ADMA concept and descriptor format

For ADMA1, including 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.

For ADMA2, including the following descriptors:

- Valid/Invalid descriptor.
- Nop descriptor.
- Rsv descriptor.
- Set data length & address descriptor.
- Link descriptor.
- Interrupt flag and end flag in descriptor.

ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it will ignore the high 32-bit. Address field shall be set on word aligned(lower 2-bit is always set to 0). Data length is in byte unit.

ADMA will start 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 tran type descriptor. Every tran type will trigger 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 will be the value for previous transfer, or 0 if no set descriptor is ever met.

For ADMA2, tran type descriptor contains both data length and transfer data address, so only a tran type descriptor can start a data transfer

Table 53-35. Format of the ADMA1 descriptor table

Address/page field		Address/page field		Attribute field					
31	12	11	6	5	4	3	2	1	0
Address or data length		000000		Act2	Act1	0	Int	End	Valid

Table continues on the next page...

Table 53-35. Format of the ADMA1 descriptor table (continued)

Act2	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.				

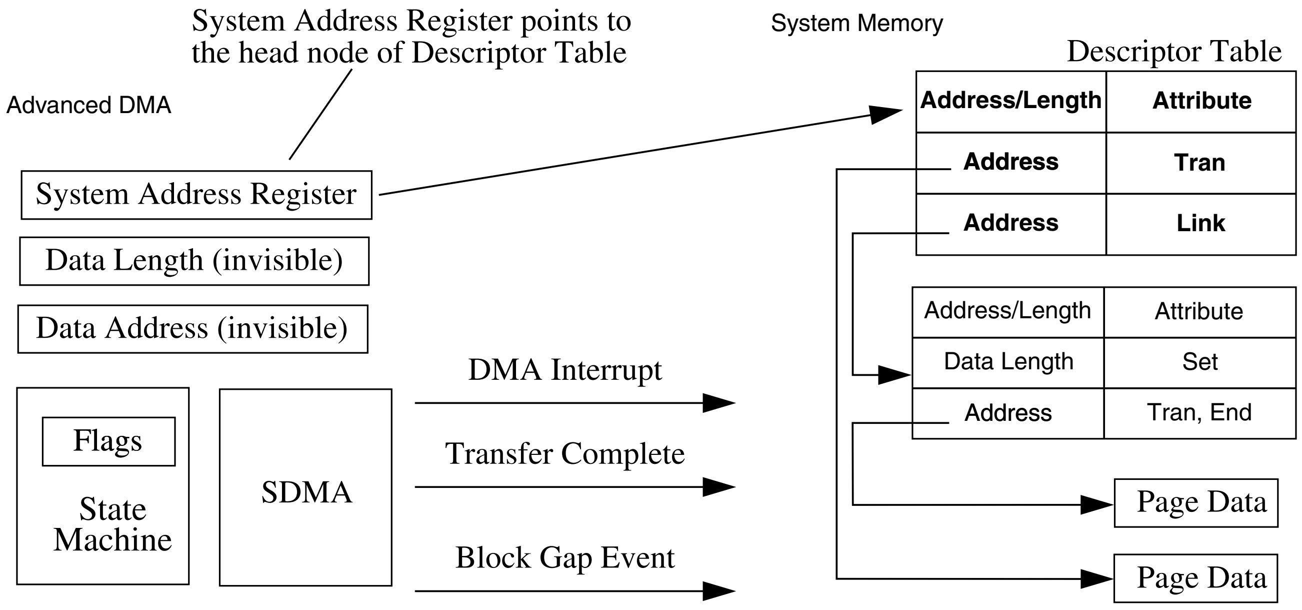


Figure 53-32. Concept and access method of ADMA1 descriptor table

Table 53-36. Format of the ADMA2 descriptor table

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		Act2	Act1	0	Int	End	Valid

Act2	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

Table continues on the next page...

Table 53-36. Format of the ADMA2 descriptor table (continued)

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 done.

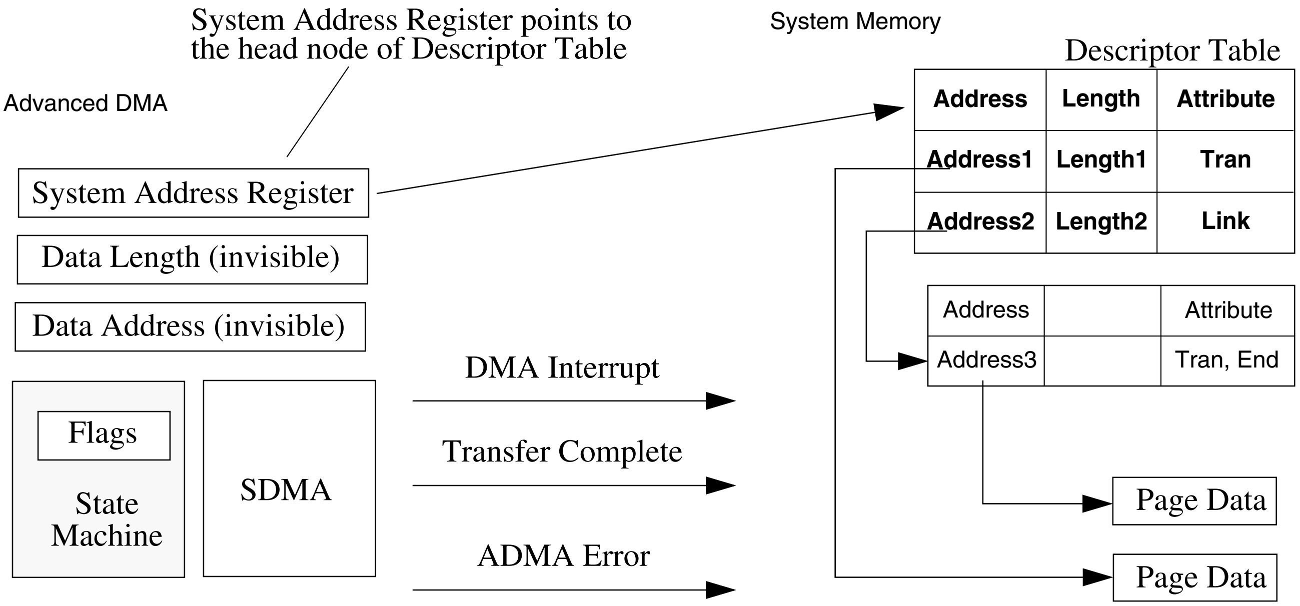


Figure 53-33. Concept and access method of ADMA2 descriptor table

53.5.2.4.2 ADMA interrupt

If the 'interrupt' flag of descriptor is set, ADMA will generate an interrupt according to different type descriptor:

For ADMA1:

- Set type descriptor: interrupt is generated when data length is set.
- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type descriptor: interrupt is generated when new descriptor address is set.
- Nop type descriptor: interrupt is generated just after this descriptor is fetched.

For ADMA2:

- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type descriptor: interrupt is generated when new descriptor address is set.
- Nop/Rsv type descriptor: interrupt is generated just after fetch this descriptor.

53.5.2.4.3 ADMA error

The ADMA stops whenever any error is encountered. These errors include:

- Fetching descriptor error
- Transfer error
- Data length mismatch error

ADMA descriptor error will be generated when it fails to detect 'valid' flag in the descriptor. If ADMA descriptor error occurs, the interrupt is not generated even if the 'interrupt' flag of this descriptor is set.

When XFERTYP[BCEN] bit is set, data length set in buffer must equal to the whole data length set in descriptor nodes, otherwise data length mismatch error will be generated.

If XFERTYP[BCEN] bit is not set, the whole data length set in descriptor should be times of block length, otherwise, when all data set in the descriptor nodes are done not at block boundary, the data mismatch error will occur.

53.5.3 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 DAT[0] line
- 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 sub modules:

1. SD transceiver.

2. SD clock and monitor.
3. Command agent.
4. Data agent.

53.5.3.1 SD transceiver

In the SD protocol unit, the transceiver is the main control module. It consists of a FSM and control module, from which the control signals for all other three modules are generated.

53.5.3.2 SD clock & monitor

This module monitors the signal level on all 8 data lines, the command lines, and directly routes the level values into the register bank. The driver can use this for debug purposes.

The module also detects the CD (card detection) line as well as the DAT[3] line. The transceiver reports the card insertion state according to the CD state, or the signal level on the DAT[3] line, when the PROCTL[D3CD] bit is set.

The module detects the WP (write protect) line. With the information of the WP state, the register bank will ignore the command, accompanied by a write operation, when the WP switch is on.

If the internal data buffer is in danger, and the SD clock must be gated off to avoid buffer over/under-run, this module will assert the gate of the output SD clock to shut the clock off. After the buffer danger has recovered, and when the system access of the buffer catches up, the clock gate of this module will open and the SD clock will be active again.

This module also drive SDHC_LCTL output signal when the PROCTL[LCTL] bit is set by the driver.

53.5.3.3 Command agent

The command agent deals with the transactions on the CMD line. The following diagram illustrates the structure for the command CRC Shift Register.

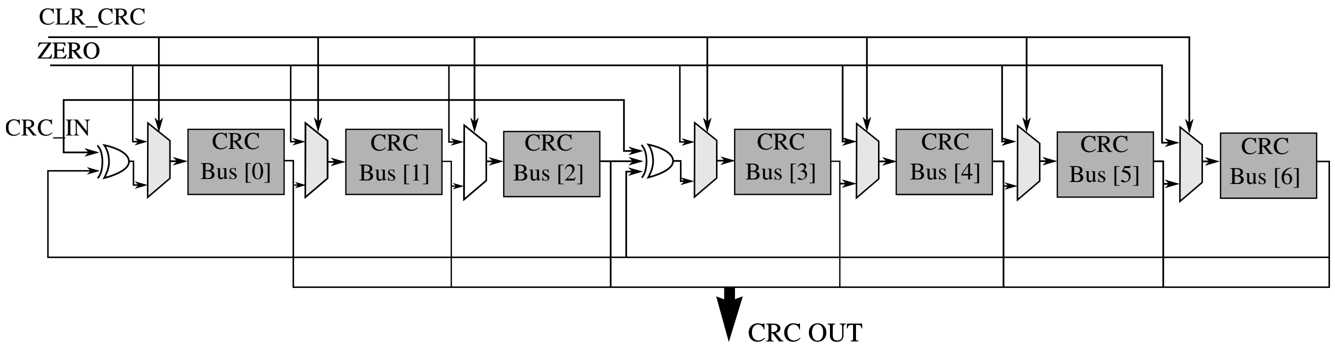


Figure 53-34. Command CRC Shift Register

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)]
```

53.5.3.4 Data agent

The data agent deals with the transactions on the eight data lines. Moreover, this module also detects the busy state on the DAT[0] line, and generate the read wait state by the request from the transceiver. The CRC polynomials for the DAT 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)]
```

53.5.4 Clock & reset manager

This module controls all the reset signals within the SDHC.

There are four kinds of reset signals within SDHC:

1. Hardware reset.
2. Software reset for all.
3. Software reset for the data part.
4. Software reset for the command part.

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.

There are three clocks inside the SDHC:

1. Bus clock.
2. SDHC clock.
3. System clock.

The module monitors the activities of all other modules, supplies the clocks for them, and when enabled, automatically gates off the corresponding clocks.

53.5.5 Clock generator

The clock generator generates the SDHC_CLK by peripheral source clock in two stages. The following diagram illustrates the structure of the divider. The term "base" represents the frequency of peripheral source clock.

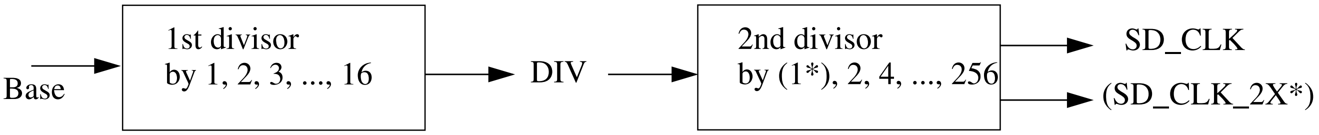


Figure 53-35. Two stages of the clock divider

The first stage outputs an intermediate clock (DIV), which can be base, base/2, base/3, ..., or base/16.

The second stage is a prescaler, and outputs the actual clock (SDHC_CLK). This clock is the driving clock for all sub modules of the SD protocol unit, and the sync FIFOs 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. Thus the highest frequency of the SDHC_CLK is base, and the next highest is base/2, while the lowest frequency is base/4096. If the base clock is of equal duty ratio (usually true), the duty cycle of SDHC_CLK is also 50%, even when the compound divisor is an odd value.

53.5.6 SDIO card interrupt

This section discusses SDIO interrupt handling.

53.5.6.1 Interrupts in 1-bit mode

In this case the DAT[1] pin is dedicated to providing the interrupt function. An interrupt is asserted by pulling the DAT[1] low from the SDIO card, until the interrupt service is finished to clear the interrupt.

53.5.6.2 Interrupt in 4-bit mode

Since the interrupt and data line 1 share Pin 8 in 4-bit mode, an interrupt will only be sent by the card and recognized by the host during a specific time. This is known as the interrupt period. The SDHC will only sample the level on pin 8 during the interrupt period. At all other times, the host will ignore the level on pin 8, and treat 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 4-bit mode, there is only a limited period of time that the interrupt period can be active due to the limited period of data line availability between the multiple blocks of data. This requires a more strict 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 SDHC_D1 line will be held low for one clock cycle with the last clock cycle pulling SDHC_D1 high. On completion of the Interrupt Period, the card releases the SDHC_D1 line into the high Z state. The SDHC samples the SDHC_D1] during the interrupt period when the PROCTL[IABG] bit is set.

Refer to SDIO Card Specification v1.10f for further information about the SDIO card interrupt.

53.5.6.3 Card interrupt handling

When the IRQSIGEN[CINTIEN] bit is set to 0, the SDHC 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 status bit is cleared by resetting the SDIO interrupt. Writing to this bit would have no effects. In 1-bit mode, the SDHC will detect the SDIO interrupt with or without the SD clock (to support wakeup). In 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 has been set, and the host driver needs to service this interrupt, so the SDIO bit in the interrupt control register of SDIO card will be cleared. This is required to clear the

SDIO interrupt status latched in the SDHC 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 Enable bit is set to 1, and the SDHC starts sampling the interrupt signal again.

The following diagram illustrates the SDIO card interrupt scheme and for the sequences of software and hardware events that take place during a card interrupt handling procedure.

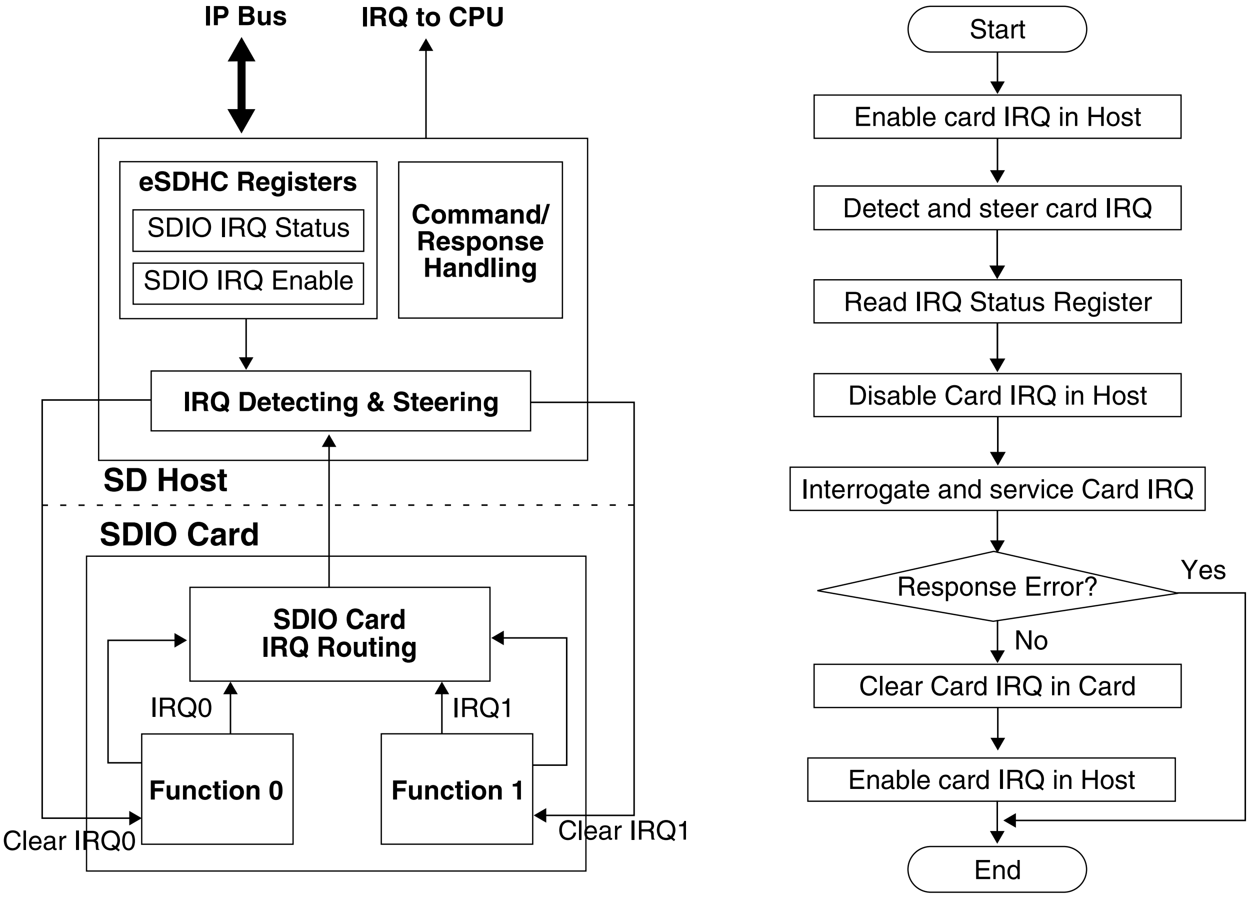


Figure 53-36. Card interrupt scheme and card interrupt detection and handling procedure

53.5.7 Card insertion and removal detection

The SDHC uses either the DAT[3] pin or the CD pin to detect card insertion or removal. When there is no card on the MMC/SD bus, the DAT[3] will be pulled to a low voltage level by default. When any card is inserted to or removed from the socket, the SDHC detects the logic value changes on the DAT[3] pin and generates an interrupt. When the DAT[3] pin is not used for card detection (for example, it is implemented in GPIO), the CD pin must be connected for card detection. Whether DAT[3] is configured for card detection or not, the CD pin is always a reference for card detection. Whether the DAT[3] pin or the CD pin is used to detect card insertion, the SDHC will send an interrupt (if enabled) to inform the Host system that a card is inserted.

53.5.8 Power management and wakeup events

When there is no operation between the SDHC and the card through the SD bus, the user can completely disable the bus clock and SDHC clock in the chip level clock control module to save power. When the user needs to use the SDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to the SDHC 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 SDHC can generate these interrupt even when there are no clocks enabled. The three interrupts which can be used as wake up events are:

1. Card removal interrupt
2. Card insertion interrupt
3. Interrupt from SDIO card

The SDHC 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 the SDHC. For maximum power saving, the user can disable all the clocks to the SDHC when there is no operation in progress.

These three wake up events (or wakeup interrupts) can also be used to wake up the system from low-power modes.

Note

> To make the interrupt a wakeup event, when all the clocks to the SDHC are disabled or when the whole system is in low power mode, the corresponding wakeup enabled bit needs to be set. Refer to protocol control register for more information.

53.5.8.1 Setting wakeup events

For the SDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters sleep mode. Before the software disables the host clock, it should ensure that all of 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

53.5.9 MMC fast boot

In Embedded MultiMediaCard(eMMC4.3) spec, add fast boot feature need hardware support.

In 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.

There are two types of fast boot mode, ' boot operation ' and ' Alternative boot operation ' in eMMC4.3 spec. Each type also has with acknowledge and without acknowledge modes.

Note

for the eMMC4.3 card setting, please refer to eMMC4.3 spec

53.5.9.1 Boot operation

Note

in this block guide, this fast boot is called normal fast boot mode

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 1 second after the CMD line goes low, the slave starts to send the first boot data to the master on the DAT line(s). The master must keep the CMD line low to read all of the boot data.

If boot acknowledge is enabled, the slave has to send acknowledge pattern '010' to the master within 50 ms after the CMD line goes low. If boot acknowledge is disabled, the slave will not send out acknowledge pattern '010'.

The master can terminate boot mode with the CMD line high.

Boot operation will be terminated when all contents of the enabled boot data are sent to the master. After boot operation is executed, the slave shall be ready for CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.

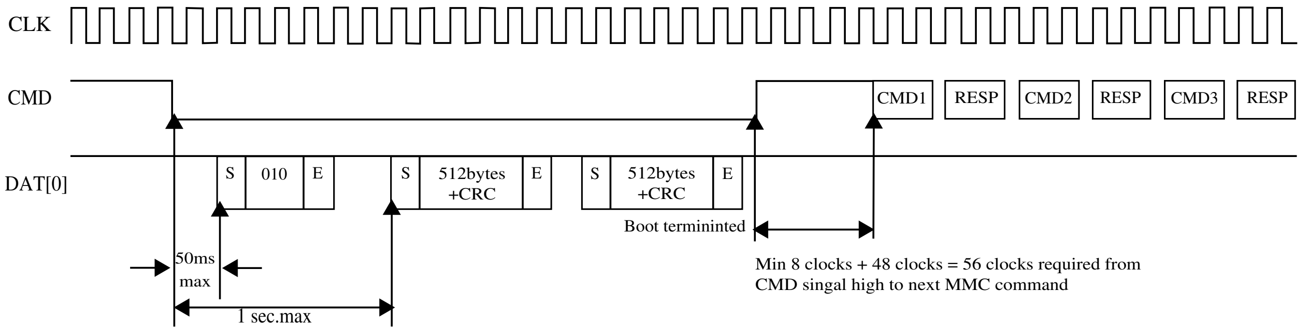


Figure 53-37. Multimediacard state diagram (normal boot mode)

53.5.9.2 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 CMD1is issued or the CMD line goes low, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within 1 second after CMD0 with the argument of 0xFFFFFFFA is issued, the slave starts to send the first boot data to the master on the DAT 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, theslave will not send out acknowledge pattern '010'.

The master can terminate boot mode by issuing CMD0 (Reset).

Boot operation will be terminated when all contents of the enabled boot data are sent to the master. After boot operation is executed, the slave shall be ready for CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.

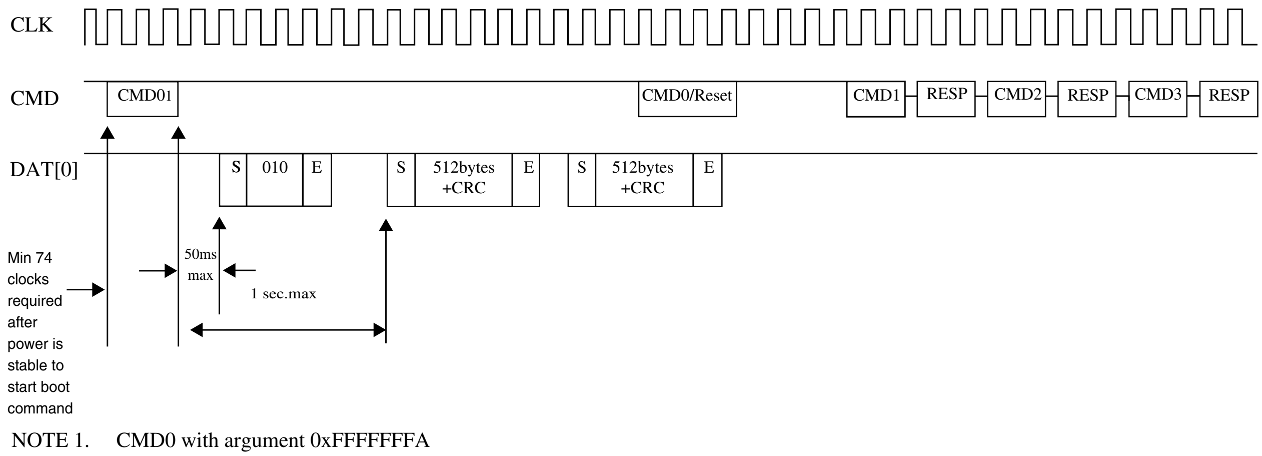

CLK
CMD
CMD01
CMD0/Reset
CMD1
RESP
CMD2
RESP
CMD3
RESP
DAT[0]
S
010
E
S
512bytes +CRC
E
S
512bytes +CRC
E
Min 74 clocks required after power is stable to start boot command
50ms max
1 sec.max
NOTE 1. CMD0 with argument 0xFFFFFFFA

Figure 53-38. MultiMediaCard state diagram (alternative boot mode)

53.6 Initialization/application of SDHC

All communication between 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", "ALL_SEND_CID" and etc. In broadcast mode, all cards are in the open-drain mode to avoid bus contention. Refer to Commands for MMC/SD/SDIO/CE-ATA for the commands of bc and bcr categories.

After the broadcast command CMD3 is issued, the cards enter standby mode. Addressed type commands are used from this point. In this mode, the CMD/DAT I/O pads will turn to push-pull mode, to have the driving capability for maximum frequency operation. Refer to Commands for MMC/SD/SDIO/CE-ATA for the commands of ac and adtc categories.

53.6.1 Command send and response receive basic operation

Assuming the data type WORD is an unsigned 32-bit integer, the below flow is 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. By doing this, make sure 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 go to the standby state, no response to the host when CMD2 is sent. The host driver shall deal with 'fake' errors like this with caution.

53.6.2 Card identification mode

When a card is inserted to the socket or the card was 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 the MMC cards. For CE-ATA, the device is connected in a point-to-point manner, and no RCA is needed. All data communications in the Card Identification Mode use the command line (CMD) only. Refer to CE-ATA Digital Protocol, Revision 1.1 for more details.

53.6.2.1 Card detect

The following diagram illustrates the detection of MMC, SD and SDIO cards using the SDHC.

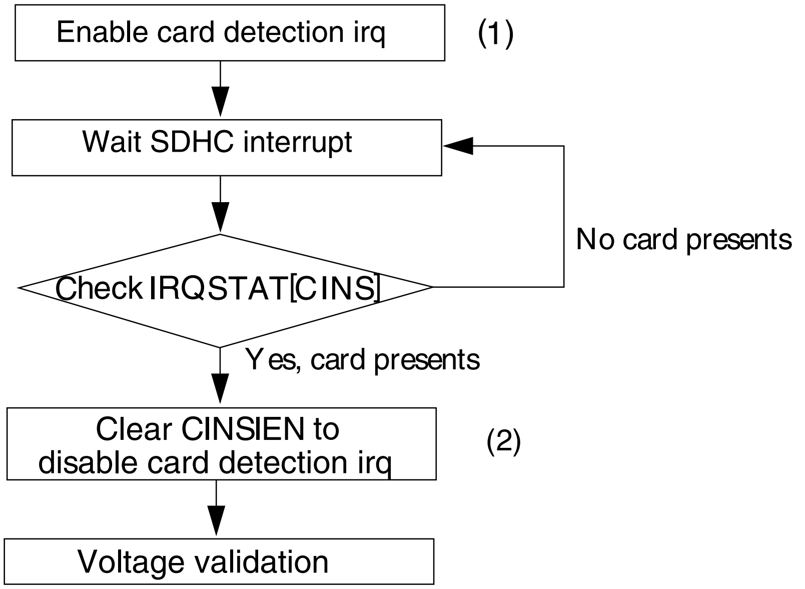


Figure 53-39. Flow diagram for card detection

Here is the card detection sequence:

- Set the CINSIEN bit to enable card detection interrupt
- When an interrupt from the SDHC is received, check the IRQSTAT[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

To detect a CE-ATA device, after completing the normal MMC reset and initialization procedures, the host driver shall issue CMD 60 to check for a CE-ATA signature. If the device responds to the command with the CE-ATA signature, a CE-ATA device has been found. Then the driver should query EXT_CSD register byte 504 (S_CMD_SET) in the MMC register space. If the ATA bit (bit 4) is set, then the MMC device is an ATA device. If the device indicates that it is an ATA device, the Driver should set the ATA bit (bit 4) of the EXT_CSD register byte 191 (CMD_SET) to activate the ATA command set for use. To choose the command set, the driver shall issue CMD6. It is possible that the CE-ATA device does not support the ATA mode, so the driver shall not issue ATA command to the device.

53.6.2.2 Reset

The host consists of three types of resets:

- Hardware reset (card and host) which is driven by POR (power on reset)

- Software reset (host only) is proceed by the write operation on the SYSCTL[RSTD], SYSCTL[RSTC], or SYSCTL[RSTA] bits 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, SD Memory cards, and CE-ATA 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 the 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.The following diagram illustrates the software flow to reset both the SDHC and the card.

For CE-ATA device that supports ATA mode, before issuing CMD0 to reset the MMC layer, two CMD39 should be issued back-to-back to the ATA control register. The first CMD39 shall have the SRST bit set to one. The second CMD39 command shall have the SRST bit cleared to zero.

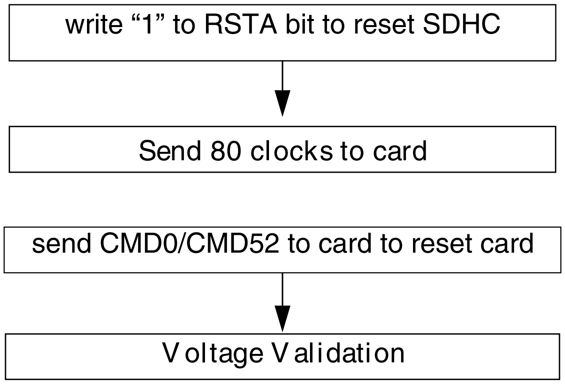

write “1” to RSTA bit to reset SDHC
Send 80 clocks to card
send CMD0/CMD52 to card to reset card
Voltage Validation

Figure 53-40. Flow chart for reset of the SDHC and SD I/O card

```
software_reset()
{
set_bit(SYSCTRL, RSTA); // software reset the Host
set DTOCV and SDCLKFS bit fields to get the SD_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>);
}
```

53.6.2.3 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 V_{DD} 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 V_{DD} conditions. This means if the host and card have non-common V_{DD} ranges, the card will not be able to complete the identification cycle, nor will it be able to send CSD data.

Therefore, a special command Send_Op_Cont (CMD1 for MMC), SD_Send_Op_Cont (ACMD41 for SD Memory) and IO_Send_Op_Cont (CMD5 for SD I/O) is used. For a CE-ATA card, the process is the same as that of an MMC card. The voltage validation procedure is designed to provide a mechanism to identify and reject cards which do not match the V_{DD} range(s) desired by the host. This is accomplished by the host sending the desired V_{DD} voltage window as the operand of this command. Cards that can't 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 shall be 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 or CE-ATA 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 ...
 if (check for CE-ATA signature succeeded) { // the card is CE-ATA
  store CE-ATA specific info from the signature;
  label the card as CE-ATA;
 } // of if (check for CE-ATA ...
 else label the card as MMC;
} // of else
}
```

53.6.2.4 Card registry

Card registry for the MMC and SD/SDIO/SD combo cards are different. For CE-ATA, it enters the tran state after reset is completed.

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 specification). At this time, the CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated, the host will request the card to send their valid operation conditions. The response to ACMD41 is the operation condition register of the card. The same command shall be send to all of 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 will be used to address the card for future data transfer operations. Once 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 system.

For MMC operation, the host starts the card identification process in 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 allow parallel card operation during card identification. After the bus is activated the host will request 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 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. Since the CID is unique for each card, only one card can be successfully send its full CID to the host. This card then goes into the identification state. Thereafter, the host issues Set_Relative_Addr (CMD3) to assign to the card a relative card address (RCA). Once the RCA is received the card state changes to the standby state, and the card does not react in further identification cycles, and 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.

For CE-ATA operation (same interface as 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 or CE-ATA) { // treat CE-ATA as MMC
  send_command(ALL_SEND_CID, <...>);
  rca = 0x1; // arbitrarily set RCA, 1 here for example, this RCA is also the
relative address to access the CE-ATA card
  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);
}
```

53.6.3 Card access

This section discusses the various card access methods.

53.6.3.1 Block write

This section discusses the block write access methods.

53.6.3.1.1 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 shall indicate the failure on the dat line. The transferred data will be discarded and not written, and all further transmitted blocks (in multiple block write mode) will be 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, and the CE-ATA card does not support partial block write, either), 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. For a CE-ATA card, check the CE-ATA card specification for its behavior in block misalignment. 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 will report an error and 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 will begin writing and hold the DAT 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 and CE-ATA cards at any time, and the card will respond 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 DAT line without interrupting the write operation. When re-selecting the card, it will reactivate the busy indication by pulling DAT 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 two methods (by means of external DMA or 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:
    a. For SD/MMC cards, use SET_BLOCKLEN (CMD16)
    b. 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)
    c. For CE-ATA cards, configure bits 1~0 in the scrControl register
3. Set the eSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the eSDHC 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 some another error, that occurred during the auto12 command sending and response receiving.

53.6.3.1.2 Write with pause

The write operation can be paused during the transfer. Instead of stopping the SD_CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the PROCTL[SABGREQ] 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 DAT0 line is not required to de-assert to release the busy state, no suspend command is needed.

Like in the flow described in Normal write, the write with pause is shown with the same kind of write operation:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:
    a. For SD/MMC, use SET_BLOCKLEN (CMD16)
    b. 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)
    c. For CE-ATA cards, configure bits 1~0 in the scrControl register
3. Set the SDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the SDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the SDHC DMA when sending the command with data transfer. The XFERTYP[AC12EN] bit should also be set.
6. Set the PROCTL[SABGREQ] bit.
7. Wait for the transfer complete interrupt.
8. Clear the PROCTL[SABGREQ] bit.
9. Check the status bit to see if a write CRC error occurred.
10. Set the PROCTL[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 BLKATTR[BLKCNT] . As the data transfer and the setting of the PROCTL[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 driver shall read the value of BLKATTR[BLKCNT] after the transfer is paused and the transfer complete interrupt is received.

It is also possible 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 actually the end of the transfer. These types of requests are ignored and the Driver should treat this 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 (if not needed), it is recommended to avoid using the suspend command for the SDIO card. This is because when such a command is sent, the SDHC thinks the system will switch to another function on the SDIO card, and flush the data buffer. The SDHC takes the resume command as a normal command with data transfer, and it is left for the 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 XFERTYP[MSBSEL] and XFERTYP[BCEN] bits are set as well as the XFERTYP[AC12EN] bit. However, the SDHC will automatically send a CMD12 to mark the end of the multi-block transfer.

53.6.3.2 Block read

This section discusses the block read access methods.

53.6.3.2.1 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 will be 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 two methods (by means of external DMA or 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, wait until card is ready for data.
2. Set the card block length/size:
   a. For SD/MMC, use SET_BLOCKLEN (CMD16)
   b. 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)
   c. For CE-ATA cards, configure bits 1~0 in the scrControl register

3. Set the SDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the SDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer read ready interrupt, configure the DMA settings and enable the SDHC DMA when sending the command with data transfer. The XFERTYP[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 some another error, occurred during the auto12 command sending and response receiving.

53.6.3.2.2 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 and wait feature can pause during the read operation. If the SDIO card support read wait (SRW bit in CCCR register is 1), the Driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks. Before setting the SABGREQ bit, make sure the RWCTL bit in the Protocol Control register is set, otherwise the eSDHC will not assert the Read Wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit once the Read Wait capability of the SDIO card is recognized.

Like in 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 the card supports Read Wait.
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:
    a. For SD/MMC, use SET_BLOCKLEN (CMD16)
    b. 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)
    c. For CE-ATA cards, configure bits 1~0 in the scrControl register

5. Set the SDHC block length register to be the same as the block length set for the card in Step 2.
6. Set the SDHC number block register (NOB), nob is 5 (for instance).
7. Disable the buffer read ready interrupt, configure the DMA setting and enable the eSDHC 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 some another error, occurred during the auto12 command sending and response receiving.

Like the write operation, it is possible to meet the ending block of the transfer when paused. In this case, the SDHC will ignore the Stop At Block Gap Request and treat 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 will be 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, the SDHC takes the command as a normal one accompanied with data transfer. It is left for the 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 SDHC will automatically send the CMD12 to mark the end of multi-block transfer.

53.6.3.3 Suspend resume

The SDHC supports the suspend resume operations of SDIO cards, although slightly different than the suggested implementation of Suspend in the SDIO card specification..

53.6.3.3.1 Suspend

After setting the PROCTL[SABGREQ] bit, the host driver may send a suspend command to switch to another function of the SDIO card. The SDHC does not monitor the content of the response, so it doesn't know if the suspend command succeeded or not. Accordingly, it doesn't de-assert read wait for read pause. To solve this problem, the driver shall not mark the suspend command as a "suspend", (i.e. setting the XFERTYP[CMDTYP] bits to 01). Instead, the driver shall send 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 the SDHC that the current transfer is suspended. As shown in the following sequence for suspend operation:

1. Set the PROCTL[SABGREQ] bit to pause the current data transfer at block gap.
2. After the IRQSTAT[BGE] bit is set, send the suspend command to suspend the active function. The XFERTYP[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 XFERTYP[CMDTYP] of this command must be 2'b01, so the SDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, the SDHC stops driving DAT2 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.

53.6.3.3.2 Resume

To resume the data transfer, a resume command shall be issued:

1. To resume the suspended function, restore the context register with the saved value in step #5 of the suspend operation above.
2. Send the resume command. In the transfer type register, all 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 will be resumed.

53.6.3.4 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 accomplish this are:

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 (4 KB 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 BLKATTR register.
6. Set the DSADDR register to the address of the first descriptor and set the PROCTL[DMAS] field to 01 to select the ADMA.
7. Issue a write or read command with the XFERTYP[DMAEN] bit set to 1.

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.

53.6.3.5 Transfer error

This section discusses the handling of transfer errors.

53.6.3.5.1 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 shall be 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 shall issue a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the XFERTYP[AC12EN] and BCEND bits are set, the SDHC 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 will be sent by the SDHC. In this case, the driver shall re-send or re-obtain the last block with a single block transfer.

53.6.3.5.2 Internal DMA error

During the data transfer with internal simple DMA, if the DMA engine encounters some error on the system bus, the DMA operation is aborted and DMA error interrupt is sent to the host system. When acknowledged by such an interrupt, the driver shall calculate the start address of data block in which the error occurs. The start address can be calculated by either:

1. Read the DMA system address register. The error occurs during the previous burst. Taking the block size, the previous burst length and the start address of the next burst transfer into account, it is straight forward to obtain the start address of the corrupted block.
2. Read the BLKCNT field of the block attribute register. By 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 does 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 re-start the transfer from the corrupted block to recover from the error.

53.6.3.5.3 ADMA error

There are three kinds of possible ADMA errors. The transfer, invalid descriptor, and data-length mismatch errors. 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.

1. 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 the next block if no block is corrupted.

2. Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and re-create 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.
3. 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.

53.6.3.5.4 Auto CMD12 error

After the last block of the multi block transfer is sent or received, and the XFERTYP[AC12EN] bit is set when the data transfer is initiated by the data command, the SDHC automatically sends a CMD12 to the card to stop the transfer. When errors with this command occur, it is recommended to the driver to deal with the situations in the following manner:

1. Auto CMD12 response time-out. It is not certain whether the command is accepted by the card or not. The driver should clear the IRQSTAT[AC12E] bits and re-send the CMD12 until it is accepted by the card.
2. Auto CMD12 response CRC error. Since card responds to the CMD12, the card will abort the transfer. The driver may ignore the error and clear the IRQSTAT[AC12E] bit.
3. Auto CMD12 conflict error or not sent. The command is not sent, so the driver shall send a CMD12 manually.

53.6.3.6 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 DAT[1] line during some special period. For the CE-ATA card, it can be a pulse on the CMD line to inform the host controller that the command and its response is finished, and it is possible that some additional external interrupt behaviors are defined. The SDHC only monitors the DAT[1] line and supports the SDIO interrupt.

When the SDIO interrupt is captured by the SDHC, and the host system is informed by the SDHC asserting the SDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt factor is controlled by the external card, the interrupt from the SDIO card must be served before the IRQSTAT[CINT] bit is cleared by written 1. Refer to Card interrupt handling for the card interrupt handling flow.

53.6.4 Switch function

MMC cards transferring data at bus widths other than 1-bit is a new feature added to the MMC specifications. The high speed timing mode for all card devices, was also recently defined in various card specifications. To enable these new features, a "switch" command shall be issued by the host driver.

For SDIO cards, the high speed mode is enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed. For SD cards, the high speed mode is queried and enabled by a CMD6 (with the mnemonic symbol as SWITCH_FUNC). For MMC cards (and CE-ATA over MMC interface), the high speed mode is queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The 4-bit and 8-bit bus width of the 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 flowcharts do not show current capability check, which is recommended in the function switch process.

53.6.4.1 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 eSDHC 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 eSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

53.6.4.2 Query, enable and disable SD high speed mode

```
enable_sd_high_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0xFFFFF1 and read 64 bytes of data accompanying the R1 response;
wait data transfer done bit is set;
check if the bit 401 of received 512 bit is set;
if (bit 401 is '0') report the SD card does not support high speed mode and return;
send CMD6, with argument 0x80FFFFF1 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 eSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sd_high_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 eSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

53.6.4.3 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 eSDHC 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 eSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}
```

53.6.4.4 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, x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}
```

53.6.5 ADMA operation

This section presents code examples for ADMA operation.

53.6.5.1 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 align);
      }
      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;
}
```

53.6.5.2 ADMA2 operation

```
Set_adma2_descriptor
{
      if (to start data transfer) {
          // Make sure the address is 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';
}
```

53.6.6 Fast boot operation

This section discusses fast boot operations.

53.6.6.1 Normal fast boot flow

1. Software need to configure SYSCTL[INITA] to make sure 74 card clocks are finished.
2. Software need to configure MMCBOOT[BOOTEN] to 1, and MMCBOOT[BOOTMODE] to 0, and MMCBOOT[BOOTACK] to select the ack mode or not. If need to send through DMA mode, need to configure MMCBOOT[AUTOSABGEN] to enable automatically stop at block gap feature. And need to configure MMCBOOT[DTOCVACK] to select the ack timeout value according to the sd clk frequence.
3. Software then need to configure BLKATTR register to set block size/no.
4. Software need to configure PROCTL[DTW].
5. Software need to configure CMDARG to set argument if needed(no need in normal fast boot).

6. Software need to configure XFERTYP register to start the boot process . In normal boot mode, XFERTYP[CMDINX], XFERTYP[CMDTYP], XFERTYP[RSPTYP], XFERTYP[CICEN], XFERTYP[CCCEN], XFERTYP[AC12EN], XFERTYP[BCEN] and XFERTYP[DMAEN] are kept default value. XFERTYP[DPSEL] bit is set to 1, XFERTYP[DTDSEL] is set to 1, XFERTYP[MSBSEL] is set to 1. Note XFERTYP[DMAEN] should be configured as 0 in polling mode. And if XFERTYP[BCEN] is configured as 1, better to configure BLKATTR[BLKSIZE] to the max value.
7. When the step 6 is configured, boot process will begin. Software need to poll the data buffer ready status to read the data from buffer in time. If boot time-out happened(ack time out or the first data read time out), Interrupt will be triggered, and software need to configure MMCBOOT[]BOOTEN] to 0 to disable boot. Thus will make CMD high, and then after at least 56 clocks, it is ready to begin normal initialization process.
8. If no time out, software need to decide the data read is finished and then configure MMCBOOT[]BOOTEN] to 0 to disable boot. This will make CMD line high and command completed asserted. After at least 56 clocks, it is ready to begin normal initialization process.
9. Reset the host and then can begin the normal process.

53.6.6.2 Alternative fast boot flow

1. Software need to configure init_active bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software need to configure MMCBOOT [BOOTEN] to 1, and MMCBOOT [BOOTMODE] to 1, and MMCBOOT [BOOTACK] to select the ack mode or not. If need to send through DMA mode, need to configure MMCBOOT [AUTOSABGEN] to enable automatically stop at block gap feature. And need to configure MMCBOOT [DTOCVACK] to select the ack timeout value according to the sd clk frequence.
3. Software then need to configure BLKATTR register to set block size/no.
4. Software need to configure PROCTL[DTW].
5. Software need to configure CMDARG register to set argument to 0xFFFFFFFA.
6. Software need to configure XFERTYP register to start the boot process by CMD0 with 0xFFFFFFFA argument . In alternative boot, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN are kept default value. DPSEL bit

is set to 1, DTDSEL is set to 1, MSBSEL is set to 1. Note DMAEN should be configured as 0 in polling mode. And if BCEN is configured as 1 in polling mode, better to configure blk no in Bock Attributes Register to the max value.

7. When the step 6 is configured, boot process will begin. Software need to poll the data buffer ready status to read the data from buffer in time. If boot time out (ack data time out in 50ms or data time out in 1s), host will send out the interrupt and software need to send CMD0 with reset and then configure boot enable bit to 0 to stop this process. After command completed, configure MMCBOOT[BOOTEN] to 0 to disable boot. After at least 8 clocks from command completed, card is ready for identification step.
8. If no time out, software need to decide when to stop the boot process, and send out the CMD0 with reset and then after command completed, configure MMCBOOT[BOOTEN] to stop the process. After 8 clocks from command completed, slave(card) is ready for identification step.
9. Reset the host and then can begin the normal process.

53.6.6.3 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, need to switch DMA parameters on the fly during MMC fast boot.

In fast boot, host can use ADMA2(advanced DMA2) with two destinations.

The detail flow:

1. Software need to configure init_active bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software need to configure MMCBOOT[BOOTEN] to 1; and MMCBOOT[BOOTMODE] to 0 (normal fast boot), to 1(alternative boot); and MMCBOOT[BOOTACK] to select the ack mode or not. In DMA mode, configure MMCBOOT[AUTOSABGEN] to 1 for enable automatically stop at block gap feature. Also configure MMCBOOT[BOOTBLKCNT] to set the VAULE1(value of block count that need to trans first time), that host will stop at block gap when card block counter is equal to this value. And need to configure MMCBOOT[DTOCVACK] to select the ack timeout value according to the sd clk frequence.

3. Software then need to configure BLKATTR register to set block size/no. In DMA mode, it is better to set block number to the max value(16'hffff).
4. Software need to configure PROCTL[DTW].
5. Software enable ADMA2 by configuring PROCTL[DMAS].
6. Software need to set at least three pairs ADMA2 descriptor in boot memory (ie, in IRAM, at least 6 words). The first pair descriptor define the start address (ie, IRAM) and data length(ie,512byte*VALUE1) of first part boot code. Software also need to set the second pair descriptor, the second start address (any value that is writable), data length is suggest to set 1~2word (record as VAULE2). Note: the second couple desc also transfer useful data even at lease 1 word. Because our ADMA2 can't support 0 data_length data transfer descriptor.
7. Software need to configure CMDARG register to set argument to 0xFFFFFFFA in alternative fast boot, and don't need set in normal fast boot.
8. Software need to configure XFERTYP register to start the boot process . XFERTYP[CMDINX], XFERTYP[CMDTYP], XFERTYP[RSPTYP], XFERTYP[CICEN], XFERTYP[CCCEN], XFERTYP[AC12EN], XFERTYP[BCEN] and XFERTYP[DMAEN] are kept default value. XFERTYP[DPSEL] bit is set to 1, XFERTYP[DTDSEL] is set to 1, XFERTYP[MSBSEL] is set to 1. XFERTYP[DMAEN] is configured as 1 in DMA mode. And if XFERTYP[BCEN] is configured as 1, better to configure blk no in BLKATTR register to the max value.
9. When the step 8 is configured, boot process will begin, the first VAULE1 block number data has transfer. Software need to polling IRQSTAT[TC] bit to determine first transfer is end. Also software need to polling IRQSTAT[BGE] bit to determine if first transfer stop at block gap.
10. When IRQSTAT[TC] and IRQSTAT[BGE] bits are 1, . SW can analyzes the first code of VAULE1 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 boot code (VAULE3 the remain boot code block). Remember set the last descriptor with END.
11. Software need to configure MMCBOOT register (offset 0xc4) again. Set MMCBOOT[BOOTEN] bit to 1; and MMCBOOT[BOOTMODE] bit to 0 (normal fast boot), to 1(alternative boot); and MMCBOOT[BOOTACK] bit to select the ack mode or not. In DMA mode, configure MMCBOOT[AUTOSABGEN] bit to 1 for enable automatically stop at block gap feature. Also configure MMCBOOT[BOOTBLKCNT] bit to set the (VAULE1+1+VAULE3), that host will

stop at block gap when card block counter is equal to this value. And need to configure MMCBOOT[DTOCVACK] bit to select the ack timeout value according to the sd clk frequence.

12. Software need to clear IRQSTAT[TC] and IRQSTAT[BGE] bit. And software need to clear PROCTL[SABGREQ], and set PROCTL[CREQ] to 1 to resume the data transfer. Host will transfer the VALUE2 and VAULE3 data to the destination that is set by descriptor.
13. Software need to polling IRQSTAT[BGE] bit to determine if the fast boot is over.

Note

1. When ADMA boot flow is started, for SDHC, it is like a normal ADMA read operation. So setting ADMA2 descriptor as the normal ADMA2 transfer.
2. Need a few words length memory to keep descriptor.
3. For the 1~2 words data in second descriptor setting, it is the useful data, so software need to deal the data due to the application case.

53.6.7 Commands for MMC/SD/SDIO/CE-ATA

The following table lists the commands for the MMC/SD/SDIO/CE-ATA cards.

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 DAT.
4. addressed (point-to-point) data transfer commands (adtc).

Table 53-37. Commands for MMC/SD/SDIO/CE-ATA cards

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all MMC and SD memory cards to idle state.

Table continues on the next page...

Table 53-37. Commands for MMC/SD/SDIO/CE-ATA cards (continued)

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
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 their 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 [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to "The 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.

Table continues on the next page...

Table 53-37. Commands for MMC/SD/SDIO/CE-ATA cards (continued)

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
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	Reserved				
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.
CMD21-23	Reserved				
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.

Table continues on the next page...

Table 53-37. Commands for MMC/SD/SDIO/CE-ATA cards (continued)

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
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 shall be 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 53-37. Commands for MMC/SD/SDIO/CE-ATA cards (continued)

CMD INDEX	Type	Argument	Resp	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.

Table continues on the next page...

Table 53-37. Commands for MMC/SD/SDIO/CE-ATA cards (continued)

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
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 [15:8] stuff bits [7:0] byte count	R1b	RW_MULTIPLE_REGISTER	CE-ATA devices contain a set of Status and Control registers that begin at register offset 80h.These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 shall be 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 for the ATA command.
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[5]	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD memory card status.
ACMD22[6]	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[7]	ac		R1	SET_WR_BLK_ERASE_COUNT	-
ACMD41[8]	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[9]	ac		R1	SET_CLR_CARD_DETECT	-

Table continues on the next page...

Table 53-37. Commands for MMC/SD/SDIO/CE-ATA cards (continued)

CMD INDEX	Type	Argument	Resp	Abbreviation	Description
ACMD51[10]	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 as well as for CE-ATA cards over the MMC interface. 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 53-38.
4. ACMDs shall be preceded with the APP_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).
5. ACMDs shall be preceded with the APP_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).
6. ACMDs shall be preceded with the APP_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).
7. ACMDs shall be preceded with the APP_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).
8. ACMDs shall be preceded with the APP_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).
9. ACMDs shall be preceded with the APP_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).
10. ACMDs shall be 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 in the following table.

Table 53-38. EXT_CSD Access Modes

Bits	Access Name	Operation
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 1 bits in the Value field.
10	Clear Bits	The bits in the pointed byte are cleared, according to the 1 bits in the Value field.
11	Write Byte	The Value field is written into the pointed byte.

53.7 Software restrictions

Software for the SDHC must observe the following restrictions.

53.7.1 Initialization active

The driver cannot set SYSCTL[INITA] bit when any of the command line or data lines is active, so the driver must ensure both PRSSTAT[CDIHB] and PRSSTAT[CIHB] bits are cleared. And in order to auto clear the SYSCTL[INITA] bit, the SYSCTL[SDCLKEN] bit must be '1', otherwise no clocks can go out to the card and SYSCTL[INITA] will never clear.

53.7.2 Software polling procedure

For polling read or write, once 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 the times of the value in watermark level register (read and write respectively), the software must access exactly the remained number of words at the end of each block. For example, for read operation, if the WML[RDWML] 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.

53.7.3 Suspend operation

In order to suspend the data transfer, the software must inform SDHC 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 (XFERTYP[CMDTYP] bits set as '01') to inform SDHC that the transfer is suspended.

If software needs resume the suspended transfer, it should read the value in BLKATTR[BLKCNT] to save the remained number of blocks before sending the normal command marked as suspend, otherwise on sending such 'suspend' command, SDHC will regard the current transfer is aborted and change BLKATTR[BLKCNT] to its original value, instead of keeping the remained number of blocks.

53.7.4 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 times of 4.

53.7.5 (A)DMA address setting

To configure ADMA1/ADMA2/DMA address register, when TC[IRQSTAT] bit is set, the register will always update itself with the internal address value to support dynamic address synchronization, so software must make sure TC[IRQSTAT] bit is cleared prior to configuring ADMA1/ADMA2/DMA address register.

53.7.6 Data port access

Data port does not support parallel access. For example, during an external 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 external DMA. Otherwise the data would be corrupted inside the SDHC buffer.

53.7.7 Change clock frequency

SDHC does not automatically gates off the card clock when the host driver changes the clock frequency. To remove possible glitch on the card clock, clear SYSCTL[SDCLKEN] bit when changing clock divisor value and set SYSCTL[SDCLKEN] bit to '1' after PRSSTAT[SDSTB] bit is '1' again.

53.7.8 Multi-block read

For pre-defined multi-block read operation, i.e., 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 SDHC after the pre-defined number of blocks are done, to drive the internal start response timeout. It is recommended to manually send an abort command with XFERTYP[RSPTYP] both bits cleared.

Chapter 54
Synchronous Audio Interface (SAI)

54.1 Introduction

NOTE

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

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, and CODEC/DSP interfaces.

54.1.1 Features

- Transmitter with independent Bit Clock and Frame Sync supporting 2 data channels
- Receiver with independent Bit Clock and Frame Sync supporting 2 data channels
- 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 8 × 32-bit FIFO for each Transmit and Receive Channel
- Graceful restart after FIFO Error

54.1.2 Block diagram

The following block diagram also shows the module clocks.

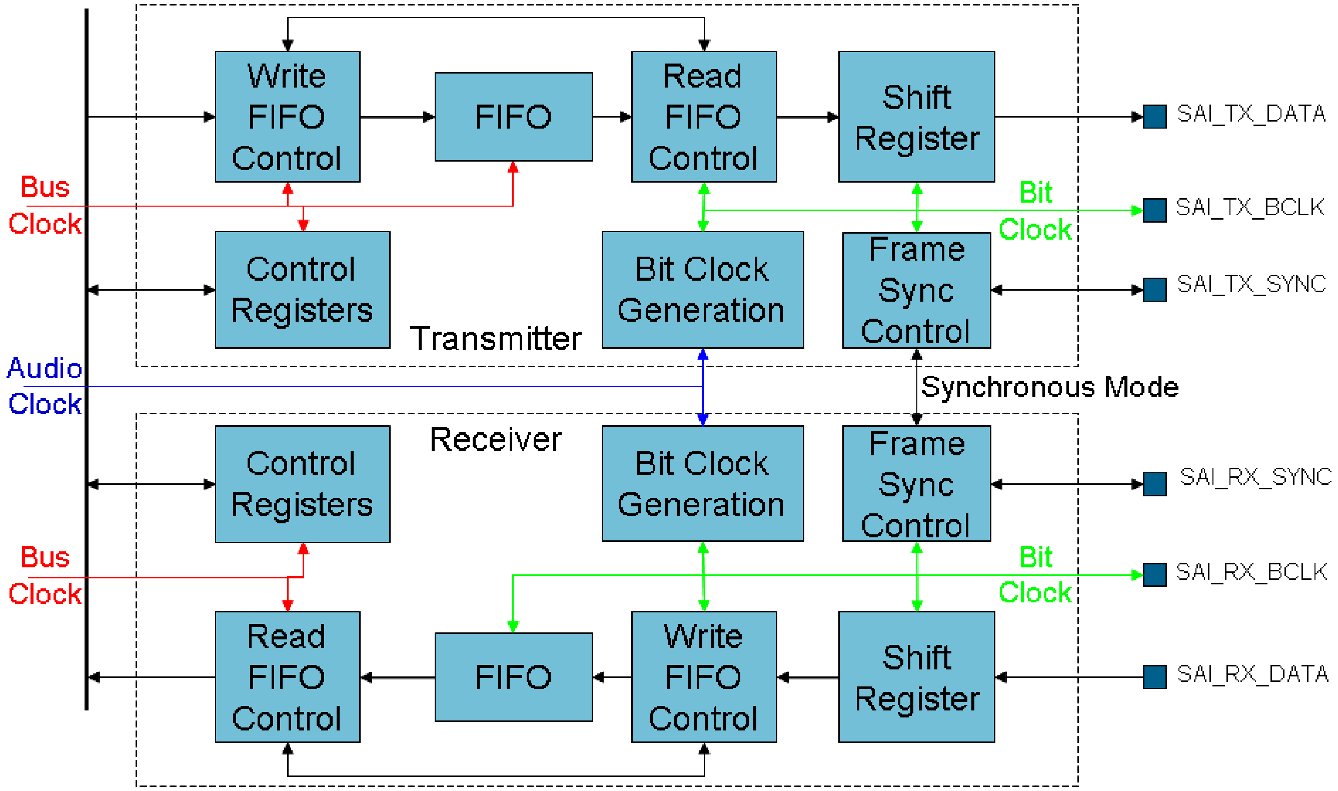


Figure 54-1. I²S/SAI block diagram

54.1.3 Modes of Operation

The module operates in these MCU power modes: run mode, stop modes, low-leakage modes, and debug mode.

54.1.3.1 Run mode

In run mode, the SAI transmitter and receiver operate normally.

54.1.3.2 Stop modes

In stop mode, the SAI transmitter and/or receiver can continue operating provided the 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.

54.1.3.3 Low-leakage modes

When entering low-leakage 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.

54.1.3.4 Debug mode

In debug mode, the SAI transmitter and/or receiver can continue operating provided the Debug Enable bit is set. When the Transmitter or Receiver Debug Enable (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.

54.2 External signals

Name	Function	I/O	Reset	Pull
SAI_TX_BCLK	Transmit Bit Clock	I/O	0	—
SAI_TX_SYNC	Transmit Frame Sync	I/O	0	—
SAI_TX_DATA[1:0]	Transmit Data	O	0	—
SAI_RX_BCLK	Receive Bit Clock	I/O	0	—
SAI_RX_SYNC	Receive Frame Sync	I/O	0	—
SAI_RX_DATA[1:0]	Receive Data	I	0	—
SAI_MCLK	Audio Master Clock	I/O	0	—

54.3 Memory Map and Register Definition

A read or write access to an address after the last register will result in a bus error.

I2S memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_F000	SAI Transmit Control Register (I2S0_TCSR)	32	R/W	0000_0000h	54.3.1/ 1835
4002_F004	SAI Transmit Configuration 1 Register (I2S0_TCR1)	32	R/W	0000_0000h	54.3.2/ 1838
4002_F008	SAI Transmit Configuration 2 Register (I2S0_TCR2)	32	R/W	0000_0000h	54.3.3/ 1838
4002_F00C	SAI Transmit Configuration 3 Register (I2S0_TCR3)	32	R/W	0000_0000h	54.3.4/ 1840
4002_F010	SAI Transmit Configuration 4 Register (I2S0_TCR4)	32	R/W	0000_0000h	54.3.5/ 1840
4002_F014	SAI Transmit Configuration 5 Register (I2S0_TCR5)	32	R/W	0000_0000h	54.3.6/ 1842
4002_F020	SAI Transmit Data Register (I2S0_TDR0)	32	W (always reads zero)	0000_0000h	54.3.7/ 1843
4002_F024	SAI Transmit Data Register (I2S0_TDR1)	32	W (always reads zero)	0000_0000h	54.3.7/ 1843
4002_F040	SAI Transmit FIFO Register (I2S0_TFR0)	32	R	0000_0000h	54.3.8/ 1843
4002_F044	SAI Transmit FIFO Register (I2S0_TFR1)	32	R	0000_0000h	54.3.8/ 1843
4002_F060	SAI Transmit Mask Register (I2S0_TMR)	32	R/W	0000_0000h	54.3.9/ 1844
4002_F080	SAI Receive Control Register (I2S0_RCSR)	32	R/W	0000_0000h	54.3.10/ 1845
4002_F084	SAI Receive Configuration 1 Register (I2S0_RCR1)	32	R/W	0000_0000h	54.3.11/ 1848
4002_F088	SAI Receive Configuration 2 Register (I2S0_RCR2)	32	R/W	0000_0000h	54.3.12/ 1848
4002_F08C	SAI Receive Configuration 3 Register (I2S0_RCR3)	32	R/W	0000_0000h	54.3.13/ 1850

Table continues on the next page...

I2S memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4002_F090	SAI Receive Configuration 4 Register (I2S0_RCR4)	32	R/W	0000_0000h	54.3.14/ 1850
4002_F094	SAI Receive Configuration 5 Register (I2S0_RCR5)	32	R/W	0000_0000h	54.3.15/ 1852
4002_F0A0	SAI Receive Data Register (I2S0_RDR0)	32	R	0000_0000h	54.3.16/ 1853
4002_F0A4	SAI Receive Data Register (I2S0_RDR1)	32	R	0000_0000h	54.3.16/ 1853
4002_F0C0	SAI Receive FIFO Register (I2S0_RFR0)	32	R	0000_0000h	54.3.17/ 1853
4002_F0C4	SAI Receive FIFO Register (I2S0_RFR1)	32	R	0000_0000h	54.3.17/ 1853
4002_F0E0	SAI Receive Mask Register (I2S0_RMR)	32	R/W	0000_0000h	54.3.18/ 1854
4002_F100	SAI MCLK Control Register (I2S0_MCR)	32	R/W	0000_0000h	54.3.19/ 1855
4002_F104	MCLK Divide Register (I2S0_MDR)	32	R/W	0000_0000h	54.3.20/ 1856
400A_F000	SAI Transmit Control Register (I2S1_TCSR)	32	R/W	0000_0000h	54.3.1/ 1835
400A_F004	SAI Transmit Configuration 1 Register (I2S1_TCR1)	32	R/W	0000_0000h	54.3.2/ 1838
400A_F008	SAI Transmit Configuration 2 Register (I2S1_TCR2)	32	R/W	0000_0000h	54.3.3/ 1838
400A_F00C	SAI Transmit Configuration 3 Register (I2S1_TCR3)	32	R/W	0000_0000h	54.3.4/ 1840
400A_F010	SAI Transmit Configuration 4 Register (I2S1_TCR4)	32	R/W	0000_0000h	54.3.5/ 1840
400A_F014	SAI Transmit Configuration 5 Register (I2S1_TCR5)	32	R/W	0000_0000h	54.3.6/ 1842
400A_F020	SAI Transmit Data Register (I2S1_TDR0)	32	W (always reads zero)	0000_0000h	54.3.7/ 1843
400A_F024	SAI Transmit Data Register (I2S1_TDR1)	32	W (always reads zero)	0000_0000h	54.3.7/ 1843
400A_F040	SAI Transmit FIFO Register (I2S1_TFR0)	32	R	0000_0000h	54.3.8/ 1843
400A_F044	SAI Transmit FIFO Register (I2S1_TFR1)	32	R	0000_0000h	54.3.8/ 1843

Table continues on the next page...

I2S memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400A_F060	SAI Transmit Mask Register (I2S1_TMR)	32	R/W	0000_0000h	54.3.9/ 1844
400A_F080	SAI Receive Control Register (I2S1_RCSR)	32	R/W	0000_0000h	54.3.10/ 1845
400A_F084	SAI Receive Configuration 1 Register (I2S1_RCR1)	32	R/W	0000_0000h	54.3.11/ 1848
400A_F088	SAI Receive Configuration 2 Register (I2S1_RCR2)	32	R/W	0000_0000h	54.3.12/ 1848
400A_F08C	SAI Receive Configuration 3 Register (I2S1_RCR3)	32	R/W	0000_0000h	54.3.13/ 1850
400A_F090	SAI Receive Configuration 4 Register (I2S1_RCR4)	32	R/W	0000_0000h	54.3.14/ 1850
400A_F094	SAI Receive Configuration 5 Register (I2S1_RCR5)	32	R/W	0000_0000h	54.3.15/ 1852
400A_F0A0	SAI Receive Data Register (I2S1_RDR0)	32	R	0000_0000h	54.3.16/ 1853
400A_F0A4	SAI Receive Data Register (I2S1_RDR1)	32	R	0000_0000h	54.3.16/ 1853
400A_F0C0	SAI Receive FIFO Register (I2S1_RFR0)	32	R	0000_0000h	54.3.17/ 1853
400A_F0C4	SAI Receive FIFO Register (I2S1_RFR1)	32	R	0000_0000h	54.3.17/ 1853
400A_F0E0	SAI Receive Mask Register (I2S1_RMR)	32	R/W	0000_0000h	54.3.18/ 1854
400A_F100	SAI MCLK Control Register (I2S1_MCR)	32	R/W	0000_0000h	54.3.19/ 1855
400A_F104	MCLK Divide Register (I2S1_MDR)	32	R/W	0000_0000h	54.3.20/ 1856

54.3.1 SAI Transmit Control Register (I2Sx_TCSR)

Addresses: I2S0_TCSR is 4002_F000h base + 0h offset = 4002_F000h

I2S1_TCSR is 400A_F000h base + 0h offset = 400A_F000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TE	STOPE	DBGE	BCE	0		0	SR		0		WSF	SEF	FEF	FWF	FRF
W							FR					w1c	w1c	w1c		
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		WSIE	SEIE	FEIE	FWIE	FRIE		0			0		FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2Sx_TCSR field descriptions

Field	Description
31 TE	Transmitter enable Enables/disables the transmitter. When software clears this bit, the transmitter remains enabled (and this bit remains set) until the end of the current frame. 0 Transmitter is disabled. 1 Transmitter is enabled, or transmitter has been disabled and not end of frame.
30 STOPE	Stop enable Configures transmitter operation in Stop mode. This bit is ignored and the transmitter is disabled in all low-leakage stop modes. 0 Transmitter disabled in stop mode. 1 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. 0 Transmitter is disabled in debug mode, after completing the current frame. 1 Transmitter is enabled in debug mode.
28 BCE	Bit Clock Enable Enables the transmit bit clock, separately from the transmit enable. This bit is automatically set whenever the transmit enable is set. When software clears this bit, the transmit bit clock remains enabled (and this bit remains set) until the end of the current frame.

Table continues on the next page...

I2Sx_TCSR field descriptions (continued)

Field	Description
	0 Transmit bit clock is disabled 1 Transmit bit clock is enabled
27–26 Reserved	This read-only field is reserved and always has the value zero.
25 FR	FIFO reset Resets the FIFO pointers. Reading this bit will always return zero. 0 No effect. 1 FIFO reset.
24 SR	Software reset When set, resets the internal transmitter logic including the FIFO pointers. Software visible-registers are not affected, except for the status registers. 0 No effect. 1 Software reset.
23–21 Reserved	This read-only field is reserved and always has the value zero.
20 WSF	Word start flag Indicates that the start of the configured word has been detected. Write a logic one to this register bit to clear this flag. 0 Start of word not detected. 1 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 one to this register bit to clear this flag. 0 Sync error not detected. 1 Frame sync error detected.
18 FEF	FIFO error flag Indicates that an enabled transmit FIFO has underrun. Write a logic one to this register bit to clear this flag. 0 Transmit underrun not detected. 1 Transmit underrun detected.
17 FWF	FIFO warning flag Indicates that an enabled transmit FIFO is empty. 0 No enabled transmit FIFO is empty. 1 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.

Table continues on the next page...

I2Sx_TCSR field descriptions (continued)

Field	Description
	0 Transmit FIFO watermark not reached. 1 Transmit FIFO watermark has been reached.
15–13 Reserved	This read-only field is reserved and always has the value zero.
12 WSIE	Word start interrupt enable Enables/disables word start interrupts. 0 Disables interrupt. 1 Enables interrupt.
11 SEIE	Sync error interrupt enable Enables/disables sync error interrupts. 0 Disables interrupt. 1 Enables interrupt.
10 FEIE	FIFO error interrupt enable Enables/disables FIFO error interrupts. 0 Disables the interrupt, 1 Enables the interrupt.
9 FWIE	FIFO warning interrupt enable Enables/disables FIFO warning interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
8 FRIE	FIFO request interrupt enable Enables/disables FIFO request interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
7–5 Reserved	This read-only field is reserved and always has the value zero.
4–2 Reserved	This read-only field is reserved and always has the value zero.
1 FWDE	FIFO warning DMA enable Enables/disables DMA requests. 0 Disables the DMA request. 1 Enables the DMA request.
0 FRDE	FIFO request DMA enable Enables/disables DMA requests.

Table continues on the next page...

I2Sx_TCSR field descriptions (continued)

Field	Description
	0 Disables the DMA request. 1 Enables the DMA request.

54.3.2 SAI Transmit Configuration 1 Register (I2Sx_TCR1)

Addresses: I2S0_TCR1 is 4002_F000h base + 4h offset = 4002_F004h
I2S1_TCR1 is 400A_F000h base + 4h offset = 400A_F004h

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																													TFW		
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

I2Sx_TCR1 field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value zero.
2–0 TFW	Transmit FIFO watermark Configures the watermark level for all enabled transmit channels.

54.3.3 SAI Transmit Configuration 2 Register (I2Sx_TCR2)

This register should not be altered when the transmit enable bit is set.

Addresses: I2S0_TCR2 is 4002_F000h base + 8h offset = 4002_F008h
I2S1_TCR2 is 400A_F000h base + 8h offset = 400A_F008h

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	SYNC		BCS	BCI	MSEL		BCP	BCD	0																DIV							
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

I2Sx_TCR2 field descriptions

Field	Description
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...

I2Sx_TCR2 field descriptions (continued)

Field	Description
	00 Asynchronous mode. 01 Synchronous with receiver. 10 Synchronous with another SAI transmitter. 11 Synchronous with another SAI receiver.
29 BCS	Bit Clock Swap When set in asynchronous mode, the transmitter is clocked by the receiver bit clock. When set in synchronous mode, the transmitter is clocked by the transmitter bit clock, but uses the receiver frame sync. This bit has no effect when synchronous with another SAI peripheral. 0 Use the normal bit clock source. 1 Swap the bit clock source.
28 BCI	Bit Clock Input When set in either asynchronous or synchronous mode and using an internally generated bit clock, configures the internal logic to be clocked as if the bit clock was externally generated. This has the effect of decreasing data input setup time, but increasing data output valid time. This bit has no effect when configured for an externally generated bit clock or when synchronous to another SAI peripheral. 0 No effect. 1 Internal logic is clocked by external bit clock.
27–26 MSEL	MCLK Select Selects the Audio Master Clock used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. 00 Bus Clock selected. 01 Master Clock 1 selected. 10 Master Clock 2 selected. 11 Master Clock 3 selected.
25 BCP	Bit clock polarity Configures the polarity of the bit clock. 0 Bit Clock is active high (drive outputs on rising edge and sample inputs on falling edge). 1 Bit Clock is active low (drive outputs on falling edge and sample inputs on rising edge).
24 BCD	Bit clock direction Configures the direction of the bit clock. 0 Bit clock is generated externally (slave mode). 1 Bit clock is generated internally (master mode).
23–8 Reserved	This read-only field is reserved and always has the value zero.
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.

54.3.4 SAI Transmit Configuration 3 Register (I2Sx_TCR3)

This register should not be altered when the transmit enable bit is set.

Addresses: I2S0_TCR3 is 4002_F000h base + Ch offset = 4002_F00Ch
I2S1_TCR3 is 400A_F000h base + Ch offset = 400A_F00Ch

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="14">0</td><td colspan="2" rowspan="2">TCE</td><td colspan="11">0</td><td colspan="5" rowspan="2">WDFL</td></tr>
<tr><td>W</td><td colspan="14"></td><td colspan="11"></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

I2Sx_TCR3 field descriptions

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value zero.
17–16 TCE	Transmit channel enable Enables a data channel for a transmit operation. A channel should be enabled before its FIFO is accessed.
15–5 Reserved	This read-only field is reserved and always has the value zero.
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.

54.3.5 SAI Transmit Configuration 4 Register (I2Sx_TCR4)

This register should not be altered when the transmit enable bit is set.

Addresses: I2S0_TCR4 is 4002_F000h base + 10h offset = 4002_F010h
I2S1_TCR4 is 400A_F000h base + 10h offset = 400A_F010h

<table>
<tr><td>Bit</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>R</td><td colspan="11">0</td><td colspan="5" rowspan="2">FRSZ</td><td colspan="3">0</td><td colspan="5" rowspan="2">SYWD</td><td colspan="3">0</td><td rowspan="2">MF</td><td rowspan="2">FSE</td><td>0</td><td rowspan="2">FSP</td><td rowspan="2">FSD</td></tr>
<tr><td>W</td><td colspan="11"></td><td colspan="3"></td><td colspan="3"></td><td></td></tr>
<tr><td>Reset</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

I2Sx_TCR4 field descriptions

Field	Description
31–21 Reserved	This read-only field is reserved and always has the value zero.
20–16 FRSZ	Frame size Configures the number of words in each frame. The value written should 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	This read-only field is reserved and always has the value zero.
12–8 SYWD	Sync width Configures the length of the frame sync in number of bit clocks. The value written should 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.
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 MF	MSB first Specifies whether the LSB or the MSB is transmitted/received first. 0 LBS is transmitted/received first. 1 MBS is transmitted/received first.
3 FSE	Frame sync early 0 Frame sync asserts with the first bit of the frame. 1 Frame sync asserts one bit before the first bit of the frame.
2 Reserved	This read-only field is reserved and always has the value zero.
1 FSP	Frame sync polarity Configures the polarity of the frame sync. 0 Frame sync is active high. 1 Frame sync is active low.
0 FSD	Frame sync direction Configures the direction of the frame sync. 0 Frame Sync is generated externally (slave mode). 1 Frame Sync is generated internally (master mode).

54.3.6 SAI Transmit Configuration 5 Register (I2Sx_TCR5)

This register should not be altered when the transmit enable bit is set.

Addresses: I2S0_TCR5 is 4002_F000h base + 14h offset = 4002_F014h
I2S1_TCR5 is 400A_F000h base + 14h offset = 400A_F014h

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			WNW					0			W0W					0			FBT					0							
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

I2Sx_TCR5 field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
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 should be one less than the number of bits per word. This field must be configured greater than or equal to Word 0 Width even when there is only one word in each frame. Words of fewer than 8 bits wide are not supported.
23–21 Reserved	This read-only field is reserved and always has the value zero.
20–16 W0W	Word 0 width Configures the number of bits in the first word in each frame. The value written should be one less than the number of bits in the first word. Words of less than 8 bits wide are not supported if there is only one word per frame.
15–13 Reserved	This read-only field is reserved and always has the value zero.
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 should be greater than or equal to the word width when configured for MSB First. The value written should be less than or equal to 31-word width when configured for LSB First.
7–0 Reserved	This read-only field is reserved and always has the value zero.

54.3.7 SAI Transmit Data Register (I2Sx_TDR)

Addresses: I2S0_TDR0 is 4002_F000h base + 20h offset = 4002_F020h
I2S0_TDR1 is 4002_F000h base + 24h offset = 4002_F024h
I2S1_TDR0 is 400A_F000h base + 20h offset = 400A_F020h
I2S1_TDR1 is 400A_F000h base + 24h offset = 400A_F024h

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																															
W	TDR[31:0]																															
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

I2Sx_TDR*n* field descriptions

Field	Description
31–0 TDR[31:0]	Transmit data register The corresponding transmit channel enable should be set before accessing a 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.

54.3.8 SAI Transmit FIFO Register (I2Sx_TFR)

The MSB of the read pointer and write pointer 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.

Addresses: I2S0_TFR0 is 4002_F000h base + 40h offset = 4002_F040h
I2S0_TFR1 is 4002_F000h base + 44h offset = 4002_F044h
I2S1_TFR0 is 400A_F000h base + 40h offset = 400A_F040h
I2S1_TFR1 is 400A_F000h base + 44h offset = 400A_F044h

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												WFP				0												RFP			
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

I2Sx_TFR*n* field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–16 WFP	Write FIFO pointer

Table continues on the next page...

I2Sx_TFR*n* field descriptions (continued)

Field	Description
	FIFO write pointer for transmit data channel.
15–4 Reserved	This read-only field is reserved and always has the value zero.
3–0 RFP	Read FIFO pointer FIFO read pointer for transmit data channel.

54.3.9 SAI Transmit Mask Register (I2S*x*_TMR)

This register is double-buffered and updates when the transmit enable bit is first set and then at the end of each frame. This allows the masked words in each frame to change from frame to frame.

Addresses: I2S0_TMR is 4002_F000h base + 60h offset = 4002_F060h
I2S1_TMR is 400A_F000h base + 60h offset = 400A_F060h

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	TWM																															
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

I2S*x*_TMR field descriptions

Field	Description
31–0 TWM	Transmit word mask For each word in the frame, configures whether the transmit word is masked. 0 Word N is enabled. 1 Word N is masked. The transmit data pins are tri-stated when masked.

54.3.10 SAI Receive Control Register (I2Sx_RCSR)

Addresses: I2S0_RCSR is 4002_F000h base + 80h offset = 4002_F080h
I2S1_RCSR is 400A_F000h base + 80h offset = 400A_F080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RE	STOPE	DBGE	BCE	0		0	SR	0			WSF	SEF	FEF	FWF	FRF
W							FR					w1c	w1c	w1c		
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			WSIE	SEIE	FEIE	FWIE	FRIE	0			0			FWDE	FRDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2Sx_RCSR field descriptions

Field	Description
31 RE	Receiver enable Enables/disables the receiver. When software clears this bit, the receiver remains enabled (and this bit remains set) until the end of the current frame. 0 Receiver is disabled. 1 Receiver is enabled, or receiver has been disabled and not end of frame.
30 STOPE	Stop enable Configures receiver operation in Stop mode. This bit is ignored and the receiver is disabled in all low-leakage stop modes. 0 Receiver disabled in stop mode. 1 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. 0 Receiver is disabled in debug mode, after completing the current frame. 1 Receiver is enabled in debug mode.
28 BCE	Bit Clock enable Enables the receive bit clock, separately from the receive enable. This bit is automatically set whenever the receive enable is set. When software clears this bit, the receive bit clock remains enabled (and this bit remains set) until the end of the current frame. 0 Receive bit clock is disabled 1 Receive bit clock is enabled

Table continues on the next page...

I2Sx_RCSR field descriptions (continued)

Field	Description
27–26 Reserved	This read-only field is reserved and always has the value zero.
25 FR	FIFO reset Resets the FIFO pointers. Reading this bit will always return zero. 0 No effect. 1 FIFO reset.
24 SR	Software reset When set, resets the internal receiver logic including the FIFO pointers. Software visible-registers are not affected, except for the status registers. 0 No effect. 1 Software reset.
23–21 Reserved	This read-only field is reserved and always has the value zero.
20 WSF	Word start flag Indicates that the start of the configured word has been detected. Write a logic one to this register bit to clear this flag. 0 Start of word not detected. 1 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 one to this register bit to clear this flag. 0 Sync error not detected. 1 Frame sync error detected.
18 FEF	FIFO error flag Indicates that an enabled receive FIFO has overflowed. Write a logic one to this register bit to clear this flag. 0 Receive overflow not detected. 1 Receive overflow detected.
17 FWF	FIFO warning flag Indicates that an enabled receive FIFO is full. 0 No enabled receive FIFO is full. 1 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.

Table continues on the next page...

I2Sx_RCSR field descriptions (continued)

Field	Description
	0 Receive FIFO watermark not reached. 1 Receive FIFO watermark has been reached.
15–13 Reserved	This read-only field is reserved and always has the value zero.
12 WSIE	Word start interrupt enable Enables/disables word start interrupts. 0 Disables interrupt. 1 Enables interrupt.
11 SEIE	Sync error interrupt enable Enables/disables sync error interrupts. 0 Disables interrupt. 1 Enables interrupt.
10 FEIE	FIFO error interrupt enable Enables/disables FIFO error interrupts. 0 Disables the interrupt, 1 Enables the interrupt.
9 FWIE	FIFO warning interrupt enable Enables/disables FIFO warning interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
8 FRIE	FIFO request interrupt enable Enables/disables FIFO request interrupts. 0 Disables the interrupt. 1 Enables the interrupt.
7–5 Reserved	This read-only field is reserved and always has the value zero.
4–2 Reserved	This read-only field is reserved and always has the value zero.
1 FWDE	FIFO warning DMA enable Enables/disables DMA requests. 0 Disables the DMA request. 1 Enables the DMA request.
0 FRDE	FIFO request DMA enable Enables/disables DMA requests.

Table continues on the next page...

I2Sx_RCSR field descriptions (continued)

Field	Description
	0 Disables the DMA request. 1 Enables the DMA request.

54.3.11 SAI Receive Configuration 1 Register (I2Sx_RCR1)

Addresses: I2S0_RCR1 is 4002_F000h base + 84h offset = 4002_F084h
I2S1_RCR1 is 400A_F000h base + 84h offset = 400A_F084h

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																													RFW		
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

I2Sx_RCR1 field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value zero.
2–0 RFW	Receive FIFO watermark Configures the watermark level for all enabled receiver channels.

54.3.12 SAI Receive Configuration 2 Register (I2Sx_RCR2)

This register should not be altered when the receive enable bit is set.

Addresses: I2S0_RCR2 is 4002_F000h base + 88h offset = 4002_F088h
I2S1_RCR2 is 400A_F000h base + 88h offset = 400A_F088h

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	SYNC		BCS	BCI	MSEL		BCP	BCD	0																DIV							
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

I2Sx_RCR2 field descriptions

Field	Description
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...

I2Sx_RCR2 field descriptions (continued)

Field	Description
	00 Asynchronous mode. 01 Synchronous with transmitter. 10 Synchronous with another SAI receiver. 11 Synchronous with another SAI transmitter.
29 BCS	Bit Clock Swap When set in asynchronous mode, the receiver is clocked by the transmitter bit clock. When set in synchronous mode, the receiver is clocked by the receiver bit clock, but uses the transmitter frame sync. This bit has no effect when synchronous with another SAI peripheral. 0 Use the normal bit clock source. 1 Swap the bit clock source.
28 BCI	Bit Clock Input When set in either asynchronous or synchronous mode and using an internally generated bit clock, configures the internal logic to be clocked as if the bit clock was externally generated. This has the effect of decreasing data input setup time, but increasing data output valid time. This bit has no effect when configured for an externally generated bit clock or when synchronous to another SAI peripheral. 0 No effect. 1 Internal logic is clocked as if bit clock was externally generated.
27–26 MSEL	MCLK Select Selects the Audio Master Clock used to generate an internally generated bit clock. This field has no effect when configured for an externally generated bit clock. 00 Bus Clock selected. 01 Master Clock 1 selected. 10 Master Clock 2 selected. 11 Master Clock 3 selected.
25 BCP	Bit clock polarity Configures the polarity of the bit clock. 0 Bit Clock is active high (drive outputs on rising edge and sample inputs on falling edge). 1 Bit Clock is active low (drive outputs on falling edge and sample inputs on rising edge).
24 BCD	Bit clock direction Configures the direction of the bit clock. 0 Bit clock is generated externally (slave mode). 1 Bit clock is generated internally (master mode).
23–8 Reserved	This read-only field is reserved and always has the value zero.
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.

54.3.13 SAI Receive Configuration 3 Register (I2Sx_RCR3)

This register should not be altered when the receive enable bit is set.

Addresses: I2S0_RCR3 is 4002_F000h base + 8Ch offset = 4002_F08Ch
I2S1_RCR3 is 400A_F000h base + 8Ch offset = 400A_F08Ch

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														RCE		0											WDFL				
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

I2Sx_RCR3 field descriptions

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value zero.
17–16 RCE	Receive channel enable Enables a data channel for a receive operation. A channel should be enabled before its FIFO is accessed.
15–5 Reserved	This read-only field is reserved and always has the value zero.
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.

54.3.14 SAI Receive Configuration 4 Register (I2Sx_RCR4)

This register should not be altered when the receive enable bit is set.

Addresses: I2S0_RCR4 is 4002_F000h base + 90h offset = 4002_F090h
I2S1_RCR4 is 400A_F000h base + 90h offset = 400A_F090h

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											FRSZ					0			SYWD					0			MF	FSE	0	FSP	FSD
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

I2Sx_RCR4 field descriptions

Field	Description
31–21 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

I2Sx_RCR4 field descriptions (continued)

Field	Description
20–16 FRSZ	Frame size Configures the number of words in each frame. The value written should 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	This read-only field is reserved and always has the value zero.
12–8 SYWD	Sync width Configures the length of the frame sync in number of bit clocks. The value written should 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.
7–5 Reserved	This read-only field is reserved and always has the value zero.
4 MF	MSB first Specifies whether the LSB or the MSB is transmitted/received first. 0 LBS is transmitted/received first. 1 MBS is transmitted/received first.
3 FSE	Frame sync early 0 Frame sync asserts with the first bit of the frame. 1 Frame sync asserts one bit before the first bit of the frame.
2 Reserved	This read-only field is reserved and always has the value zero.
1 FSP	Frame sync polarity Configures the polarity of the frame sync. 0 Frame sync is active high. 1 Frame sync is active low.
0 FSD	Frame sync direction Configures the direction of the frame sync. 0 Frame Sync is generated externally (slave mode). 1 Frame Sync is generated internally (master mode).

54.3.15 SAI Receive Configuration 5 Register (I2Sx_RCR5)

This register should not be altered when the receive enable bit is set.

Addresses: I2S0_RCR5 is 4002_F000h base + 94h offset = 4002_F094h
I2S1_RCR5 is 400A_F000h base + 94h offset = 400A_F094h

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			WNW					0			W0W					0			FBT					0							
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

I2Sx_RCR5 field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
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 should be one less than the number of bits per word. This field must be configured greater than or equal to Word 0 Width even when there is only one word in each frame. Words of fewer than 8 bits wide are not supported.
23–21 Reserved	This read-only field is reserved and always has the value zero.
20–16 W0W	Word 0 width Configures the number of bits in the first word in each frame. The value written should be one less than the number of bits in the first word. Words of less than 8 bits wide are not supported if there is only one word per frame.
15–13 Reserved	This read-only field is reserved and always has the value zero.
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 should be greater than or equal to the word width when configured for MSB First. The value written should be less than or equal to (31 - word with) when configured for LSB First.
7–0 Reserved	This read-only field is reserved and always has the value zero.

54.3.16 SAI Receive Data Register (I2Sx_RDR)

Reading this register introduces one additional peripheral clock wait state on each read.

Addresses: I2S0_RDR0 is 4002_F000h base + A0h offset = 4002_F0A0h
I2S0_RDR1 is 4002_F000h base + A4h offset = 4002_F0A4h
I2S1_RDR0 is 400A_F000h base + A0h offset = 400A_F0A0h
I2S1_RDR1 is 400A_F000h base + A4h offset = 400A_F0A4h

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	RDR[31:0]																															
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

I2Sx_RDR*n* field descriptions

Field	Description
31–0 RDR[31:0]	Receive data register The corresponding receive channel enable should be set before accessing a 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.

54.3.17 SAI Receive FIFO Register (I2Sx_RFR)

The MSB of the read pointer and write pointer 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.

Addresses: I2S0_RFR0 is 4002_F000h base + C0h offset = 4002_F0C0h
I2S0_RFR1 is 4002_F000h base + C4h offset = 4002_F0C4h
I2S1_RFR0 is 400A_F000h base + C0h offset = 400A_F0C0h
I2S1_RFR1 is 400A_F000h base + C4h offset = 400A_F0C4h

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												WFP				0												RFP			
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

I2Sx_RFR*n* field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.

Table continues on the next page...

I2Sx_RFR*n* field descriptions (continued)

Field	Description
19–16 WFP	Write FIFO pointer FIFO write pointer for receive data channel.
15–4 Reserved	This read-only field is reserved and always has the value zero.
3–0 RFP	Read FIFO pointer FIFO read pointer for receive data channel.

54.3.18 SAI Receive Mask Register (I2Sx_RMR)

This register is double-buffered and updates when the receive enable bit is first set and then at the end of each frame. This allows the masked words in each frame to change from frame to frame.

Addresses: I2S0_RMR is 4002_F000h base + E0h offset = 4002_F0E0h
I2S1_RMR is 400A_F000h base + E0h offset = 400A_F0E0h

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	RWM																															
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

I2Sx_RMR field descriptions

Field	Description
31–0 RWM	Receive word mask For each word in the frame, configures if the receive word is masked. 0 Word N is enabled. 1 Word N is masked.

54.3.19 SAI MCLK Control Register (I2Sx_MCR)

The MCLK Control Register controls the clock source and direction of the Audio Master Clock.

Addresses: I2S0_MCR is 4002_F000h base + 100h offset = 4002_F100h

I2S1_MCR is 400A_F000h base + 100h offset = 400A_F100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DUF	MOE	0				MICS		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															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

I2Sx_MCR field descriptions

Field	Description
31 DUF	Divider Update Flag Provides the status of on-the-fly updates to the MCLK Divider ratio. 0 MCLK Divider ratio is not being updated currently. 1 MCLK Divider ratio is updating on-the-fly. Furthur updates to the MCLK Divider ratio are blocked while this flag remains set.
30 MOE	MCLK Output Enable Enables the MCLK Divider and configures the SAI_MCLK pin as an output. When software clears this bit, this bit remains set until the MCLK divider is fully disabled. 0 SAI_MCLK pin is configured as an input that bypasses the MCLK Divider. 1 SAI_MCLK pin is configured as an output from the MCLK Divider and the MCLK Divider is enabled.
29–26 Reserved	This read-only field is reserved and always has the value zero.
25–24 MICS	MCLK Input Clock Select Selects the clock input to the MCLK Divider. This field cannot be changed when the MCLK divider is enabled. See the Chip Configuration details for information about the connections to these inputs. 00 MCLK Divider input clock 0 selected. 01 MCLK Divider input clock 1 selected.

Table continues on the next page...

I2Sx_MCR field descriptions (continued)

Field	Description
	10 MCLK Divider input clock 2 selected. 11 MCLK Divider input clock 3 selected.
23–0 Reserved	This read-only field is reserved and always has the value zero.

54.3.20 MCLK Divide Register (I2Sx_MDR)

Configures the MCLK Divide Ratio. Although the MCLK Divide Register can be changed when the MCLK divided clock is enabled, additional writes to the MCLK Divide Register are blocked while the Divider Update Flag is set. Writes to the MCLK Divide Register when the MCLK divided clock is disabled do not set the Divider Update Flag.

Addresses: I2S0_MDR is 4002_F000h base + 104h offset = 4002_F104h

I2S1_MDR is 400A_F000h base + 104h offset = 400A_F104h

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												FRACT								DIVIDE											
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

I2Sx_MDR field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–12 FRACT	MCLK Fraction The MCLK FRACT must be set equal or less than the MCLK DIVIDE. Sets the MCLK divide ratio such that: MCLK output = MCLK input * ((FRACT + 1) / (DIVIDE + 1))
11–0 DIVIDE	MCLK Divide The MCLK FRACT must be set equal or less than the MCLK DIVIDE. Sets the MCLK divide ratio such that: MCLK output = MCLK input * ((FRACT + 1) / (DIVIDE + 1))

54.4 Functional description

54.4.1 SAI clocking

The SAI clocks include the audio master clock, the bit clock, and the bus clock.

54.4.1.1 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.

Each SAI peripheral can control the input clock selection, pin direction and divide ratio of one audio master clock. The input clock selection and pin direction cannot be altered if an SAI using that audio master clock has been enabled. The master clock divide ratio can be altered while an SAI is using that master clock, although the change in the divide ratio takes several cycles. The divide update flag can be polled to determine when the divide ratio change has completed.

The audio master clock generation and selection is chip specific. Refer to chip-specific clocking information about how the audio master clocks are generated. A typical implementation appears in the following figure.

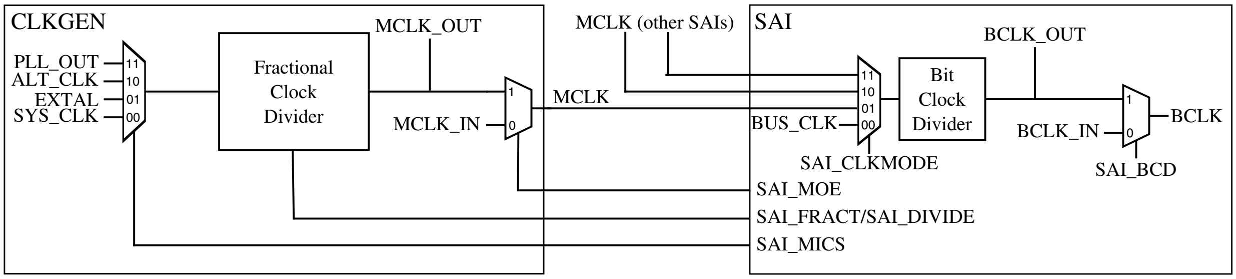


Figure 54-86. SAI Master Clock Generation

54.4.1.2 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.

Externally generated bit clocks should be enabled before the SAI transmitter or receiver is enabled and should be disabled after the SAI transmitter or receiver is disabled and they have completed their current frames.

54.4.1.3 Bus Clock

The bus clock is used by the control and configuration registers and to generate synchronous interrupts and DMA requests.

54.4.2 SAI resets

The SAI is asynchronously reset on system reset. The SAI has a software reset and a FIFO reset.

54.4.2.1 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.

54.4.2.2 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 the Transmit FIFO Error Flag is set, and before the FIFO is re-initialized and the Error Flag is cleared. The FIFO Reset is asserted for one cycle only.

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 Receive FIFO Error Flag is set and any remaining data has been read from the FIFO, and before the Error Flag is cleared. The FIFO Reset is asserted for one cycle only.

54.4.3 Synchronous Modes

The SAI transmitter and receiver can operate synchronously to each other or synchronously to other SAI peripherals.

54.4.3.1 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 should be configured for asynchronous operation and the receiver for synchronous operation. In synchronous mode, the receiver is only enabled when both the transmitter and receiver are both 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 should be configured for asynchronous operation and the transmitter for synchronous operation. In synchronous mode, the transmitter is only enabled 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 should be configured consistently across both the transmitter and receiver.

54.4.3.2 Multiple SAI Synchronous Mode

Synchronous operation between multiple SAI peripherals is not supported on all devices, and 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 should be configured consistently across both the transmitter and receiver.

54.4.4 Frame sync configuration

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
- Asserts with first bit in frame or asserts one bit early
- Asserts for between 1 bit clock and first word length
- Frame length can be configured from 1 word per frame to 32 words per frame
- Word length can be configured to support from 8 bits to 32 bits per word
    - First word length and remaining word lengths can be configured separately
- Can be configured for Most Significant Bit first or Least Significant Bit first

These configuration options cannot be changed after the SAI transmitter or receiver is enabled.

54.4.5 Data FIFO

54.4.5.1 Data alignment

Each transmit and receive channel includes a FIFO of size 8 × 32-bit. The FIFO data is accessed using the SAI Transmit/Receive Data Registers. 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 54-87 for LSB First configurations and Figure 54-88 for MSB First configurations.

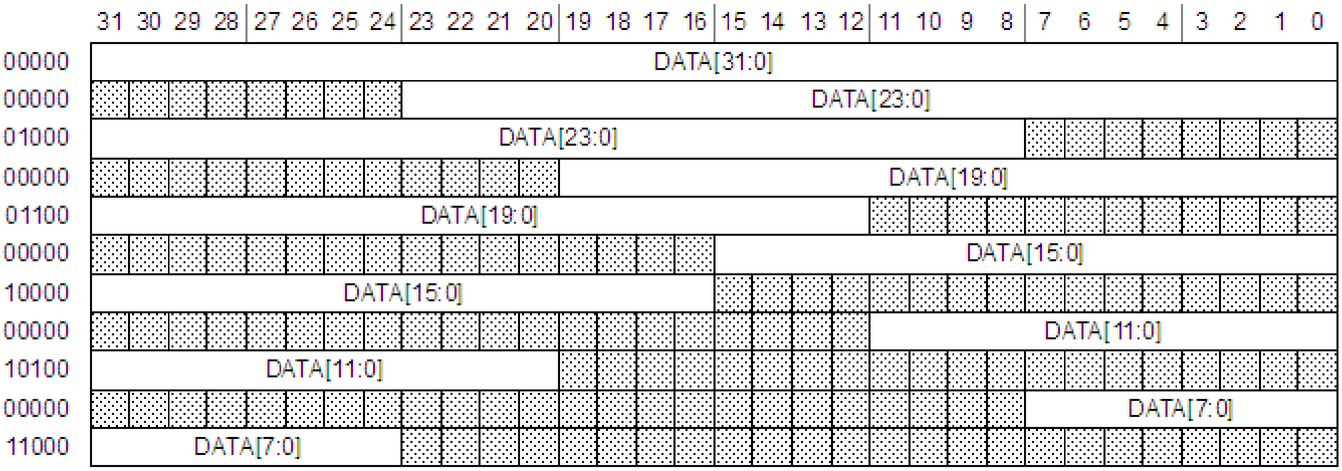


Figure 54-87. SAI First Bit Shifted, LSB First

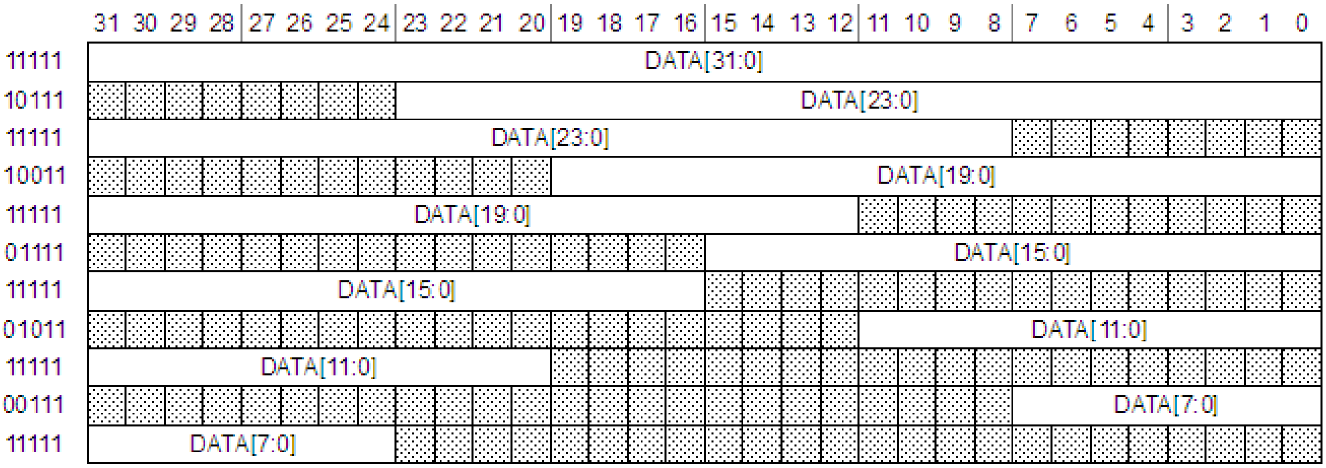


Figure 54-88. SAI First Bit Shifted, MSB First

54.4.5.2 FIFO pointers

When writing to the Transmit Data Register (TDR), the write FIFO pointer increments after each valid write. The SAI supports 8-bit and 16-bit writes to TDR for transmitting 8-bit and 16-bit data respectively.

Writes to the Transmit Data Register are ignored if the corresponding Transmit Channel Enable is clear or if the FIFO is full. If the Transmit FIFO is empty, the Transmit Data Register must be written at least three bit clocks before the start of the next unmasked word to avoid a FIFO underrun.

When reading the Receive Data Register (RDR), the read FIFO pointer increments after each valid read. The SAI supports 8-bit and 16-bit reads from RDR for receiving 8-bit and 16-bit data respectively.

Reads from the Receive Data Register are ignored if the corresponding Receive Channel Enable is clear or if the FIFO is empty. If the Receive FIFO is full, the Receive Data Register must be read at least three bit clocks before the end of an unmasked word to avoid a FIFO overrun.

54.4.6 Word mask register

The SAI transmitter and receiver each contain a word mask register that can be used to mask any word in the frame. Since 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 transmitter word mask 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 receiver word mask causes the received data for each selected word to be discarded and not written to the receive FIFO.

54.4.7 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.

54.4.7.1 FIFO data ready flag

The FIFO data ready flag is set based on the number of entries in the FIFO and the FIFO watermark configuration.

The transmit data ready 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 data ready 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 data ready flag can generate an interrupt or a DMA request.

54.4.7.2 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.

54.4.7.3 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 repeat the last valid word read from the transmit FIFO until the transmit FIFO error flag is cleared and the start of the next transmit frame. All enabled transmit FIFOs should be reset and initialized with new data before the transmit FIFO error flag is cleared.

The receive FIFO error flag is set when the any of the enabled receive FIFOs overflow. After it is set, all enabled receive channels discard received data until the receive FIFO error flag is cleared and the start of the next receive frame. All enabled receive FIFOs should be emptied before the receive FIFO error flag is cleared.

The FIFO error flag can generate an interrupt only.

54.4.7.4 Sync error flag

The sync error flag 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. The transmitter or receiver continues checking for frame sync assertion at the end of each frame (or when idle) when the sync error flag is set.

The sync error flag can generate an interrupt only.

54.4.7.5 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 55
General purpose input/output (GPIO)

55.1 Introduction

NOTE

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

The general purpose input and output (GPIO) module interfaces to the processor core via a zero wait state interface for maximum pin performance. Accesses of any data size are supported to the GPIO registers.

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 banging of the general purpose outputs is supported through the addition of set, clear and toggle write-only registers for each port output data register.

55.1.1 Features

- Rapid general purpose input and output
  - Pin input data register visible in all digital pin-muxing modes
  - Pin output data register with corresponding set/clear/toggle registers
  - Pin data direction register
  - Zero wait state access to GPIO registers

55.1.2 Modes of operation

55.1.2.1 Run mode

In run mode, the GPIO operates normally.

55.1.2.2 Wait mode

In wait mode, the GPIO operates normally.

55.1.2.3 Stop mode

The GPIO is disabled in stop mode, although the pins retain their state.

55.1.2.4 Debug mode

In debug mode, the GPIO operates normally.

55.1.3 GPIO signal descriptions

Table 55-1. GPIO signal descriptions

Signal	Description	I/O
PORTA[31:0]	General purpose input/output	I/O
PORTB[31:0]	General purpose input/output	I/O
PORTC[31:0]	General purpose input/output	I/O
PORTD[31:0]	General purpose input/output	I/O
PORTE[31:0]	General purpose input/output	I/O
PORTF[31:0]	General purpose input/output	I/O

NOTE

Not all pins within each port are implemented on each device. Refer to the Signal Multiplexing chapter for the number of GPIO ports available in the device.

55.1.3.1 Detailed signal description

Table 55-2. GPIO interface-detailed signal descriptions

Signal	I/O	Description	
PORTA[31:0] PORTB[31:0] PORTC[31:0] PORTD[31:0] PORTE[31:0] PORTF[31:0]	I/O	General purpose input/output.	
		State meaning	Asserted - pin is logic one. Negated - pin is logic zero.
		Timing	Assertion - when output, occurs on rising edge of the system clock. For input, may occur at any time and input may be asserted asynchronously to the system clock. Negation - when output, occurs on rising edge of the system clock. For input, may occur at any time and input may be asserted asynchronously to the system clock.

55.2 Memory map and register definition

Any read or write access to the GPIO 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.

GPIO memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F000	Port Data Output Register (GPIOA_PDOR)	32	R/W	0000_0000h	55.2.1/ 1870
400F_F004	Port Set Output Register (GPIOA_PSOR)	32	W (always reads zero)	0000_0000h	55.2.2/ 1871
400F_F008	Port Clear Output Register (GPIOA_PCOR)	32	W (always reads zero)	0000_0000h	55.2.3/ 1871
400F_F00C	Port Toggle Output Register (GPIOA_PTOR)	32	W (always	0000_0000h	55.2.4/ 1872

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
			reads zero)		
400F_F010	Port Data Input Register (GPIOA_PDIR)	32	R	0000_0000h	55.2.5/ 1873
400F_F014	Port Data Direction Register (GPIOA_PDDR)	32	R/W	0000_0000h	55.2.6/ 1873
400F_F040	Port Data Output Register (GPIOB_PDOR)	32	R/W	0000_0000h	55.2.1/ 1870
400F_F044	Port Set Output Register (GPIOB_PSOR)	32	W (always reads zero)	0000_0000h	55.2.2/ 1871
400F_F048	Port Clear Output Register (GPIOB_PCOR)	32	W (always reads zero)	0000_0000h	55.2.3/ 1871
400F_F04C	Port Toggle Output Register (GPIOB_PTOR)	32	W (always reads zero)	0000_0000h	55.2.4/ 1872
400F_F050	Port Data Input Register (GPIOB_PDIR)	32	R	0000_0000h	55.2.5/ 1873
400F_F054	Port Data Direction Register (GPIOB_PDDR)	32	R/W	0000_0000h	55.2.6/ 1873
400F_F080	Port Data Output Register (GPIOC_PDOR)	32	R/W	0000_0000h	55.2.1/ 1870
400F_F084	Port Set Output Register (GPIOC_PSOR)	32	W (always reads zero)	0000_0000h	55.2.2/ 1871
400F_F088	Port Clear Output Register (GPIOC_PCOR)	32	W (always reads zero)	0000_0000h	55.2.3/ 1871
400F_F08C	Port Toggle Output Register (GPIOC_PTOR)	32	W (always reads zero)	0000_0000h	55.2.4/ 1872
400F_F090	Port Data Input Register (GPIOC_PDIR)	32	R	0000_0000h	55.2.5/ 1873
400F_F094	Port Data Direction Register (GPIOC_PDDR)	32	R/W	0000_0000h	55.2.6/ 1873
400F_F0C0	Port Data Output Register (GPIOD_PDOR)	32	R/W	0000_0000h	55.2.1/ 1870

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F0C4	Port Set Output Register (GPIOD_PSOR)	32	W (always reads zero)	0000_0000h	55.2.2/ 1871
400F_F0C8	Port Clear Output Register (GPIOD_PCOR)	32	W (always reads zero)	0000_0000h	55.2.3/ 1871
400F_F0CC	Port Toggle Output Register (GPIOD_PTOR)	32	W (always reads zero)	0000_0000h	55.2.4/ 1872
400F_F0D0	Port Data Input Register (GPIOD_PDIR)	32	R	0000_0000h	55.2.5/ 1873
400F_F0D4	Port Data Direction Register (GPIOD_PDDR)	32	R/W	0000_0000h	55.2.6/ 1873
400F_F100	Port Data Output Register (GPIOE_PDOR)	32	R/W	0000_0000h	55.2.1/ 1870
400F_F104	Port Set Output Register (GPIOE_PSOR)	32	W (always reads zero)	0000_0000h	55.2.2/ 1871
400F_F108	Port Clear Output Register (GPIOE_PCOR)	32	W (always reads zero)	0000_0000h	55.2.3/ 1871
400F_F10C	Port Toggle Output Register (GPIOE_PTOR)	32	W (always reads zero)	0000_0000h	55.2.4/ 1872
400F_F110	Port Data Input Register (GPIOE_PDIR)	32	R	0000_0000h	55.2.5/ 1873
400F_F114	Port Data Direction Register (GPIOE_PDDR)	32	R/W	0000_0000h	55.2.6/ 1873
400F_F140	Port Data Output Register (GPIOF_PDOR)	32	R/W	0000_0000h	55.2.1/ 1870
400F_F144	Port Set Output Register (GPIOF_PSOR)	32	W (always reads zero)	0000_0000h	55.2.2/ 1871
400F_F148	Port Clear Output Register (GPIOF_PCOR)	32	W (always reads zero)	0000_0000h	55.2.3/ 1871

Table continues on the next page...

GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
400F_F14C	Port Toggle Output Register (GPIOF_PTOR)	32	W (always reads zero)	0000_0000h	55.2.4/ 1872
400F_F150	Port Data Input Register (GPIOF_PDIR)	32	R	0000_0000h	55.2.5/ 1873
400F_F154	Port Data Direction Register (GPIOF_PDDR)	32	R/W	0000_0000h	55.2.6/ 1873

55.2.1 Port Data Output Register (GPIOx_PDOR)

Addresses: GPIOA_PDOR is 400F_F000h base + 0h offset = 400F_F000h
GPIOB_PDOR is 400F_F040h base + 0h offset = 400F_F040h
GPIOC_PDOR is 400F_F080h base + 0h offset = 400F_F080h
GPIOD_PDOR is 400F_F0C0h base + 0h offset = 400F_F0C0h
GPIOE_PDOR is 400F_F100h base + 0h offset = 400F_F100h
GPIOF_PDOR is 400F_F140h base + 0h offset = 400F_F140h

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	PDO																															
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

GPIOx_PDOR field descriptions

Field	Description
31–0 PDO	Port Data Output Unimplemented pins for a particular device read as zero. 0 Logic level 0 is driven on pin provided pin is configured for General Purpose Output. 1 Logic level 1 is driven on pin provided pin is configured for General Purpose Output.

55.2.2 Port Set Output Register (GPIOx_PSOR)

Addresses: GPIOA_PSOR is 400F_F000h base + 4h offset = 400F_F004h
GPIOB_PSOR is 400F_F040h base + 4h offset = 400F_F044h
GPIOC_PSOR is 400F_F080h base + 4h offset = 400F_F084h
GPIOD_PSOR is 400F_F0C0h base + 4h offset = 400F_F0C4h
GPIOE_PSOR is 400F_F100h base + 4h offset = 400F_F104h
GPIOF_PSOR is 400F_F140h base + 4h offset = 400F_F144h

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																															
W	PTSO																															
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

GPIOx_PSOR field descriptions

Field	Description
31–0 PTSO	Port Set Output Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to logic one.

55.2.3 Port Clear Output Register (GPIOx_PCOR)

Addresses: GPIOA_PCOR is 400F_F000h base + 8h offset = 400F_F008h
GPIOB_PCOR is 400F_F040h base + 8h offset = 400F_F048h
GPIOC_PCOR is 400F_F080h base + 8h offset = 400F_F088h
GPIOD_PCOR is 400F_F0C0h base + 8h offset = 400F_F0C8h
GPIOE_PCOR is 400F_F100h base + 8h offset = 400F_F108h
GPIOF_PCOR is 400F_F140h base + 8h offset = 400F_F148h

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																															
W	PTCO																															
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

GPIOx_PCOR field descriptions

Field	Description
31–0 PTCO	Port Clear Output

GPIOx_PCOR field descriptions (continued)

Field	Description
	Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic zero.

55.2.4 Port Toggle Output Register (GPIOx_PTOR)

Addresses: GPIOA_PTOR is 400F_F000h base + Ch offset = 400F_F00Ch
GPIOB_PTOR is 400F_F040h base + Ch offset = 400F_F04Ch
GPIOC_PTOR is 400F_F080h base + Ch offset = 400F_F08Ch
GPIOD_PTOR is 400F_F0C0h base + Ch offset = 400F_F0CCh
GPIOE_PTOR is 400F_F100h base + Ch offset = 400F_F10Ch
GPIOF_PTOR is 400F_F140h base + Ch offset = 400F_F14Ch

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																															
W	PTTO																															
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

GPIOx_PTOR field descriptions

Field	Description
31–0 PTTO	Port Toggle Output Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows: 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state.

55.2.5 Port Data Input Register (GPIOx_PDIR)

Addresses: GPIOA_PDIR is 400F_F000h base + 10h offset = 400F_F010h
GPIOB_PDIR is 400F_F040h base + 10h offset = 400F_F050h
GPIOC_PDIR is 400F_F080h base + 10h offset = 400F_F090h
GPIOD_PDIR is 400F_F0C0h base + 10h offset = 400F_F0D0h
GPIOE_PDIR is 400F_F100h base + 10h offset = 400F_F110h
GPIOF_PDIR is 400F_F140h base + 10h offset = 400F_F150h

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	PDI																															
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

GPIOx_PDIR field descriptions

Field	Description
31–0 PDI	Port Data Input Unimplemented pins for a particular device read as zero. Pins that are not configured for a digital function read as zero. If the corresponding Port Control and Interrupt module is disabled, then that Port Data Input Register does not update. 0 Pin logic level is logic zero or is not configured for use by digital function. 1 Pin logic level is logic one.

55.2.6 Port Data Direction Register (GPIOx_PDDR)

The PDDR configures the individual port pins for input or output.

Addresses: GPIOA_PDDR is 400F_F000h base + 14h offset = 400F_F014h
GPIOB_PDDR is 400F_F040h base + 14h offset = 400F_F054h
GPIOC_PDDR is 400F_F080h base + 14h offset = 400F_F094h
GPIOD_PDDR is 400F_F0C0h base + 14h offset = 400F_F0D4h
GPIOE_PDDR is 400F_F100h base + 14h offset = 400F_F114h
GPIOF_PDDR is 400F_F140h base + 14h offset = 400F_F154h

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	PDD																															
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

GPIOx_PDDR field descriptions

Field	Description
31–0 PDD	Port data direction 0 Pin is configured as general purpose input, if configured for the GPIO function 1 Pin is configured for general purpose output, if configured for the GPIO function

55.3 Functional description

55.3.1 General purpose input

The logic state of each pin is available via the pin data input registers, provided the pin is configured for a digital function and the corresponding port control and interrupt module is enabled.

The pin data input registers return the synchronized pin state after any enabled digital filter in the port control and interrupt module. 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.

55.3.2 General purpose output

The logic state of each pin can be controlled via the pin data output registers and port data direction registers, provided the pin is configured for the GPIO function.

If a pin is configured for the GPIO function and the corresponding port data direction register bit is clear then the pin is configured as an input.

If a pin is configured for the GPIO function and the corresponding port data direction register bit is set then the pin is configured as an output and the logic state of the pin is equal to the corresponding pin data output register.

To facilitate efficient bit banging on the general purpose outputs, pin data set, pin data clear and pin data toggle registers exist to allow one or more outputs within the 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 pin output enable registers and pin data output registers (including the set/clear/toggle registers).

Chapter 56
Touch sense input (TSI)

56.1 Introduction

NOTE

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

The touch sensing input (TSI) module provides capacitive touch sensing detection with high sensitivity and enhanced robustness. Each TSI pin implements the capacitive measurement of an electrode having individual programmable detection thresholds and result registers. The TSI module can be functional in several low power modes with ultra low current adder and waking up the CPU in a touch event. It provides a solid capacitive measurement module to the implementation of touch keypad, rotaries and sliders.

56.2 Features

TSI module features included:

- Support as many as 16 input capacitive touch sensing pins with individual result registers
- Automatic detection of electrode capacitance change in low power mode with programmable upper and lower threshold
- Automatic periodic scan unit with different duty cycles for run and low power modes
- Fully support with FSL touch sensing SW library suite the implementation of keypads, rotaries and sliders
- Operation across all low power modes: WAIT,STOP, VLPR, VLPW, VLPS, LLS,VLLS{3,2,1}
- Capability to wake up MCU from low power modes.
- Configurable interrupts:
  - End-of-scan or out-of-range interrupt
  - TSI error interrupts: pad short to V_{DD}/V_{SS} or conversion overrun

- Compensate temperature and supply voltage variations
- Stand alone operation not requiring any external crystal even in low power modes
- Configurable integration of each electrode capacitance measurement from 1 to 4096 periods
- Programmable Electrode Oscillator and TSI Reference Oscillator allowing high sensitivity, small scan time and low power functionality.
- Only uses one pin per electrode implementation with no external hardware required

56.3 Overview

This section presents an overview of the TSI module. The following figure presents the simplified TSI module block diagram.

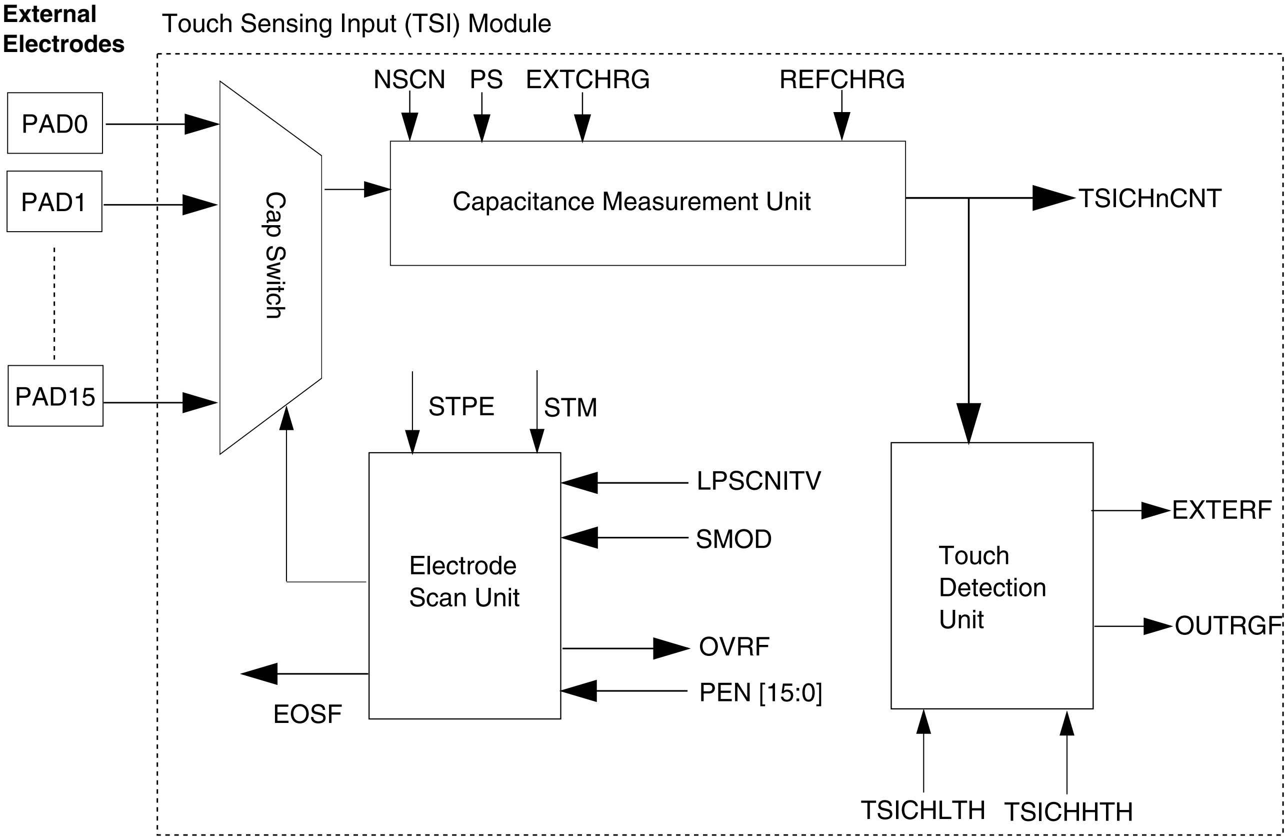


Figure 56-1. Touch sensing input block diagram

56.3.1 Electrode capacitance measurement unit

The electrode capacitance measurement unit senses the capacitance of a TSI pin and outputs a 16-bit result. This module is based in dual oscillator architecture. One oscillator is connected to the external electrode array and oscillates according to the electrode capacitance, while the other according to an internal reference capacitor. The pin capacitance measurement is given by the counted number of periods of the reference oscillator during a pre-defined number of electrode oscillations.

The electrode oscillator charges and discharges the pin capacitance with a programmable current source in order to accommodate several different sizes of electrode capacitances. The electrode oscillator frequency, before being compared to that of the reference oscillator, goes through a prescaler and module counter to decrease its frequency and consecutively increase the measurement resolution and noise robustness.

The following figure presents the simplified block diagram of how the electrode capacitance is measured.

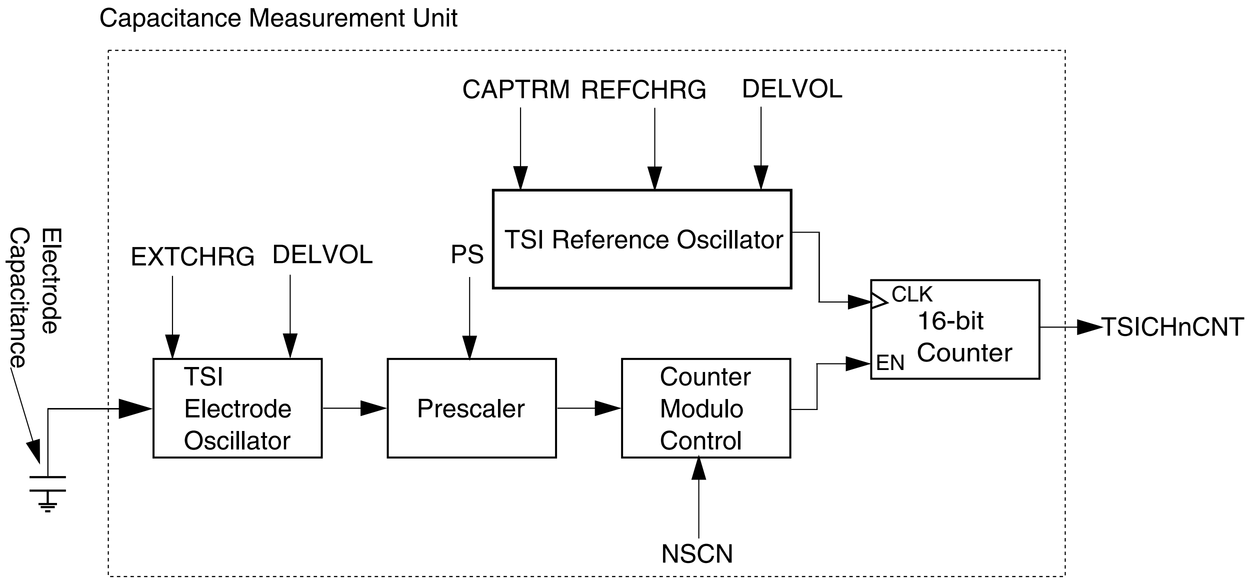


Figure 56-2. TSI capacitance measurement unit block diagram

56.3.2 Electrode scan unit

This session describes the functionality of the electrode scan unit. It is responsible for triggering the start of the active electrode scan.

The touch sense input module needs to periodically scan all active electrodes to determine if a touch event has occurred. The electrode scan unit is responsible for defining two independent scan periods, one for TSI active mode and the other for TSI low power mode. This independent control allows the application to configure longer scan period during low power mode, so contributing to smaller average power consumption. The TSI, in low power mode, has the capability to wake up the CPU upon an electrode capacitance change. When the CPU wakes, the TSI enters active mode, and a shorter scan period can provide a faster response time and more robust touch detection. Apart from the periodical mode, the electrode scan unit also allows software triggering of the electrode scans. This feature is very useful for initialization of the touch application for detecting the initial electrode capacitances. This module generates configurable end-of-scan interrupt to indicate the application that all electrodes were scanned. In the event starting a new electrode scan while a previous one is still in progress an overrun error flag is generated.

56.3.3 Touch detection unit

The touch detection unit indicates any change in the low power electrode pin capacitance. The purpose of this module is to only wake up th CPU from low power modes in the event of a electrode capacitance change. So, if there is no capacitance change in the electrode, the MCU stays in low power mode indefinitely, while keeping the electrode monitoring, ensuring minimal power consumption.

This module compares the pin capacitance value in the result register with a pre-configured low and high threshold. If the capacitance result register value is outside the ranges defined by upper and lower threshold the touch detection unit generates an out-of-range flag indicating a pin capacitance change.

The upper and lower threshold values are configurable allowing the application to select the magnitude of the capacitance change to trigger the out-of-range flag. With the threshold values programmed properly, the application noise level does not cause frequent CPU interrupts, so minimizes the CPU usage.

56.4 Modes of operation

The TSI module has three operation modes: disabled, active mode and low power mode.

Table 56-1. TSI Module funtionality in MCU operation modes

MCU operation mode	TSI clock sources	TSI operation mode when TSIEN = 1	Functional electrode pins	Required STPE state
Run	LPOCLK, MSGIRCLK, OSC0ERCLK	Active mode	All	Don't care
Wait	LPOCLK, MSGIRCLK, OSC0ERCLK	Active mode	All	Don't care

Table continues on the next page...

Table 56-1. TSI Module funtionality in MCU operation modes (continued)

MCU operation mode	TSI clock sources	TSI operation mode when TSIEN = 1	Functional electrode pins	Required STPE state
Stop	LPOCLK, MSGIRCLK, OSC0ERCLK	Active mode	All	1
VLPRun	LPOCLK, MSGIRCLK, OSC0ERCLK	Active mode	All	Don't care
VLPWait	LPOCLK, MSGIRCLK, OSC0ERCLK	Active mode	All	Don't care
VLPStop	LPOCLK MSGIRCLK, OSC0ERCLK	Active mode	All	1
LLS	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1
VLLS3	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1
VLLS2	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1
VLLS1	LPOCLK, VLPOSCCLK	Low power mode	Determined by PEN[LPSP]	1

56.4.1 TSI disabled mode

When GENCS[TSIEN] is cleared, the TSI module is disabled, and does not perform any functionally in any MCU operation mode.

56.4.2 TSI active mode

In active mode, the TSI module has its full functionality, being able to scan up to 16 electrodes. The TSI can be in active mode with the MCU in any of the following operational modes: run, wait, stop, VLPR, VLPW and VLPS.

Three clocks sources can be selected for the TSI module in active mode: LPOCLK, MCGIRCLK and OSC0ERCLK.

56.4.3 TSI low power mode

The TSI modules enters in low power mode if the GENCS[STPE] is set to one and the MCU enters in one of the following operational modes: LLS, VLLS1, VLLS2 and VLLS3. In low power mode, only one selectable pin is active, being able to perform capacitance measurements. The scan period is defined by GENCS[LPSCNITV] . Two low power clock sources are available in the TSI low power mode, LPOCLK and VLPOSCCLK, being selected by the GENCS[LPCLKS].

In low power mode the TSI interrupt can also be configured as end-of-scan or out-of-range and the GENCS[TSIIEN] must be set in order to generate these interrupts. The TSI interrupt causes the exit of the low power mode and entrance in the active mode, and the MCU also wakes up.

In low power mode the electrode scan unit is always configured to periodical low power scan.

56.4.4 Block diagram

The following figure shows the block diagram of TSI module.[1]

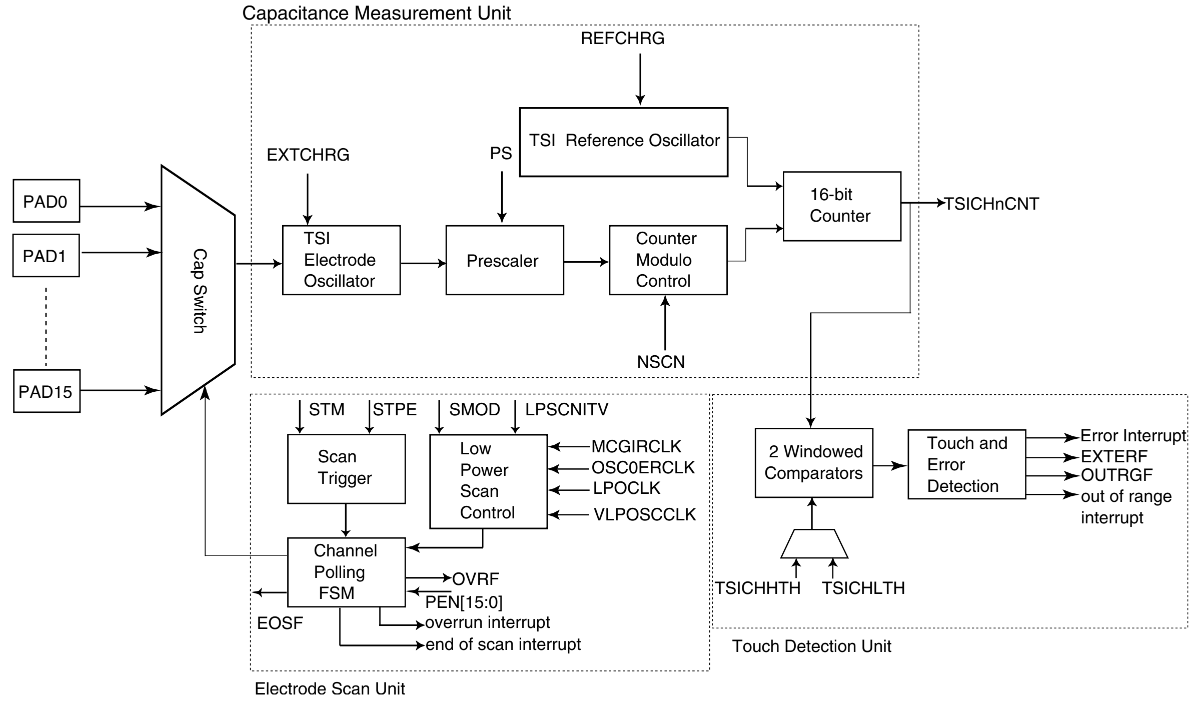


Figure 56-3. TSI block diagram

[1] The out of range functinality present in the Touch Detection Unit is only available in low power modes.

56.5 TSI signal descriptions

The TSI module has up to 16 external pins for touch sensing. The table below itemizes all the TSI external pins.

Table 56-2. TSI signal descriptions

Signal	Description	I/O
TSI_IN[15:0]	TSI capacitive pins. Switchable driver that connects directly to the electrode pins TSI[15:0] can operate as GPIO pins	I/O

56.5.1 TSI_IN[15:0]

When TSI functionality is enabled by the PEN[PENn], the TSI analog portion uses corresponding TSI_IN[n] pin to connect the module with the external electrode. The connection between the pin and the touch pad must be kept as short as possible to reduce distribution capacity on board.

56.6 Memory map and register definition

This section presents the touch sensing input module memory map and registers definition.

TSI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_5000	General Control and Status Register (TSI0_GENCS)	32	R/W	0000_0000h	56.6.1/ 1882
4004_5004	SCAN Control Register (TSI0_SCANC)	32	R/W	0000_0000h	56.6.2/ 1886
4004_5008	Pin Enable Register (TSI0_PEN)	32	R/W	0000_0000h	56.6.3/ 1887
4004_500C	Wake-Up Channel Counter Register (TSI0_WUCNTR)	32	R/W	0000_0000h	56.6.4/ 1890
4004_5100	Counter Register (TSI0_CNTR1)	32	R	0000_0000h	56.6.5/ 1890

Table continues on the next page...

TSI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
4004_5104	Counter Register (TSI0_CNTR3)	32	R	0000_0000h	56.6.5/ 1890
4004_5108	Counter Register (TSI0_CNTR5)	32	R	0000_0000h	56.6.5/ 1890
4004_510C	Counter Register (TSI0_CNTR7)	32	R	0000_0000h	56.6.5/ 1890
4004_5110	Counter Register (TSI0_CNTR9)	32	R	0000_0000h	56.6.5/ 1890
4004_5114	Counter Register (TSI0_CNTR11)	32	R	0000_0000h	56.6.5/ 1890
4004_5118	Counter Register (TSI0_CNTR13)	32	R	0000_0000h	56.6.5/ 1890
4004_511C	Counter Register (TSI0_CNTR15)	32	R	0000_0000h	56.6.5/ 1890
4004_5120	Low Power Channel Threshold Register (TSI0_THRESHOLD)	32	R/W	0000_0000h	56.6.6/ 1891

56.6.1 General Control and Status Register (TSIx_GENCS)

Addresses: TSI0_GENCS is 4004_5000h base + 0h offset = 4004_5000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			LPCLKS	LPSCNITV				NSCN					PS		
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	EOSF	OUTRGF	EXTERF	OVRF	0		SCNIP		TSIEN	TSIIE	ERIE	ESOR	0	Reserved	STM	STPE
W	w1c	w1c	w1c	w1c				SWTS								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TSIx_GENCS field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value zero.
28 LPCLKS	Low Power Mode Clock Source Selection. This bit-field can only be changed if the TSI module is disabled (TSIEN bit = 0). 0 LPOCLK is selected to determine the scan period in low power mode 1 VLPOSCCLK is selected to determine the scan period in low power mode
27–24 LPSCNITV	TSI Low Power Mode Scan Interval. This bit-field can only be changed if the TSI module is disabled (TSIEN bit = 0). 0000 1 ms scan interval 0001 5 ms scan interval 0010 10 ms scan interval 0011 15 ms scan interval 0100 20 ms scan interval 0101 30 ms scan interval 0110 40 ms scan interval 0111 50 ms scan interval 1000 75 ms scan interval 1001 100 ms scan interval 1010 125 ms scan interval 1011 150 ms scan interval 1100 200 ms scan interval 1101 300 ms scan interval 1110 400 ms scan interval 1111 500 ms scan interval
23–19 NSCN	Number of Consecutive Scans per Electrode electrode. This bit-field can only be changed if the TSI module is disabled (TSIEN bit = 0). 00000 Once per electrode 00001 Twice per electrode 00010 3 times per electrode 00011 4 times per electrode 00100 5 times per electrode 00101 6 times per electrode 00110 7 times per electrode 00111 8 times per electrode 01000 9 times per electrode 01001 10 times per electrode 01010 11 times per electrode 01011 12 times per electrode 01100 13 times per electrode 01101 14 times per electrode 01110 15 times per electrode 01111 16 times per electrode

Table continues on the next page...

TSIx_GENCS field descriptions (continued)

Field	Description
	10000 17 times per electrode 10001 18 times per electrode 10010 19 times per electrode 10011 20 times per electrode 10100 21 times per electrode 10101 22 times per electrode 10110 23 times per electrode 10111 24 times per electrode 11000 25 times per electrode 11001 26 times per electrode 11010 27 times per electrode 11011 28 times per electrode 11100 29 times per electrode 11101 30 times per electrode 11110 31 times per electrode 11111 32 times per electrode
18–16 PS	Electrode Oscillator prescaler. . This bit-field can only be changed if the TSI module is disabled (TSIEN bit = 0) 000 Electrode Oscillator Frequency divided by 1 001 Electrode Oscillator Frequency divided by 2 010 Electrode Oscillator Frequency divided by 4 011 Electrode Oscillator Frequency divided by 8 100 Electrode Oscillator Frequency divided by 16 101 Electrode Oscillator Frequency divided by 32 110 Electrode Oscillator Frequency divided by 64 111 Electrode Oscillator Frequency divided by 128
15 EOSF	End of Scan Flag. This flag is set when all active electrodes are scanned is ended after a scan trigger. Writing "1" to this bit will clear the flag to 0.
14 OUTRGF	Out of Range Flag. This flag is set if the result register of the low power enabled electrode is outside the range defined by the TSI_THRESHOLD register. This flag is only set when the TSI is in low power mode. It can be read once the CPU wakes up. Writing "1" to this bit will clear the flag to 0.
13 EXTERF	External Electrode error occurred This flag is set when an active electrode has a result register either 0x0000 or 0xFFFF. Writing "1" to this bit will clear the flag to 0. 0 No fault happend on TSI electrodes 1 Short to VDD or VSS was detected on one or more electrodes.
12 OVRF	Overrun error Flag. This flag is set when a scan trigger occurs while a scan is still in progress. Writing "1" to this bit will clear the flag to 0.

Table continues on the next page...

TSIx_GENCS field descriptions (continued)

Field	Description
	0 No over run. 1 Over Run occurred.
11–10 Reserved	This read-only field is reserved and always has the value zero.
9 SCNIP	Scan In Progress status "1" indicates a scanning process is in progress,this bit is read-only and changes automatically by the TSI model.
8 SWTS	Software Trigger Start Write a "1" to this bit will start a scan sequence and write a "0" to this bit has no effect.
7 TSIEN	Touch Sensing Input Module Enable 0 TSI module is disabled 1 TSI module is enabled
6 TSIIE	Touch Sensing Input Interrupt Module Enable 0 Interrupt from TSI is disabled 1 Interrupt from TSI is enabled
5 ERIE	Error Interrupt Enable Caused either by a Short or Overrun Error. 0 Interrupt disabled for error. 1 Interrupt enabled for error.
4 ESOR	End-of-Scan or Out-of-Range Interrupt select 0 Out-of-Range interrupt is allowed. 1 End-of-Scan interrupt is allowed.
3 Reserved	This read-only field is reserved and always has the value zero.
2 Reserved	Reserved This field is reserved.
1 STM	Scan Trigger Mode. This bit-field can only be changed if the TSI module is disabled (TSIEN bit = 0). 0 Software trigger scan. 1 Periodical Scan.
0 STPE	TSI STOP Enable while in Low Power Modes (STOP, VLPS, LLS and VLLS{3,2,1}) 0 Disable TSI when MCU goes into low power modes. 1 Allows TSI to continue running in all low power modes.

56.6.2 SCAN Control Register (TSIx_SCANC)

Addresses: TSI0_SCANC is 4004_5000h base + 4h offset = 4004_5004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				REFCHRG				0				EXTCHRG			
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	SMOD								0		0	AMCLKS		AMPSC		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TSIx_SCANC field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value zero.
27–24 REFCHRG	Ref OSC Charge Current select 0000 2 μA charge current. 0001 4 μA charge current. 0010 6 μA charge current. 0011 8 μA charge current. 0100 10 μA charge current. 0101 12 μA charge current. 0110 14 μA charge current. 0111 16 μA charge current. 1000 18 μA charge current. 1001 20 μA charge current. 1010 22 μA charge current. 1011 24 μA charge current. 1100 26 μA charge current. 1101 28 μA charge current. 1110 30 μA charge current. 1111 32 μA charge current.
23–20 Reserved	This read-only field is reserved and always has the value zero.
19–16 EXTCHRG	External OSC Charge Current select 0000 2 μA charge current. 0001 4 μA charge current. 0010 6 μA charge current. 0011 8 μA charge current. 0100 10 μA charge current. 0101 12 μA charge current.

Table continues on the next page...

TSIx_SCANC field descriptions (continued)

Field	Description
	0110 14 µA charge current. 0111 16 µA charge current. 1000 18 µA charge current. 1001 20 µA charge current. 1010 22 µA charge current. 1011 24 µA charge current. 1100 26 µA charge current. 1101 28 µA charge current. 1110 30 µA charge current. 1111 32 µA charge current.
15–8 SMOD	Scan Module 00000000 Continue Scan. Others Scan Period Modulus.
7–6 Reserved	This read-only field is reserved and always has the value zero.
5 Reserved	This read-only field is reserved and always has the value zero.
4–3 AMCLKS	Active Mode Clock Source 00 LPOSCCLK 01 MCGIRCLK. 10 OSC0ERCLK. 11 Not valid.
2–0 AMPSC	Active Mode Prescaler 000 Input Clock Source divided by 1. 001 Input Clock Source divided by 2. 010 Input Clock Source divided by 4. 011 Input Clock Source divided by 8. 100 Input Clock Source divided by 16. 101 Input Clock Source divided by 32. 110 Input Clock Source divided by 64. 111 Input Clock Source divided by 128.

56.6.3 Pin Enable Register (TSIx_PEN)

Do not change the settings when TSIEN is 1.

Addresses: TSI0_PEN is 4004_5000h base + 8h offset = 4004_5008h

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												LPSP				PEN15	PEN14	PEN13	PEN12	PEN11	PEN10	PEN9	PEN8	PEN7	PEN6	PEN5	PEN4	PEN3	PEN2	PEN1	PEN0
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

TSIx_PEN field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value zero.
19–16 LPSP	Low Power Scan Pin 0000 TSI_IN[0] is active in low power mode. 0001 TSI_IN[1] is active in low power mode. 0010 TSI_IN[2] is active in low power mode. 0011 TSI_IN[3] is active in low power mode. 0100 TSI_IN[4] is active in low power mode. 0101 TSI_IN[5] is active in low power mode. 0110 TSI_IN[6] is active in low power mode. 0111 TSI_IN[7] is active in low power mode. 1000 TSI_IN[8] is active in low power mode. 1001 TSI_IN[9] is active in low power mode. 1010 TSI_IN[10] is active in low power mode. 1011 TSI_IN[11] is active in low power mode. 1100 TSI_IN[12] is active in low power mode. 1101 TSI_IN[13] is active in low power mode. 1110 TSI_IN[14] is active in low power mode. 1111 TSI_IN[15] is active in low power mode.
15 PEN15	Touch Sensing Input Pin Enable Register 15 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
14 PEN14	Touch Sensing Input Pin Enable Register 14 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
13 PEN13	Touch Sensing Input Pin Enable Register 13 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
12 PEN12	Touch Sensing Input Pin Enable Register 12 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
11 PEN11	Touch Sensing Input Pin Enable Register 11 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
10 PEN10	Touch Sensing Input Pin Enable Register 10 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
9 PEN9	Touch Sensing Input Pin Enable Register 9

Table continues on the next page...

TSIx_PEN field descriptions (continued)

Field	Description
	0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
8 PEN8	Touch Sensing Input Pin Enable Register 8 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
7 PEN7	Touch Sensing Input Pin Enable Register 7 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
6 PEN6	Touch Sensing Input Pin Enable Register 6 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
5 PEN5	Touch Sensing Input Pin Enable Register 5 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
4 PEN4	Touch Sensing Input Pin Enable Register 4 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
3 PEN3	Touch Sensing Input Pin Enable Register 3 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
2 PEN2	Touch Sensing Input Pin Enable Register 2 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
1 PEN1	Touch Sensing Input Pin Enable Register 1 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.
0 PEN0	Touch Sensing Input Pin Enable Register 0 0 The corresponding pin is not used by TSI. 1 The corresponding pin is used by TSI.

56.6.4 Wake-Up Channel Counter Register (TSIx_WUCNTR)

Addresses: TSI0_WUCNTR is 4004_5000h base + Ch offset = 4004_500Ch

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																WUCNT															
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

TSIx_WUCNTR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 WUCNT	TouchSensing wake-up Channel 16bit counter value

56.6.5 Counter Register (TSIx_CNTR)

Addresses: TSI0_CNTR1 is 4004_5000h base + 100h offset = 4004_5100h
TSI0_CNTR3 is 4004_5000h base + 104h offset = 4004_5104h
TSI0_CNTR5 is 4004_5000h base + 108h offset = 4004_5108h
TSI0_CNTR7 is 4004_5000h base + 10Ch offset = 4004_510Ch
TSI0_CNTR9 is 4004_5000h base + 110h offset = 4004_5110h
TSI0_CNTR11 is 4004_5000h base + 114h offset = 4004_5114h
TSI0_CNTR13 is 4004_5000h base + 118h offset = 4004_5118h
TSI0_CNTR15 is 4004_5000h base + 11Ch offset = 4004_511Ch

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	CTN																CTN1															
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

TSIx_CNTR*n* field descriptions

Field	Description
31–16 CTN	TouchSensing Channel n 16-bit counter value
15–0 CTN1	TouchSensing Channel n-1 16-bit counter value

56.6.6 Low Power Channel Threshold Register (TSIx_THRESHOLD)

Addresses: TSI0_THRESHOLD is 4004_5000h base + 120h offset = 4004_5120h

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	LTHH																HTHH															
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

TSIx_THRESHOLD field descriptions

Field	Description
31–16 LTHH	Touch Sensing Channel Low Threshold value
15–0 HTHH	Touch Sensing Channel High Threshold value

56.7 Functional descriptions

This section provides functional description of the TSI module.

56.7.1 Capacitance measurement

The electrode pin capacitance measurement uses a dual oscillator approach. The TSI electrode oscillator has its frequency dependable of the external electrode capacitance and of the TSI module configuration. After going to a configurable prescaler, the TSI electrode oscillator signal goes to the input of the module counter. The time for the module counter to reach its module value is measured using the TSI reference oscillator. The measured electrode capacitance is directly proportional to the time.

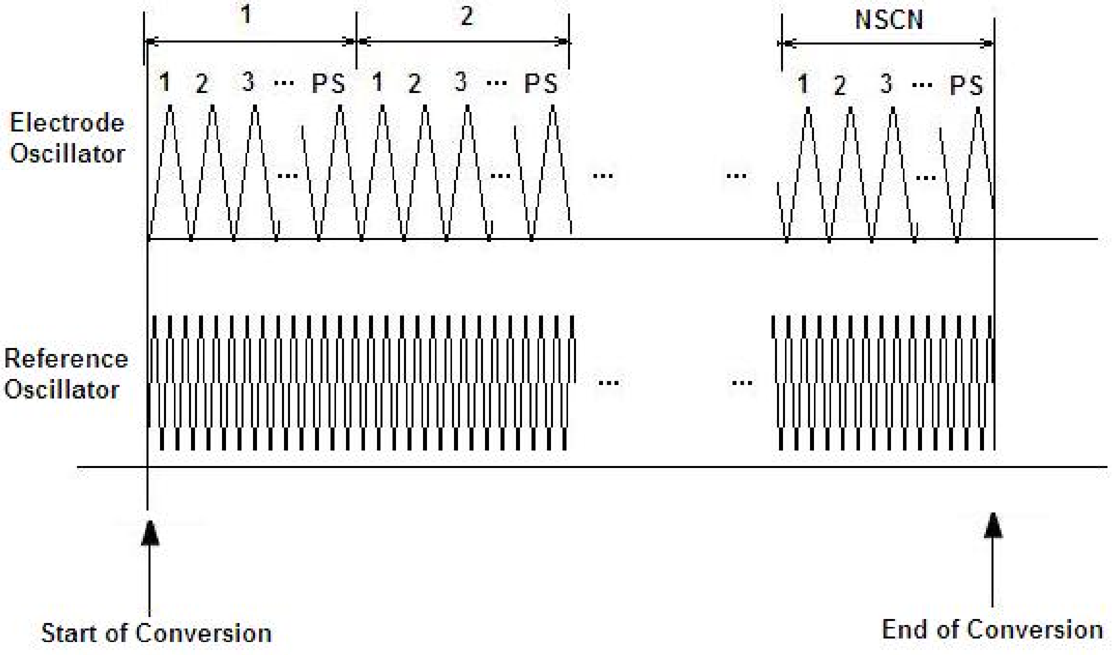


Figure 56-32. Dual Electrode Capacitance Measurement

56.7.1.1 TSI electrode oscillator

The TSI electrode oscillator circuit is illustrated in the following figure. A configurable constant current source is used to charge and discharge the external electrode capacitance. A buffer hysteresis defines the oscillator delta voltage. The delta voltage defines the margin of high and low voltage which are the reference input of the comparator in different time.

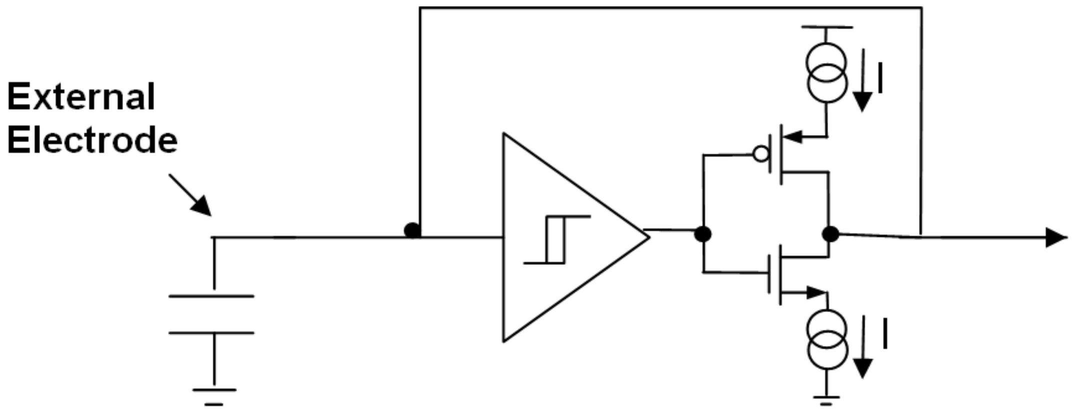


Figure 56-33. TSI electrode oscillator circuit

The current source applied to the pad capacitance is controlled by the SCANC[EXTCHRG]. The hysteresis delta voltage is defined inn the module electrical specifications present in the device DataSheet.. The figure below shows the voltage amplitude waveform of the electrode capacitance charging and discharging with a programmable current.

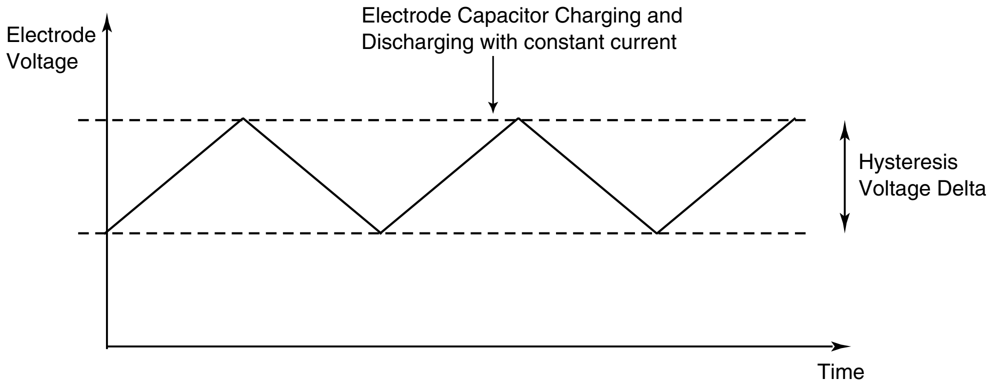


Figure 56-34. TSI electrode oscillator chart

The oscillator frequency is give by the following equation

$$F_{elec} = \frac{I}{2 * C_{elec} * \Delta V}$$

Figure 56-35. Equation 1: TSI electrode oscillator frequency

Where:

I: constant current

C_{elec}: electrode capacitance

ΔV: Hysteresis delta voltage

So by this equation, for example, an electrode with C_{elec}= 20 pF, with a current source of I = 16 µA and ΔV = 600 mV have the following oscillation frequency:

$$F_{elec} = \frac{16\ \mu A}{2 * 20pF * 600mV} = 0.67MHz$$

Figure 56-36. Equation 2: TSI electrode oscillator frequency

The current source is used to accommodate the TSI electrode oscillator frequency with different electrode capacitance sizes.

56.7.1.2 Electrode oscillator and counter module control

The TSI oscillator frequency signal goes through a prescaler defined by the GENCS[PS] and then enters in a module counter. The bit field GENCS[NSCN] defines the maximum count value of the module counter.

The pin capacitance sampling time is given by the time the module counter takes to go from zero to its maximum value, defined by NSCN. The electrode sample time is expressed by the following equation:

$$T_{cap\_samp} = \frac{PS * NSCN}{F_{elec}}$$

Using Equation 1.

$$T_{cap\_samp} = \frac{2 * PS * NSCN * C_{elec} * \Delta V}{I}$$

Figure 56-37. Equation 3: Electrode sampling time

Where:

PS: prescaler value

NSCN: module counter maximum value

I: constant current

C_{elec}: electrode capacitance

ΔV: Hysteresis delta voltage

By this equation we have that an electrode with C = 20 pF, with a current source of I = 16 µA and ΔV = 600 mV, PS = 2 and NSCN = 16 have the following sampling time:

$$T_{cap\_samp} = \frac{2*2*16*20pF*600mV}{16\mu A} = 48\mu s$$

56.7.1.3 TSI reference oscillator

The TSI reference oscillator has the same topology of the TSI electrode oscillator. The TSI reference oscillator instead of using an external capacitor for the electrode oscillator has an internal reference capacitor.

The TSI reference oscillator has an independent programmable current source controlled by the SCANC[REFCHRG].

The reference oscillator frequency is given by the following equation:

$$F_{ref\_osc} = \frac{I_{ref}}{2 * C_{ref} * \Delta V}$$

Figure 56-38. Equation 4: TSI reference oscillator frequency

Where:

C_{ref}: Internal reference capacitor

I_{ref}: Reference oscillator current source

ΔV : Hysteresis delta voltage

Considering $C_{ref} = 1.0$ pF, $I_{ref} = 12$ µA and $\Delta V = 600$ mV, follows

$$F_{ref\_osc} = \frac{12\mu A}{2 * 1.0pF * 600mV} = 10.0MHz$$

56.7.2 TSI measurement result

The capacitance measurement result is defined by the number of TSI reference oscillator periods during the sample time and is stored in the TSICHnCNT register.

$\mathrm{TSICHnCNT} = T_{cap\_samp} * F_{ref\_osc}$

Using Equation 2 and Equation 1 follows:

$$\mathrm{TSICHnCNT} = \frac{I_{ref} * PS * NSCN}{C_{ref} * I_{elec}} * C_{elec}$$

Figure 56-39. Equation 5: Capacitance result value

In the example where $F_{ref_osc} = 10.0$MHz and $T_{cap\_samp} = 48$ µs, TSICHnCNT = 480

56.7.3 Electrode scan unit

This session describes the functionality of the electrode scan unit. It is responsible for triggering the start of the active electrode scan.

The touch sense input module needs to periodically scan all active electrodes to determine if a touch event has occurred. The electrode scan unit is responsible for defining two independent scan periods, one for TSI active mode and the other for TSI low power mode. This independent control allows the application to configure longer scan period during low power mode, so contributing to smaller average power consumption. The TSI, in low power mode, has the capability to wake up the CPU upon an electrode capacitance change. When the CPU wakes, the TSI enters active mode, and a shorter scan period can provide a faster response time and more robust touch detection. Apart from the periodical mode, the electrode scan unit also allows software triggering of the electrode scans. This feature is very useful for initialization of the touch application for detecting the initial electrode capacitances. This module generates configurable end-of-scan interrupt to indicate the application that all electrodes were scanned. In the event starting a new electrode scan while a previous one is still in progress an overrun error flag is generated.

56.7.3.1 Active electrodes

The electrode scan unit is responsible to start the capacitance measurement of all active electrodes. Each electrode pin should be activated by writing a 1 to the respective PEN[PEN] bit.

Once an electrode scan is triggered, the electrode scan unit, controls the scanning of all the active electrodes sequentially. It starts the scanning of the electrode pin TSI_IN[0] and goes sequentially scanning until it reaches the electrode pin TSI_IN[15]. The electrode pins that does not have its enable bit (PEN[PEN]) are not scanned and are skipped.

Only one electrode pin is functional in the low power mode scan and it’s defined by the bit-field PEN[LPSP]. In low power scan mode the configuration of PEN[PEN] bits are ignored.

56.7.3.2 Scan trigger

The scan trigger can be set to periodical scan or software trigger. The bit GENCS[STM] determines the TSI scan trigger mode. If STM = 1 the trigger mode is selected as continuous. If STM = 0, the software trigger mode is selected. In periodic mode the scan trigger is generated automatically by the electrode scan unit

56.7.3.3 Software trigger mode

The software trigger scan is started by writing 1 to the bit GENCS[SWTS]. A single scan of all active electrodes is performed. The software trigger scan only can be initiated by the GENCS[SWTS] bit if the STM = 0. If STM = 1, any write in the GENCS[SWTS] bit is ignored.

56.7.3.4 Periodic scan control

The electrode scan unit operates both in TSI active mode and TSI low power mode. It has a separate scan period control for each one of these modes. It allows the application to controls the trade-off of the scan frequency and the average TSI module power consumption.

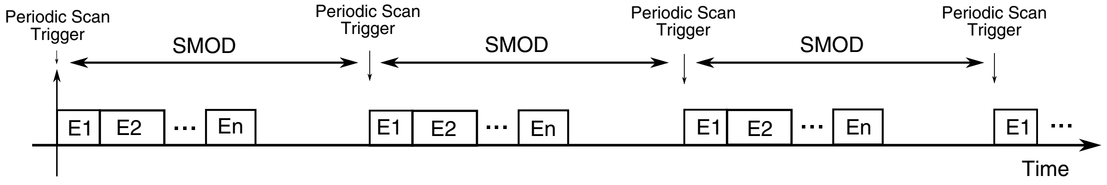


Figure 56-40. Periodical Scan Time Chart

56.7.3.4.1 Active mode periodic scan

In active mode periodic scan the scan following clocks can be selected: LPOOSCCLK, MCGIRCLK and OSC0ERCLK. The bit field SCANC[AMCLKS] selects the TSI clock source for the active mode scan. The scan period is determined by the SCANC[SMOD] value. SMOD is the module of the counter that determines the scan period.

The following figure presents the scan sequence performed by the TSI module. Every active electrode is scanned sequentially, stating with the TSI_IN[0] and ending with the TSI_IN[15] pin, if they are active.

When the electrode scan unit starts a scan sequence, all the active electrodes will be scanned sequentially with each electrode has the scanned time defined by the GENCS[NSCN]. The counter value is the sum of the total scan times of that electrode.

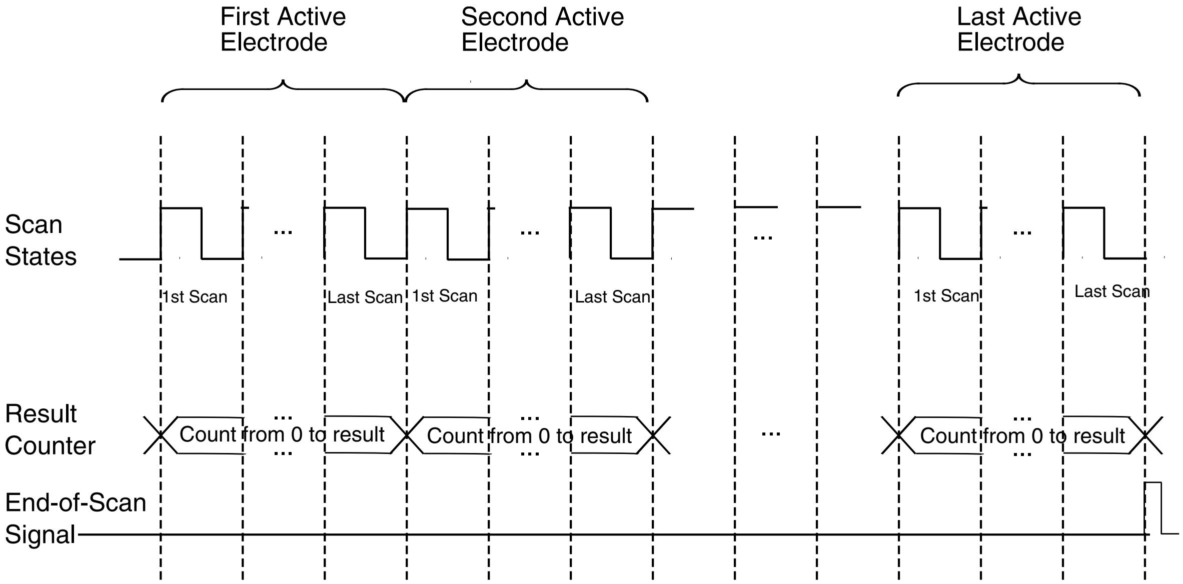


Figure 56-41. Scan sequence

56.7.3.4.2 Low power mode scan

In low power periodic scan the scan period is define by the GENCS[LPSCNITV]. The TSI module is only enabled in low power modes only if the bit GENCS[STPE] is 1.

Only one electrode pin is functional in the low power mode scan and it's defined by the bit-field PEN[LPSP].

56.7.3.4.3 End-of-scan interrupt

The electrode scan unit sets the EOSF flag in the GENCS registers once all the active electrode scan finishes. The EOSF Flag generate an end-of-scan interrupt request if it is enabled.. The interrupt is asserted if enabled by GENCS[TSIIE] and GENCS[ESOR] bits.

The GENCS[EOSF] indicates that all active electrode scans are finished and the respective capacitance results are in the TSICHnCNT registers. The GENCS[EOSF] is cleared by writing one to it.

56.7.3.4.4 Over-run interrupt

If an electrode scan is in progress and there is a scan trigger the electrode scan unit generates and over-run error by asserting the GENCS[OVRF]. If the TSI error interrupt is active by setting the GENCS[ERIE] bit a interrupt request is asserted. The OVRF flag is cleared by writing 1 to it.

56.7.4 Touch detection unit

The touch detection unit is responsible to detect electrode capacitance changes while in low power mode.

It also detects the occurrence of error with the electrode in the case it capacitance result is 0x0000 or 0xFFFF. The errors can be cause by electrode pin short circuit to V_{DD} or V_{SS}. Or by electrode capacitances out of the configuration range of the TSI module.

56.7.4.1 Capacitance change threshold

Each TSI pin has it result register TSICHnCNT. In low power mode only one electrode can be active, at the end of the low power active electrode conversion the touch detection unit compares if the TSICHnCNT result value is inside a configurable range. The comparison range is defined individually registers, TSICHHTH, the upper threshold value and TSICHLTH, the lower threshold value. If the TSICHnCNT happens to be out of the range defined by TSICHLTH and TSICHHTH the GENCS[OUTRGF] flag is set indicating that a capacitance change occurred in the low power active electrode..

56.7.4.1.1 Out-of-range interrupt

The GENCS[OUTRGF] flag generates a TSI interrupt request if the GENCS[TSIIE] bit is set and GENCS[ESOR] bit is cleared. With this configuration, after the end-of-electrode scan, the TSI interrupt is only requested if there is a capacitance change. If the low power electrode capacitance does not vary, the TSI Interrupt do not interrupt the CPU.

56.7.4.2 Error interrupt

The GENCS[EXTERF] is set in the case the capacitance result registers, TSICHnCNT, of a TSI pin is either 0 or 0xFFFF, the two possible extreme values. The EXTERF flag generates a TSI Error Interrupt request if the GENCS[ERIE] bit is set.

56.8 Application information

After enable the TSI module for the first time, it is highly recommended a calibration to all the enabled channels by setting proper high and low threshold value for each active channel. All the channel dedicated counter values can be read from each counter value registers, software suite can then adjust the threshold based on these values.

Follow proper PCB layout guidelines for board design on electrode shapes, sizes, routes, etc. Visit www.freescale.com/touch for application notes and reference designs.

56.8.1 TSI module sensitivity

The TSI module sensitivity is defined by the increment cause in the TSICHnCNT result registers caused by a 1 pF delta in the electrode pin capacitance.

It is given by the following equation:

$$TSI_{sensitivity} = \frac{C_{ref} * I}{I_{ref} * PS * NSCN}$$

For the example provided, I_{ref} = 2 µA, PS = 2; NSCN = 16, C_{ref} = 1.0 pF and I =2 µA, the $TSI_{sensitivity}$ = 0.03125 pf/count

56.9 TSI module initialization

This section provides the recommended initialization sequence for the TSI module.

Prior to enable TSI module by setting TSI_GENCS[TSIEN] bit, it is required to configure other bits first. The pin enable registers are set to select which channels will be sampled, the dual oscillators configuration bits are set in order to make the scan and conversion more accurate. Also remember not to change the settings while TSI is working in progress. To switch from different scan modes, for instance, it is required to do a software reset to TSI by disabling and then enabling TSI_GENCS[TSIEN].

56.9.1 Initialization Sequence

Freescale TSS library has complete support for TSI, which make the configuration and application much easier. For detailed information on how to work with TSI and TSS together, visit www.freescale.com/touchsensing to get the application notes for details.

Chapter 57
JTAG Controller (JTAGC)

57.1 Introduction

NOTE

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

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.

57.1.1 Block diagram

The following is a block diagram of the JTAG Controller (JTAGC) block.

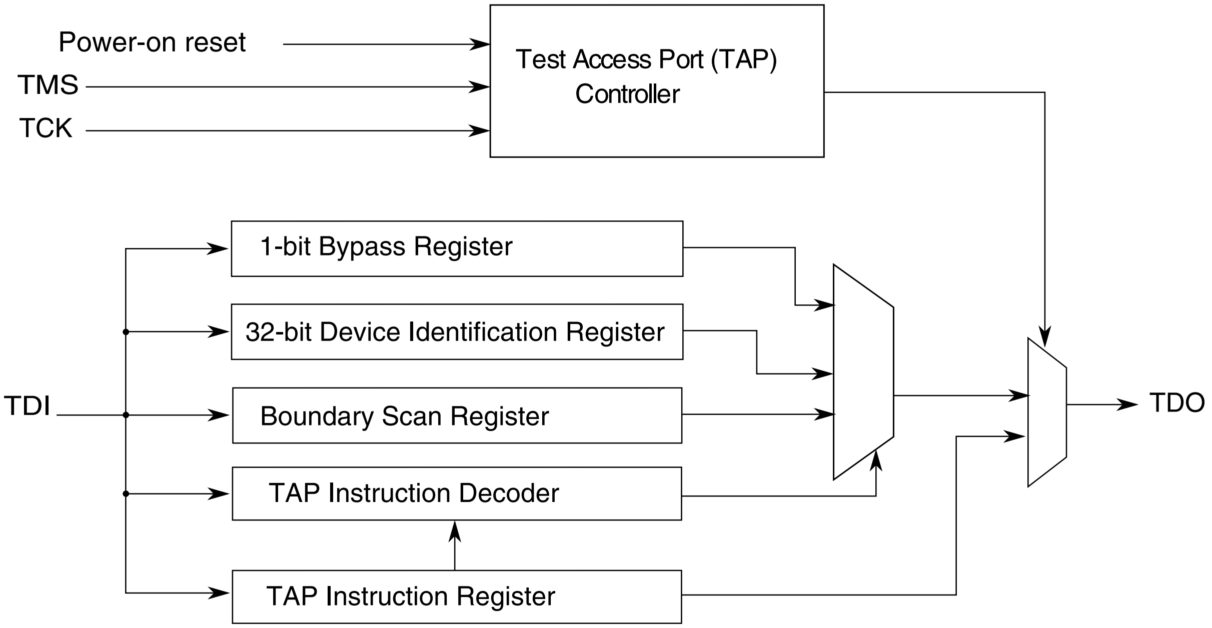


Figure 57-1. JTAG (IEEE 1149.1) block diagram

57.1.2 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
  - 4 pins (TDI, TMS, TCK, and TDO)
- Instruction register that supports several IEEE 1149.1-2001 defined instructions as well as several public and private device-specific instructions. Refer to Table 57-3 for a list of supported instructions.
- Data registers, 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.

57.1.3 Modes of operation

The JTAGC block uses 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.

57.1.3.1 Reset

The JTAGC block is placed in reset when either power-on reset is asserted, or the 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 results in asynchronous entry into the reset state. While 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

57.1.3.2 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 while 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 while 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 BYPASS, HIGHZ, CLAMP or reserved instructions are active. The functionality of each test mode is explained in more detail in JTAGC block instructions.

57.1.3.3 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. While 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.

57.2 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 57-1. JTAG signal properties

Name	I/O	Function	Reset State	Pull
TCK	Input	Test Clock	—	Down
TDI	Input	Test Data In	—	Up
TDO	Output	Test Data Out	High Z[1]	—
TMS	Input	Test Mode Select	—	Up

[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.

57.2.1 TCK—Test clock input

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

57.2.2 TDI—Test data input

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

57.2.3 TDO—Test data output

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is three-stateable 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.

57.2.4 TMS—Test mode select

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.

57.3 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.

57.3.1 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 while 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 0001b , 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 57-2. Instruction register

57.3.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the BYPASS, CLAMP, HIGHZ or reserve instructions are active. 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.

57.3.3 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 while 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.)				0	0	0	0	0	0	0	1	1	1	0	1

The following table describes the device identification register functions.

Table 57-2. Device identification register field descriptions

Field	Description
PRN	Part Revision Number. Contains the revision number of the part. Value is 0x0.
DC	Design Center. Indicates the design center. Value is 0x2C.
PIN	Part Identification Number. Contains the part number of the device. Value is TBD.
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.

57.3.4 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, and 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.

57.4 Functional description

This section explains the JTAGC functional description.

57.4.1 JTAGC reset configuration

While 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.

57.4.2 IEEE 1149.1-2001 (JTAG) Test Access Port

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 though 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.

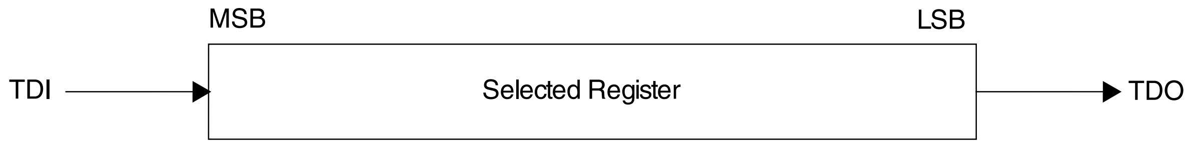
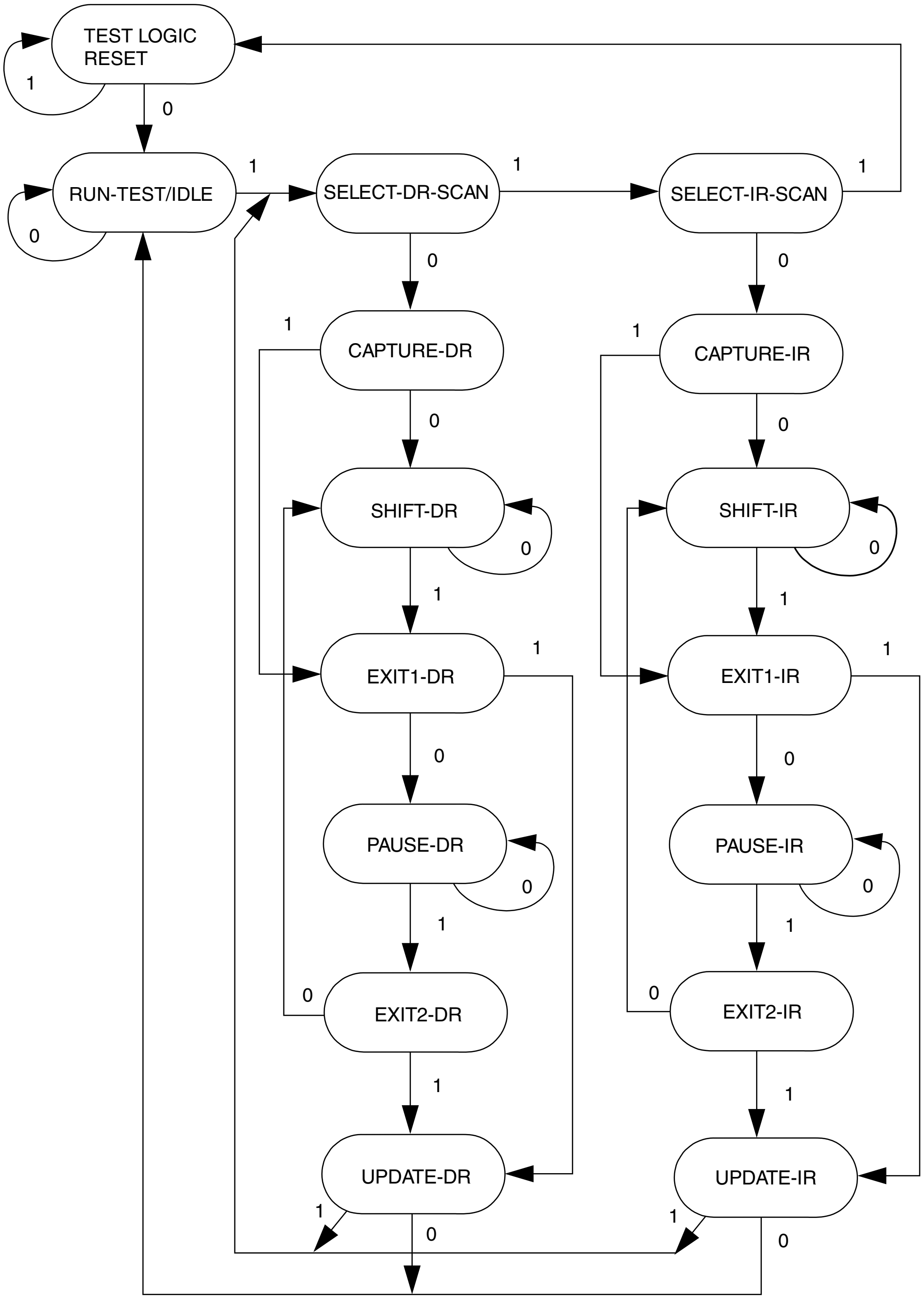


Figure 57-3. Shifting data through a register

57.4.3 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 following figure shows, holding TMS at logic 1 while 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 57-4. IEEE 1149.1-2001 TAP controller finite state machine

57.4.3.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting the JTAGC enable to a logic 1 value.

57.4.3.2 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 while 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 once the required number of bits have been acquired.

57.4.4 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; refer to the IEEE 1149.1-2001 standard for more details. All undefined opcodes are reserved.

Table 57-3. 4-bit JTAG instructions

Instruction	Code[3:0]	Instruction Summary
IDCODE	0000	Selects device identification register for shift
EZPORT	0001	Enables the EZPORT function for the SoC
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 while applying preloaded values to output pins and asserting functional reset
Factory debug reserved	0101	Intended for factory debug only
Factory debug reserved	0110	Intended for factory debug only
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 while three-stating all output pins and asserting functional reset
ARM JTAG-DP Reserved	1010	This instruction goes the ARM JTAG-DP controller. See the ARM JTAG-DP documentation for more information.

Table continues on the next page...

Table 57-3. 4-bit JTAG instructions (continued)

Instruction	Code[3:0]	Instruction Summary
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 while applying preloaded values to output pins and asserting 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

57.4.4.1 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.

57.4.4.2 EZPORT instruction

The EZPORT instruction allows for the EZPORT module to program the on-chip flash from a simple 4-pin interface. The JTAGC forces the core into a reset state and forces the EZPORT mode select/chip select low. In this mode, the flash can be programmed through the JTAG test port pins, which are connected to the EZPORT module.

57.4.4.3 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.

57.4.4.4 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.

57.4.4.5 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 while performing external boundary scan operations.

57.4.4.6 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. While HIGHZ is active all output drivers are placed in an inactive drive state (e.g., high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

57.4.4.7 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register while 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) while 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.

57.4.4.8 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. While the BYPASS instruction is active the system logic operates normally.

57.4.5 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.

57.5 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. Place the JTAGC in reset through TAP controller state machine transitions controlled by TMS
2. Load the appropriate instruction for the test or action to be performed

Appendix A
Release Notes for Revision 2

A.1 General changes throughout document

- No substantial content changes

A.2 About This Document chapter changes

- No substantial content changes

A.3 Introduction chapter changes

- Updated 'Kinetis MCU portfolio' diagram for K6x family.
- In 'Communication interfaces' section, updated USB OTG (low-/full-/high-speed) description.

A.4 Chip Configuration chapter changes

- Corrected reset value of AXBS_PRSn register.
- In Cache regions, updated FlexNVM address range.
- Changed OSCERCLK to OSC0ERCLK.
- In I2S Configuration, updated MCLK input clock selection encodings.
- Updated 'LLWU inputs' table.
- Updated 'DMA request sources - MUX 0' table for source numbers 36-39, 54-57.
- Updated ADC0 and ADC1 Channel Assignments tables for channel AD20.
- Clarified 'PDB Module Interconnections' section.
- In 'UART interrupts' section, updated LON and ISO7816 interrupt sources.
- Clarified 'I2S/SAI clock generation' section.
- Added note to 'VREF Overview' section.
- Added notes to USB controller and USB voltage regulator configurations.

- Clarified the "Wake-up Sources" section within the "Low-Leakage Wake-up Unit (LLWU) Configuration" section. Renamed the "LLWU inputs" table to "Wakeup sources for LLWU inputs", removed the multiplexed signals and provided the pin name only. Added a reference to the signal multiplexing table for the individual signal options.

• In ADCx Channel Assignment, updated AD27 and AD29 input signal description. • Clarified ADC and PGA Reference Options section. • In LPTMR pulse counter input options section, clarified LPTMR_CSR[TPS]=11 chip input. • Added notes to VREF Overview and DAC External Trigger Input Connections sections.
• Updated ADC and PGA Reference Options section.
• Updated Local memory controller region assignment section.

A.5 Memory Map chapter changes

• Added Alternate non-volatile IRC user trim description topic.
• In System Memory Map table, added Slave Port column. • Added note to 'FlexBus aliased areas' section. • Updated 'Peripheral bridge 0 slot assignments table' for slots 48 and 'Peripheral bridge 1 slot assignments table' for slot 47. • Added 'Alternate Non-Volatile IRC User Trim Description' topic.
• Updated system memory map, slave port, column.

A.6 Clock Distribution chapter changes

• Updated Clock Diagram for standard XTAL, EXTAL, and MCG clock names. • In Clock Definition table, added MCGPLL0CLKL, MCGPLL1CLK. • In Device Clock Summary table, updated bus clock, external reference clock, USB FS clock, and TRACE clock. • In Internal Clocking Requirements, updated bus clock. • In Module Clocks table, updated MCG. • Changed OSCERCLK to OSC0ERCLK as required. • Updated Debug Trace, USB FS, USB HS, TSI, and NFC clock diagrams.
• Added note to Debug trace clock section. • In Clock Summary table, added RTC_CLKOUT clock.
• Updated Debug trace clock diagram.
• Updated the "Internal clocking requirements" section. • For the "USB FS OTG Controller clocking" section, added this note: "The MCGFLLCLK does not meet the USB jitter specifications for certification."
• Clarified OSC clock names in Clock definitions section. • Updated USB HS controller clock generation section.

A.7 Reset and Boot chapter changes

• In System resets section, updated associated input pins for JTAG.
• For the section "Reset pin filter", added more information about the separate LPO filter in the LLWU.

A.8 Power Management chapter changes

• In 'Module Operation in Low Power Modes' section, changed LPT to LPTMR.

- In 'Entering and exiting power modes' section, updated wake-up flow from VLLSx.

A.9 Security chapter changes

- No substantial content changes

A.10 Debug chapter changes

- Updated the section "Debug Resets".

- For the section "Debug in Low Power Modes", removed "and trace" from "If the debug signal is active and the system attempts to enter stop or VLPS, FCLK continues to run to support core register access and trace."

A.11 Signal Multiplexing and Signal Descriptions chapter changes

- Updated pinout diagrams and tables

- In 'Port control and interrupt module features' section, updated digital filter clock cycles from 1 to 32.
- Updated CMPx_IN signals to 5:0.
- Added $\overline{\mathrm{UART0\_COL}}$ signal.

- In 'System Signal Descriptions' table, modified RESET_b pin to I/O.
- Updated module signal names from SAI_RX_DATA and SAI_TX_DATA to SAI_RX_DATA[1:0] and SAI_TX_DATA[1:0].
- In 'USB HS OTG Signal Descriptions' table, updated ULPI_CLK signal I/O direction.
- Updated 'Pinout' section.

A.12 PORT changes

- No substantial content changes

A.13 SIM changes

- For the SIM_SOPT1 register, updated the descriptions for the USBSSTBY and USBSTBY fields.
- For the SIM_SOPT6 register, removed the RSTFLTSEL and RSTFLTEN fields. The functions provided by these fields are now located within the RCM.

- Updated bank 2 registers addresses.
- Updated SOPT1 description, clarified 31-29 bit definitions. Added RAMSIZE. From bit 19, removed, LPTMR and MCG clock sources reference.
- Updated PLL clocks as PLL0 and PLL1.
- In SOPT2, updated USBF_CLKSEL field definition.
- In SOPT6, bits 31:24 are marked as reserved.
- In SDID, updated 6:0 bit fields to describe FAMID and PINID. Removed DIEID and marked it as reserved. Updated 9:7 bit fields reset value and encoding.
- In SCGC1, added OSC1.
- In SCGC2, changed ENET1 to ENET.
- In SCGC3, added FTM3 and FTM2.
- In SCGC4, changed bits 31:28, 5:4 as non-writable and always read as 1. Changed ANL to CMP.
- In SCGC5, changed bits 18, 8:7, and 1 as non-writable and always read as 1. Removed ATX and marked this bit as reserved.
- In SCGC6, changed USBOTG to USBHS. Changed FTF to FTFE with RW access.
- In FCFG1, changed FSIZE to NVMSIZE and PFSIZE.
- In FCFG2, updated MAXADDRxx bit field definitions. Also updated bits 31 and 23.
- In MCR, added definitions for bits 7:2.
- Updated EXTAL clock to OSC0ERCLK.

A.14 RCM changes

- For RPFC[RSTFLTSS], clarified the field description.

A.15 PMC changes

- No substantial content changes

A.16 LLWU changes

- No substantial content changes

A.17 MCM changes

- No substantial content changes

A.18 Crossbar switch chapter changes

- No substantial content changes

A.19 MPU changes

- No substantial content changes

A.20 AIPS-Lite changes

- No substantial content changes

A.21 DMAMUX changes

- No substantial content changes

A.22 DMA changes

Under the section "Dynamic programming", removed subsection, "Dynamic channel linking and dynamic scatter/gather", and replaced with new subsections, "Dynamic channel linking" and "Dynamic scatter/gather".

A.23 EWM changes

- No substantial content changes

A.24 WDOG changes

Clarification added for no reset due to unlock sequence when ALLOW_UPDATE is cleared in Section "Unlocking and Updating the Watchdog".

A.25 MCG changes

- Updated PLLs, OSC clocks, and bit field names to standardize as per 0, 1, 2 …order. Added initialization examples for 16 MHz external crystal.
- Updated CMEx, VDIVx, PRDIVx, PLLCLKENx bit field descriptions
- Updated MCG block diagrams for MCGFFCLK and MCGDDRCLK2X.
- Updated C1[CLKS] encoding and also updated C10 reset value.

A.26 OSC changes

- No substantial content changes

A.27 RTC Oscillator changes

- No substantial content changes

A.28 LMEM changes

- Updated Cache Features section and changed phrase ".....increase the average speed ......" to ".........decrease the average time....".

A.29 FMC changes

- No substantial content changes
- Terminology update: changed "Directory" to "Tag" in the names of tag cache registers.

A.30 FTFE changes

- In 'EEPROM Data Set Size' section, updated data flash IFR value.

A.31 EzPort changes

- No substantial content changes

A.32 NFC changes

- Removed all instances for NAND Flash Boot as it is not supported.

A.33 FlexBus changes

- No substantial content changes

A.34 CRC changes

No substantial content changes

A.35 ADC changes

- Removed CLPD from generating gain calibration values procedure.
- Updated Pseudo-code example section for CFG1 and SC2 register bits.
- Removed band gap voltages, BGH and BGL.

A.36 CMP changes

- Updated CMPx_CR1[PMODE] field description.

A.37 DAC changes

- Updated DACx_C0[LPEN] field description.
- Added note for 32/16 bit accesses to DACx_Cn and DACx_SR registers.

A.38 VREF changes

- No substantial content changes

A.39 PDB changes

Added Debug mode and updated PDBEN encodings.

A.40 FTM changes

- No substantial content changes

A.41 PIT changes

- No substantial content changes

A.42 LPTMR changes

- No substantial content changes

A.43 CMT changes

- No substantial content changes

A.44 RTC changes

Updated RTC_CR[14] bit field access.
Updated Time Alarm section with IER[TAIE].

A.45 USB changes

- No substantial content changes

A.46 USBDCD changes

- No substantial content changes

A.47 USB VREG changes

- No substantial content changes

A.48 USB high speed OTG controller changes

Updated these registers:

- Identification Register (USBHS_ID)
- General Hardware Parameters Register (USBHS_HWGENERAL)
- Host Hardware Parameters Register (USBHS_HWHOST)
- Transmit Buffer Hardware Parameters Register (USBHS_HWTXBUF)
- Receive Buffer Hardware Parameters Register (USBHS_HWRXBUF)
- System Bus Interface Configuration Register (USBHS_USB_SBUSCFG): new
- Host Controller Interface Version Register (USBHS_HCIVERSION)
- Capability Registers Length Register (USBHS_CAPLENGTH): moved-- The CAPLENGTH field is now included within the USBHS_HCIVERSION register.
- Host Controller Structural Parameters Register (USBHS_HCSPARAMS)
- USB Status Register (USBHS_USBSTS)
- USB Interrupt Enable Register (USBHS_USBINTR)
- Device Address Register (USBHS_DEVICEADDR)
- Host TT Asynchronous Buffer Control (USBHS_TTCTRL)
- Master Interface Data Burst Size Register (USBHS_BURSTSIZE)
- Endpoint NAK Register (USBHS_ENDPTNAK) new
- Endpoint NAK Enable Register (USBHS_ENDPTNAKEN) new
- Port Status and Control Registers (USBHS_PORTSC1)
- On-the-Go Status and Control Register (USBHS_OTGSC)
- USB Mode Register (USBHS_USBMODE)
- Endpoint Setup Status Register (USBHS_EPSETUPSR)

- Changed USBHS_USBCMD[ITC] access from read-only to read-write.
- Changed USBHS_TXFILLTUNING[TXSCHOH] from an 8-bit to a 7-bit field.

A.49 FlexCAN changes

- No substantial content changes

A.50 DSPI chapter changes

- In 'Transmit FIFO Fill Interrupt or DMA Request' section, added note on using TFFF flag.

- Added links to corresponding functional description in the Delay fields in CTAR register.
- Renamed DSICR to DSICR0.
- Updated EOQ interrupt request description.
- Added SPITCF and DSITCF interrupt request descriptions and updated corresponding bit fields in SR register.
- Updated DIS_TXF and DIS_RXF bit field descriptions in MCR register.
- Updated ‘Sample MSC downstream transmission using ITSB mode’ diagram.

A.51 I2C changes

- In the "Address Matching Wakeup" section, expanded the note to clarify the feature's purpose.

A.52 UART changes

- Changed TWFIFO[TXWATER] in RDRF register to RWFIFO[RXWATER].

- Updated S2[MSBF], S2[RXINV], C3[TXINV] and C7816[INIT] register bit field descriptions to mention that Initial Character Detect feature can only be used in TTYPE = 0 mode.
- Updated Initial Characters Section in 7816 Functional Description.
- Updated Protocol T=0 Section in 7816 Functional Description.
- Updated IE[ISDIE] register bit field description.

A.53 SDHC changes

- No substantial content changes

A.54 I2S/SAI changes

- Clarified the descriptions of the TCR2, TCR3, TCR4, TCR5, TDR, TMR, RCR2, RCR3, RCR4, RCR5, RDR, and RMR registers as well as the TCSR[FR], TCR2[BCI], TCR3[TCE], RCSR[FR], RCR2[BCI], and RCR3[RCE] bits.
- Corrected the bit-setting descriptions for the TCSR[FWIE], RCSR[FWIE], and RCSR[FEF] bits.

A.55 GPIO changes

- No substantial content changes

A.56 TSI changes

- No substantial content changes

A.57 JTAG Controller changes

- No substantial content changes

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductors products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claims alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-complaint and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to http://www.freescale.com/epp.

ARM POWERED®

Document Number: K20P144M120SF3RM
Rev. 2, Dec 2011

Preliminary

freescale™